NOVA SCHOOL OF SCIENCE & TECHNOLOGY

DEPARTAMENTO DE CIÊNCIAS SOCIAIS APLICADAS

HELDER MANUEL DA CONCEIÇÃO BARÃO MARTINS

Mestre em Ensino da Matemática no 3.º Ciclo do Ensino Básico e Secundário

UTILIZAÇÃO DA LINGUAGEM *PYTHON* PARA DESENVOLVER O PENSAMENTO COMPUTACIONAL NO ENSINO SECUNDÁRIO

MESTRADO EM EDUCAÇÃO, TECNOLOGIAS NO ENSINO DE CIÊNCIAS, TECNOLOGIA, ENGENHARIA E MATEMÁTICA (CTEM)

Universidade NOVA de Lisboa

Fevereiro, 2023

DEPARTAMENTO DE
CIÊNCIAS SOCIAIS APLICADAS

NOVA SCHOOL OF
SCIENCE & TECHNOLOGY

HELDER MANUEL DA CONCEIÇÃO BARÃO MARTINS

Mestre em Ensino da Matemática no 3.º Ciclo do Ensino Básico e Secundário

UTILIZAÇÃO DA LINGUAGEM *PYTHON* PARA DESENVOLVER O PENSAMENTO COMPUTACIONAL NO ENSINO SECUNDÁRIO

MESTRADO EM EDUCAÇÃO, TECNOLOGIAS NO ENSINO DE CIÊNCIAS, TECNOLOGIA, ENGENHARIA E MATEMÁTICA (CTEM)
Universidade NOVA de Lisboa
Fevereiro, 2023

DEPARTAMENTO DE CIÊNCIAS SOCIAIS APLICADAS

UTILIZAÇÃO DA LINGUAGEM *PYTHON* PARA DESENVOLVER O PENSAMENTO COMPUTACIONAL NO ENSINO SECUNDÁRIO

HELDER MANUEL DA CONCEIÇÃO BARÃO MARTINS

Mestre em Ensino da Matemática no 3.º Ciclo do Ensino Básico e Secundário

Orientador: Prof. Doutor António Manuel Dias Domingos,
Professor Auxiliar, FCT-NOVA

Júri:

Presidente: Prof. Doutor João José de Carvalho Correia de Freitas,
Professor Auxiliar, FCT-NOVA

Arguente: Prof. Doutor Jaime Carvalho e Silva,
Professor Associado, FCT-UC

Orientador: Prof. Doutor António Manuel Dias Domingos,
Professor Auxiliar, FCT-NOVA

MESTRADO EM EDUCAÇÃO, TECNOLOGIAS NO ENSINO DE CIÊNCIAS, TECNOLOGIA, ENGENHARIA E MATEMÁTICA (CTEM)

Universidade NOVA de Lisboa
Fevereiro, 2023

Utilização da linguagem *Python* para desenvolver o pensamento computacional no Ensino Secundário

À Gabriela, ao Vasco e à Inês

AGRADECIMENTOS

Ao Professor Doutor António Manuel Dias Domingos pelas orientações, críticas e sugestões manifestadas ao longo do período em que decorreu a elaboração desta dissertação, bem como pela sua total disponibilidade e entrega.

Ao Professor Doutor Carlos Albuquerque por ter efetuado uma revisão do texto e dado sugestões de melhoria deste trabalho.

A todos aqueles que, através da sua ação, contribuíram direta ou indiretamente, para que este trabalho fosse executado, em particular aos colegas Lisete Pires e Manuel Marques e aos alunos que partiparam.

À minha família pelo tempo que lhes tomei com a minha ausência.

"Se eu vi mais longe, foi por estar sobre ombros de gigantes." (Isaac Newton)

RESUMO

A experiência de ensino que serve de base a esta tese de mestrado resultou de um trabalho de interdisciplinaridade entre as disciplinas de Matemática A e de Aplicações Informáticas B. É um estudo de natureza qualitativa, seguindo um paradigma interpretativo, de cariz empírico, em que se pretende implementar estratégias de ensino e de aprendizagem sendo que o investigador é participante ativo no processo.

A fundamentação teórica deste trabalho baseia-se em dois campos do conhecimento: Pensamento Computacional, expressão atribuída a Wing (2006), e Raciocínio Matemático, com raízes na Grécia Antiga com o desenvolvimento do raciocínio dedutivo, nomeadamente da lógica. São feitas igualmente considerações sobre o entendimento que se faz da utilização da tecnologia nos programas de Matemática do Ensino Básico e da Matemática A do Ensino Secundário, no sistema de ensino em Portugal, e é descrito o que se pretende com a introdução da linguagem de programação *Python* na Matemática do Ensino Secundário

As questões que se colocaram tiveram por base uma abordagem de desenvolvimento curricular com integração da tecnologia e do *Python* e foram:

- Como construir tarefas que incorporem o Pensamento Computacional (PC), potenciadoras do raciocínio matemático no Ensino Secundário?
- Como se processa a aprendizagem da Matemática quando se implementam tarefas que envolvem o PC?
- De que forma se materializam modelos de ensino e aprendizagem, no domínio da Matemática, recorrendo à linguagem de programação *Python*?

Assim, neste trabalho, e recorrendo a um ambiente de resolução de problemas, os alunos foram incentivados a desenvolver programas em computador que utilizassem a linguagem *Python*, possibilitando que aprendessem e consolidassem conceitos matemáticos de uma forma mais prática e dinâmica.

Esta experiência foi aplicada numa turma de vinte e oito alunos no 12.º ano de escolaridade, do Curso Científico-Humanístico de Ciências e Tecnologias, na área da grande Lisboa, tendo tido o seu início em setembro de 2021, com reuniões entre os professores de cada uma das duas disciplinas, de forma a preparar o trabalho de campo que se veio a concretizar em três aulas na disciplina de Aplicações Informáticas B em meados do 2.º período do ano letivo de 2021/22, e que teve o seu desfecho em maio de 2022.

Nas aulas de Aplicações Informáticas B foram implementados três guiões de aprendizagem de iniciação à linguagem de programação *Python e* nas aulas de Matemática A foram lecionados os conceitos que deram consistência às tarefas que foram implementadas.

De entre os resultados obtidos, verificou-se que os alunos desenvolveram diversos tipos de abordagens com recurso à linguagem de programação *Python,* conseguindo interligar conceitos que aprenderam em Matemática A.

Palavras chave: Pensamento Computacional; Raciocínio Matemático; Ensino Secundário.

ABSTRACT

The teaching experience that serves as the basis for this master's thesis resulted from an interdisciplinary work between the disciplines of Mathematics A and Computer Applications B. It is a qualitative study, following an interpretative paradigm, of empirical nature, which aims to implement teaching and learning strategies and the researcher is an active participant in the process.

The theoretical foundation of this work is based on two fields of knowledge: Computational Thinking, expression attributed to Wing (2006), and Mathematical Reasoning, with roots in Ancient Greece with the development of deductive reasoning, namely logic. Considerations are also made on the understanding of the use of technology in the Mathematics syllabus of Basic School Mathematics and Mathematics A of Secondary School, in the Portuguese educational system, and a description is given of what is intended by the introduction of Python programming language in Secondary School Mathematics.

The questions asked were based on a curriculum development approach integrating technology and Python and were:

- How to construct tasks that incorporate Computational Thinking (CP), which enhance mathematical reasoning in Secondary Education?
- How does mathematics learning occur when implementing tasks that involve CP?
- How do we materialize teaching and learning models, in the domain of Mathematics, using the Python programming language?

Thus, in this work, and using a problem-solving environment, students were encouraged to develop computer programs using the Python language, allowing them to learn and consolidate mathematical concepts in a more practical and dynamic way.

This experiment was applied to a class of twenty-eight students in the 12th grade of the Science and Technology course, in the Greater Lisbon area, which began in September 2021, with meetings between the teachers of each of the two subjects, in order to prepare the field-work that was to take place in three classes in the Computer Applications B subject in the middle of the 2nd period of the school year 2021/22, and which ended in May 2022.

In the Computer Applications B classes, three learning guides were implemented to introduce the Python programming language, and in the Mathematics A classes the concepts that gave consistency to the tasks that were implemented were taught.

Among the results obtained, it was observed that the students developed several types of approaches using the Python programming language, being able to connect concepts they had learned in Mathematics A.

Keywords: Computational Thinking; Mathematical Reasoning; Secondary Education.

ÍNDICE

ÍNDICE DE FIGURAS

ÍNDICE DE TABELAS

GLOSSÁRIO

Linguagens de programação

São linguagens utilizadas para descrever programas, escritos por um utilizador, podendo ser interpretadas num computador, *tablet*, calculadora ou *smartphone*. Por norma, as linguagens de programação possuem um conjunto de comandos relativamente pequeno e uma sintaxe rígida, para que não exista qualquer ambiguidade no momento de execução dos comandos. Exemplos de linguagens atualmente utilizadas são: *Python*, *Javascript* e *C*.

Algoritmo

É uma sequência de instruções que se seguem para a realização de um determinado programa ou tarefa. Exemplos: cálculo da área de um triângulo dadas as medidas da base e da altura, factorização de um número natural em fatores primos, receita de um bolo, etc.

Variável

Na Matemática, a noção de variável tem uma utilização ligeiramente diferente da que é dada em programação. Em Matemática, uma variável representa um elemento genérico de um dado conjunto (dito universo ou domínio da variável), sendo normalmente designada por uma letra minúscula (x, y, t, etc.). No contexto da programação, uma variável é um espaço que é alocado na memória do computador que permite armazenar uma dada informação de um determinado tipo. Alguns dos tipos mais comuns são: números inteiros; números com parte decimal; valores booleanos (verdadeiro ou falso); e, cadeias de carateres.

Expressão Condicional

É um dos conceitos mais utilizados em qualquer linguagem de programação. Consiste numa instrução ou comando que permite direcionar os passos seguintes a serem executados pelo computador, conforme a condição seja, ou não, verdadeira para um determinado valor. Normalmente, expressa-se no formato 'se condição, então faz isto, senão faz aquilo'.

Exemplo: Pense-se num algoritmo que identifique se um dado número natural é par ou ímpar. Tem que considerar-se a seguinte condição: se o número é divisível por 2 então é 'verdade'. Caso contrário é 'falso'. Esquematicamente, pode-se escrever da seguinte forma:

```
se (número é divisível por 2) então:
    escreva(“verdade”)
senão:
    escreva(“falso”)
```

As estruturas de repetição são essenciais em programação. Embora possam existir ligeiras variações no formato, estas seguem a seguinte sintaxe: ‘repetir as seguintes ações enquanto determinada condição for verdadeira’. Por exemplo, se quisermos somar todos os números naturais de 1 até 60, inclusive, e obter o resultado, podemos esquematizar este processo através do algoritmo seguinte:

Estruturas de repetição

```
soma=0
num=1
enquanto (num<=60) faz:
    soma = soma + num
    num = num+1
escrever(soma)
```

Note-se que estas ações serão repetidas enquanto o valor da variável ‘num’ for menor ou igual a 60. No final, o conteúdo da variável soma será escrito no visor.

.

SIGLAS

DGE	Direção Geral de Educação.
ME	Ministério da Educação.
MIT	Massachusetts Institute of Technology
NCTM	National Council of Teachers of Mathematics.
NGSS	Next Generation Science Standards.
OECD	Organisation for Economic Cooperation and Development.
PISA	Programme for International Student Assessment.

SÍMBOLOS

**	Potenciação em *Python.*
//	Quociente da divisão inteira em *Python*.
%	Resto da divisão inteira em *Python*.

1

INTRODUÇÃO

Vivemos momentos na história da humanidade em que o ser humano é obrigado, de forma insistente e permanente, a resolver problemas de uma forma criativa e quase imediata. Compete ao sistema de ensino, agindo como um todo, desenvolver ferramentas pedagógicas que possibilitem que os alunos adquirem processos de raciocínio que lhes permitam no futuro, na sua vida corrente, ultrapassar as dificuldades e constrangimentos que irão enfrentar.

Os resultados obtidos na investigação matemática no final do século passado e no início deste século sugerem que não se pode continuar a ensinar matemática utilizando sempre os mesmos processos de mecanização de procedimentos (NCTM, 2017), obrigando continuamente os alunos a efetuar exercícios repetitivos e fastidiosos.

O ensino da matemática escolar tem que ser encarado sob uma dupla perspetiva. Se por um lado deve atender à maturidade do aluno e ao desenvolvimento cognitivo deste, deverá igualmente ter significado no conteúdo que se pretende transmitir.

Entre as alternativas que têm sido exploradas ao nível do ensino secundário acentua-se a utilização de sistemas algébricos computacionais (Computer Algebra Systems – CAS) e o desenvolvimento do pensamento computacional (PC) através do recurso a algoritmos de programação (NCTM, 2007), como formas de ajudar os alunos na resolução de problemas matemáticos e assim permitir o desenvolvimento de competências matemáticas.

1.1 Pertinência do estudo

Diversos estudos internacionais, entre os quais se destaca o estudo PISA (Programme for International Student Assessment), promovido pela OCDE (Organização para a Cooperação e Desenvolvimento Económico), de periodicidade trianual, que avalia se os alunos que estão a terminar o ensino básico e a iniciar o ensino secundário, com 15 anos de idade, adquiriram o conhecimento e as competências consideradas essenciais para uma participação

adequada e plena em sociedade, bem como se conseguem utilizar o que aprendem em situações de âmbito real, permitem concluir que é necessário proporcionar aos alunos alguma diversidade de situações em sala de aula, nomeadamente a análise de casos da vida real de forma a que os alunos aprendam Matemática recorrendo a processos mentais mais reflexivos e criativos (Carvalho e Silva, Canavarro, Albuquerque, Mestre, Martins, Almiro, Santos, Gabriel, Seabra & Correia, 2020).

O estudo PISA sugere igualmente a criação de ambientes de aprendizagem favoráveis, que fomentem a cooperação entre os alunos; o envolvimento dos pais nas atividades escolares; a disponibilização adequada de recursos fundamentais e uma maior colaboração entre os professores como aspetos fundamentais para a promoção de melhorias no ensino em geral e da Matemática em particular (Carvalho e Silva et al., 2020).

O PISA 2018 (OECD, 2019) destaca como fundamentais para a apreensão da literacia matemática, contextualizada numa adequada formulação, aplicação e interpretação da matemática, os seguintes processos matemáticos: saber formular matematicamente as situações; saber aplicar conceitos, factos, procedimentos, e raciocínio matemáticos; e, saber interpretar, aplicar e avaliar resultados matemáticos.

A introdução do PC na disciplina de Matemática no ensino secundário, afigura-se assim como um processo que pode contribuir para desenvolver nos alunos a necessidade de serem mais proficientes, o que para o PISA significa que os alunos consigam estudar situações onde têm a necessidade de conceptualizar, generalizar e utilizar informação, baseados nas suas investigações e na modelação de problemas complexos, além de permitirem o estabelecimento de relações matemáticas simbólicas e formais para desenvolver novas abordagens e estratégias que lhes permitam lidar com situações novas (OECD, 2019).

Como refere o estudo de Carvalho e Silva et al. (2020), citando o IFTF (Institute For The Future), uma das competências chave a desenvolver nos alunos de forma a que estes se consigam adaptar adequadamente ao mercado de trabalho no futuro é precisamente o PC, aqui encarado como a capacidade de analisar uma grande quantidade de dados e sintetizá-los de forma a criar modelos que traduzam a realidade o mais aproximadamente possível de forma a se conseguir prever a evolução de uma determinada situação real.

A expressão PC é da autoria de Wing (2006) tendo sofrido algumas mutações desde a sua criação. Para Wing (2006), “à leitura, à escrita, e aritmética, devemos acrescentar o pensamento computacional à capacidade analítica de cada criança.” (p. 33)

No entender de Bocconi et al. (2016), o PC deve ser encarado como uma metodologia de resolução de problemas associada a um determinado conjunto de conceitos e

competências como sejam a abstração, o pensamento algorítmico e a decomposição estruturada de problemas. Para estes autores o PC não deve ser confundido com competências digitais ou literacia digital que são conceitos mais latos.

Este termo aparece na maioria das vezes associado à algoritmia e à programação, no entanto é mais vasto, sendo que a algoritmia e a programação são simplesmente duas formas de expressão do PC.

Não se pretende que na disciplina de Matemática se ensine programação como um fim em si mesmo, como se de uma disciplina de Informática se tratasse, sendo que esta última disciplina existe no currículo do ensino secundário português atualmente. O que se pretende é que os alunos pensem matematicamente e utilizem a algoritmia e a programação como ferramentas para aprender Matemática de uma forma mais rica e dinâmica como preconiza o NCTM (2007).

Não sendo o PC um exclusivo do pensamento matemático este deve estar igualmente integrado noutras disciplinas curriculares e não curriculares de forma a dar-lhe contexto e significado adotando-se abordagens adequadas para cada nível de ensino.

Como referem Bocconi et al. (2016) o PC já se encontra integrado em currículos de Matemática em diversos países como sejam a França, a Finlândia, Portugal e noutros países, sendo que a razão principal para a sua inclusão se deve ao desenvolvimento das competências para o século XXI, no entanto, o que se verifica, nomeadamente em Portugal, é uma certa timidez na sua utilização, sendo feitas algumas aplicações no sistema educativo com pouca expressão.

Martins et al. (2017), referem, no documento de apresentação do *Perfil dos Alunos à Saída da Escolaridade Obrigatória* que os alunos devem "executar operações técnicas, segundo uma metodologia de trabalho adequada, para atingir um objetivo ou chegar a uma decisão ou conclusão fundamentada, adequando os meios materiais e técnicos à ideia ou intenção expressa". (p. 29)

No documento das Aprendizagens Essenciais da disciplina de Matemática para o 7.º ano do 3.º ciclo do ensino básico que foi aprovado em agosto de 2021, refere-se o seguinte:

> O pensamento computacional pressupõe o desenvolvimento, de forma integrada, de práticas como a abstração, a decomposição, o reconhecimento de padrões, a análise e definição de algoritmos, e o desenvolvimento de hábitos de depuração e otimização dos processos. Estas práticas são imprescindíveis na atividade matemática e dotam os alunos de ferramentas que lhes permitem resolver problemas, em especial relacionados com a programação. (Canavarro et al., 2021, p. 3)

E, em particular, no que respeita à algoritmia:

> Promover o desenvolvimento de práticas que visem estruturar, passo a passo, o processo de resolução de um problema, incentivando os alunos a criarem algoritmos que possam descrever essas etapas, nomeadamente com recurso à tecnologia, promovendo a criatividade e valorizando uma diversidade de resoluções e representações que favoreçam a inclusão de todos. (Canavarro et al., 2021, p. 15)

Por outro lado, nas Aprendizagens Essenciais da Matemática A para o Ensino Secundário – 12.º Ano, é possível encontrar referências à utilização da tecnologia, nomeadamente para "programar, criar e implementar algoritmos" (DGE, 2018, p. 5), como práticas essenciais de aprendizagem.

Conclui-se assim que documentos curriculares já existem em Portugal, falta somente a criação de uma metodologia e de alguma orientação de como desenvolver em concreto o PC em sala de aula, nomeadamente no ensino secundário.

1.2 Formulação do problema

Como foi referido, Portugal é um país onde os decisores do currículo para a disciplina de Matemática tiveram a preocupação de integrar o PC nos conteúdos programáticos vigentes para o ensino obrigatório.

Para Espadeiro (2022):

> O pensamento computacional poderá ser entendido como o processo de pensamento que envolve a formulação de problemas e os meios para alcançar as suas soluções, de forma a que a sua representação permita que estas ações possam ser realizadas por um agente de processamento de informações. (p. 5)

Na prática, e tal como refere Wing (2006), "o pensamento computacional é reformular um problema aparentemente difícil num que nós consigamos resolver, talvez por redução, incorporação, transformação ou simulação." (p. 33)

Um dos processos mais utilizados para fomentar o PC nos alunos passa por ensiná-los a programar, constituindo-se este como uma forma de ensinar Matemática num determinado contexto e criar deste modo ambientes informais que tenham significado real para cada aluno. Não se pretende ensinar a programar como um objetivo em si mesmo, mas sim dar aos alunos ferramentas que lhe possibilitem, de uma forma estruturada, resolver um determinado problema.

Entende-se aqui por problema uma tarefa em que cada aluno terá de fornecer instruções a uma máquina, quer esta seja um computador, um tablet, uma calculadora ou outro aplicativo, de forma a que estes utilitários as entendam e as executem.

Uma das vantagens de programar consiste no facto de estar a preparar os alunos para o seu futuro e desenvolverem assim competências que lhes permitam trabalhar com os computadores. É uma competência que tal como Wing (2006) afirma será tão ou mais necessária como saber ler, escrever ou contar.

Para Prensky (2012), numa sociedade de "nativos digitais", a literacia afeta à programação é a única competência que irá no futuro distinguir uma pessoa alfabetizada de outra não alfabetizada, constituindo-se assim como uma ferramenta essencial para o cidadão do século XXI.

Por outro lado, as linguagens de programação e a Matemática têm muito em comum, nomeadamente por obedecerem a uma lógica bivalente e por ambas se regerem por um conjunto de regras e de símbolos bem definido.

A programação pode assim ser utilizada como uma porta de entrada para a descoberta da Matemática, das suas diversas componentes, nomeadamente através da construção de pequenos programas que permitam ao aluno refletir sobre a realidade que o circunda como sejam, por exemplo: programas em que lhes sejam solicitados o valor a pagar por um determinado empréstimo, sujeito a uma taxa de juro fixa ou variável, ao fim de um determinado tempo; ou a aplicação da fórmula de uma área ou de um perímetros de uma determinada figura geométrica dados determinados parâmetros; ou ainda a construção de programas ou rotinas que permitam visualizar determinados aspetos da Matemática tal como a determinação de soluções aproximadas para uma dada equação ou aspetos mais gráficos, como sejam por exemplo os fractais, etc.

Se é verdade que em meados do século XX a Matemática que era exigida a um estudante nos primeiros anos de ensino era uma Matemática em que bastaria este saber as quatro operações básicas: somar, subtrair, multiplicar e dividir, nos dias de hoje um aluno deverá saber algo mais e exige-se uma maior preparação deste para os problemas que ainda ninguém sabe muito bem quais serão no futuro.

O aluno mais do que aprender a memorizar conceitos e regras deve aprender a pensar, nomeadamente deve aprender a efetuar estimativas rápidas entre ordens de grandeza diversas de modo a saber detetar eventuais erros criados pelas máquinas e desta forma dar resposta a problemas que lhe sejam colocados na sua vida do dia a dia, na sua fase adulta. É verdadeiramente isso que está em causa com a introdução do PC nas escolas. É somente

uma desculpa para «obrigar» o aluno a raciocinar, tendo por base o que observa no seu meio. Mais do que uma forma de ensinar é uma forma inclusiva de aprender.

Neste contexto e para os efeitos deste trabalho de investigação, escolheu-se uma turma de 12.º ano para implementar alguns conceitos da linguagem de programação *Python*, tendo por suporte as Aprendizagens Essenciais de Matemática A (DGE, 2018).

A razão pela qual a escolha da linguagem de programação incidiu sobre o *Python* deve-se ao facto de esta linguagem ser open source, de fácil utilização e ser utilizada num variadíssimo número de aplicações. De destacar que existem outras linguagens de programação que poderiam igualmente ser utilizadas para desenvolvimento do PC, como, por exemplo: *C*, *C++*, *Scratch*, *JavaScript* ou *SQL*, no entanto o *Python* tem bastantes aplicações utilizáveis em contextos educativos.

Tal como é referido nos Princípios para a Ação (NCTM, 2017), citando autores como Fuson, Kalchman, Bransford e Martin, só se consegue obter uma boa fluência procedimental se houver uma boa compreensão conceptual, nomeadamente "Quando os conceitos são relacionados com conceitos subjacentes, os alunos retêm melhor os procedimentos e ficam mais aptos a aplicá-los em novas situações" (NCTM, 2017, p. 42). Nesse sentido, a utilização do *Python* permite que os alunos aprendam e percebam as ideias matemáticas, raciocinem matematicamente e comuniquem o seu raciocínio.

1.3 Objetivos a atingir e questões de investigação

Este estudo pretende-se constituir como um meio privilegiado para analisar possíveis contribuições do PC aplicado a alunos a frequentar o ensino secundário utilizando a linguagem de programação *Python*. Assim, são objetivos deste estudo a análise e a reflexão sobre tarefas a implementar em sala de aula a alunos que estão inscritos na disciplina de Matemática A do 12.º ano de escolaridade do sistema de ensino em Portugal e que estão simultaneamente a frequentar a disciplina de Aplicações Informáticas B.

Trata-se de um estudo em que se pretende implementar estratégias de ensino e de aprendizagem resultante de uma parceria entre as disciplinas de Matemática A e de Aplicações Informáticas B. Se por um lado nas aulas de Aplicações Informáticas B, nomeadamente em três aulas de 90 minutos, foram implementados guiões de aprendizagem de iniciação à linguagem de programação *Python*, nas aulas de Matemática A foram lecionados os conceitos que deram consistência às tarefas implementadas na disciplina de Aplicações Informáticas B.

A investigação (Papert, 1980 e 1991; Ackermann, 2001), sugere que a utilização de uma linguagem de programação permite que todos os alunos, mesmo aqueles com maiores

dificuldades de aprendizagem, consigam idealizar e executar programas ou rotinas de programação algo complexas desenvolvendo deste modo o gosto pela Matemática como um todo integrado.

Nesse sentido foi solicitado aos alunos que, utilizando a linguagem *Python*, desenvolvessem em sala de aula programas, que possibilitassem, por exemplo, a determinação do declive de uma reta dados dois pontos; executassem a fórmula resolvente para equações do 2.º grau ou determinassem um zero de uma dada função, utilizando para tal o Método de Newton-Raphson.

Num ambiente de resolução de problemas, e utilizando o computador e a linguagem *Python*, tentou adotar-se determinados procedimentos que permitissem aprender Matemática utilizando processos mais dinâmicos e menos analíticos.

As questões de investigação que se colocam têm por base uma abordagem de desenvolvimento curricular com integração da tecnologia e do *Python* e são:

- Como construir tarefas, relacionadas com o Pensamento Computacional (PC), potenciadoras do raciocínio matemático no Ensino Secundário?
- Como implementar tarefas que impliquem a utilização do PC na aprendizagem da Matemática?
- De que forma se materializam modelos de ensino e aprendizagem, no domínio da Matemática, recorrendo à linguagem de programação *Python*?

Estas são três questões às quais este estudo pretende dar resposta, de forma a impulsionar e implementar uma prática letiva promotora de aprendizagens significativas nos alunos.

1.4 Organização do documento

Este trabalho encontra-se organizado em sete capítulos, sendo que não devem ser interpretados como totalmente independentes uns dos outros.

O presente capítulo pretende introduzir o tema, onde são colocados as questões de investigação e os objetivos do estudo.

No capítulo dois faz-se uma revisão da literatura, a qual incide na análise do que é o PC e qual tem sido a sua evolução ao longo da história, nomeadamente desde os finais do século XX até ao início da terceira década deste século. Nesta parte é feita igualmente uma explicação do que se entende por raciocínio matemático nas suas três componentes: raciocínio dedutivo, indutivo e abdutivo. Neste segundo capítulo faz-se ainda uma análise do uso da

tecnologia no ensino da Matemática, e é analisada a forma como esta é encarada nos programas de Matemática portugueses atualmente e como o *Python*, enquanto linguagem de programação, pode ser um artefacto apropriado para a aprendizagem da Matemática escolar.

No terceiro capítulo deste documento é apresentada a metodologia que serviu de base à apresentação dos guiões e das tarefas que foram propostos aos alunos.

No quarto capítulo são apresentados o contexto educativo em que se inseriram os alunos, bem como os alunos e os professores que fizeram parte do estudo.

No capítulo cinco são relatados os resultados da investigação efetuada em sala de aula.

No sexto capítulo são definidos os doze estudos de caso, que resultaram das tarefas elaboradas pelos alunos no seguimento da aplicação dos guiões de trabalho.

Por último, no sétimo capítulo, são apresentadas as respostas às três questões colocadas na introdução, bem como são apresentadas algumas considerações finais sobre este trabalho.

2

REVISÃO DE LITERATURA

O Pensamento Computacional (PC), tal como foi referido anteriormente, é interpretado por Wing (2006) como mais uma competência que cada criança deve possuir, a acrescentar à leitura, à escrita e à aritmética, de forma a reforçar a sua capacidade de análise, devendo ser encarada por todos os professores como um recurso que pode ser utilizado para resolver problemas, não devendo ser uma ferramenta útil somente para os engenheiros informáticos.

Neste contexto torna-se imperioso que o aluno tome igualmente contacto com os diversos tipos de Raciocínio Matemático, quer este seja do tipo dedutivo, indutivo ou abdutivo.

Por outro lado, e atendendo ao objetivo deste trabalho, torna-se necessário efetuar algumas considerações sobre a utilização da tecnologia na disciplina de Matemática, quer do ensino básico, quer do ensino secundário, e descrever como a introdução da linguagem de programação *Python* na Matemática do ensino secundário pode ser um elemento benéfico.

2.1 Pensamento computacional

Para Bocconi et al. (2016), o PC é um processo de pensamento que recorre a abordagens analíticas e algorítmicas para formular, analisar e resolver problemas. Mais do que saber efetuar um determinado programa com um conjunto de instruções que conduzam a um determinado objetivo, pretende-se com o PC resolver problemas e nesse campo há que ter em conta, além do conjunto de instruções que se fornecem à máquina, os recursos, as limitações e a plataforma que se utiliza.

Com o PC parte-se de um problema suficientemente amplo e aparentemente difícil e, utilizando o raciocínio heurístico, transforma-se noutro problema, reformulando-o através de processos de “redução, incorporação, transformação ou simulação” (Wing, 2006, p. 33).

O PC assenta numa boa conceção, antes mesmo da programação, pois tal requer a reflexão sobre variadíssimas formas de abstração; na competência, sem que tal implique necessariamente a mecanização, pois esta última induz rotinas que podem ser nefastas à criação de processos criativos; na compreensão da forma como os humanos raciocinam, usando a imaginação e a inteligência; na utilização de técnicas adstritas ao raciocínio matemático e de engenharia, uma vez que os sistemas que se constroem têm que interagir com o mundo real; no trabalho com ideias em detrimento dos artefactos; e, na generalização universal das ideias inerentes a este tipo de pensamento.

O PC está na base da aquisição do saber científico pois tal como refere Wing (2006), "Os problemas científicos intelectualmente desafiantes e motivantes continuam por compreender e resolver. O domínio do problema e o domínio da solução são limitados apenas pela nossa própria curiosidade e criatividade." (p. 35)

Para Espadeiro (2022), referindo-se às Aprendizagens Essenciais da Matemática em Portugal para o ensino básico, o PC deve ser encarado no ensino "como uma das 6 capacidades matemáticas (…), a par com a resolução de problemas, o raciocínio matemático, a comunicação matemática, as representações e as conexões matemáticas." (p. 5)

Para a execução do PC é necessário, segundo Espadeiro (2022), que tem por base Wing (2006), contemporizar 5 práticas associadas, a saber: abstração, decomposição, reconhecimento de padrões, algoritmia e depuração.

- A abstração, entende-se como a identificação de princípios gerais que podem ser aplicados em tarefas similares.
- A decomposição implica sintetizar a solução de um dado problema em partes mais simples, sem perder a informação relevante do problema inicial.
- O reconhecimento de padrões compreende a identificação de regularidades ou relações entre os objetos, nomeadamente a visualização de detalhes relevantes que ligam toda a estrutura que se está a investigar.
- A algoritmia permite a criação de etapas de resolução de um problema que conduzam à solução desse mesmo problema, podendo ser escrito em pseudocódigo ou em linguagem corrente.
- A depuração, consiste maioritariamente na identificação e correção de erros da resolução apresentada, podendo igualmente permitir a testagem, verificação, refinamento ou otimização da resolução apresentada. (Espadeiro, 2022)

Toda esta conceção do levar à prática o PC implica uma ideia de escola mais inclusiva e mais agregadora, em que o ser humano tem que estar preparado para as mudanças no que respeita ao conhecimento, devendo possuir pensamento crítico e criativo de forma a adaptar-se quando confrontado com situações algo inesperadas, tudo ideias desenvolvidas por Resnick (2008), em que segundo este autor a sociedade, como um todo, tem que ser mais criativa,

de modo a saber dar resposta aos seus problemas atendendo à velocidade a que tudo se transforma.

Pode-se colocar igualmente a questão de se saber se o desenvolvimento do PC só se consegue com recurso à tecnologia, no entanto tal não é verdade, pois como já foi referido anteriormente existem técnicas utilizadas no raciocínio matemático que não dependem desse uso mas sim da utilização da imaginação e da inteligência. Podem-se conceber tarefas bem estruturadas que permitam a obtenção de uma determinada solução sem que seja necessário efetuar um programa num computador, tablet ou calculadora, bastando para tal idealizar um algoritmo bem definido.

E mesmo quando se utiliza um computador para desenvolver este tipo de raciocínio não se tem que ter necessariamente um computador por aluno. Pode-se perfeitamente ter um computador para 2 ou 3 alunos e desenvolver igualmente a competência de trabalho colaborativo, considerada como fundamental para o século XXI, a par com:

- Saber avaliar e selecionar informação.
- Formular hipóteses e tomar decisões fundamentadas no dia a dia.
- Pensar de forma criativa, crítica e autónoma.
- Saber comunicar.
- Tomar gosto por aprender ao longo da vida. (Martins et al, 2017)

O PC constitui-se assim como um elemento fundamental para desenvolver competências adstritas à Matemática e não só nos alunos que se encontram na escolaridade obrigatória e simultaneamente criar condições para a formação integral das mulheres e homens no século XXI.

2.1.1 Evolução do conceito.

Na década de 60 do século XX, o objetivo principal da disciplina de Matemática no ensino básico consistia em preparar os alunos para trabalhar com algoritmos que implicassem a execução de cálculos aritméticos simples até à exaustão com as quatro operações fundamentais pois eram estes cálculos que eram necessários para a vida corrente do cidadão comum.

Como refere Carvalho e Silva (2022), “os algoritmos são tão antigos como o mundo (ou quase)” (p. 53), sendo que 2 mil anos antes de Cristo já os Babilónios usavam algoritmos, como, por exemplo, o algoritmo para a obtenção da raiz quadrada de um dado número natural.

Nos dias de hoje pretende-se que um aluno que saia do sistema de ensino tenha um maior desenvolvimento intelectual, que desenvolva mais competências relacionadas com o

cálculo mental, as quais devem permitir efetuar pequenas estimativas e desenvolver espírito crítico de forma a que um determinado aluno consiga interpretar um valor num determinado contexto.

O PC permite uma melhor compreensão dos números e das operações efetuadas entre estes. A programação possibilita que cada aluno observe a noção de números ou variáveis sob um determinado ponto de vista, dando-lhe um significado 'real'.

Os autores Brennan e Resnick (2012), a propósito da programação em *Scratch*, referem que cada aluno mobiliza conceitos, práticas e perspetivas quando programa usando esta linguagem de programação. As variáveis são utilizadas quando se trabalha, por exemplo, com conceitos como o de sequência, na criação de ciclos, ou de operadores ligados a condições, Desta forma cria-se um mundo de termos ligados diretamente à Matemática e com esta relacionados.

Por outro lado, o trabalho com a programação a este nível permite igualmente o desenvolvimento de práticas como a depuração, através da resolução de um determinado 'erro' de programação, ou ainda a necessidade de abstração quer por o utilizador ter de pensar e repensar num determinado problema para o ultrapassar, quer por ter de ligar diversas partes de um problema já resolvido para 'ligar' com o problema que está a tentar resolver.

Por último, a necessidade de colocar tudo em perspetiva, obriga o programador e utilizador a 'ver' o objeto que está a criar como algo de novo. O facto de programar estimula ainda um sentimento de pertença a uma determinada comunidade e que por si só pode proporcionar uma noção de partilha e por tal de inclusão,

Para Albuquerque (2022), existem diversas conexões entre a Matemática e as ciências da computação. Um dos conceitos evocados por este autor, e mais utilizados na computação, é o de algoritmo, que não é mais do que uma sequência estruturada de passos criados com um determinado fim. Este conceito tem raízes profundas na Matemática e teve na sua génese um árabe, de seu nome al-Khwarizmi, considerado o pai da Álgebra, sendo esta parte da Matemática vista aqui como um sistema de numeração posicional decimal com as suas regras e operações aritméticas.

Segundo Domingos (2017),

> Vivemos na era dos algoritmos. (…) Hoje, os algoritmos encontram-se em todos os recantos da civilização. Estão inseridos no tecido da vida quotidiana. Não apenas no nosso telemóvel ou computador portátil, mas também em nossa casa, nos nossos eletrodomésticos e brinquedos. (…) Os algoritmos agendam voos e depois pilotam os aviões. (…) Se,

> de repente, todos os algoritmos deixassem de trabalhar, seria o fim do mundo como o conhecemos. (p. 25)

Carvalho e Silva (2022a), numa conferência realizada na Escola Secundária António Damásio, em Lisboa, refere dois exemplos de algoritmos que funcionaram mal, nomeadamente o utilizado no software MACS que incorpora o Boeing 737 MAX e que em virtude de ter defeito fez com que estes tipos de aeronaves deixassem de voar, e o usado no voo do Ariane 5, a 4 de junho de 1996, que resultou na destruição do foguetão, devido a problemas no seu sistema de orientação (figura 1).

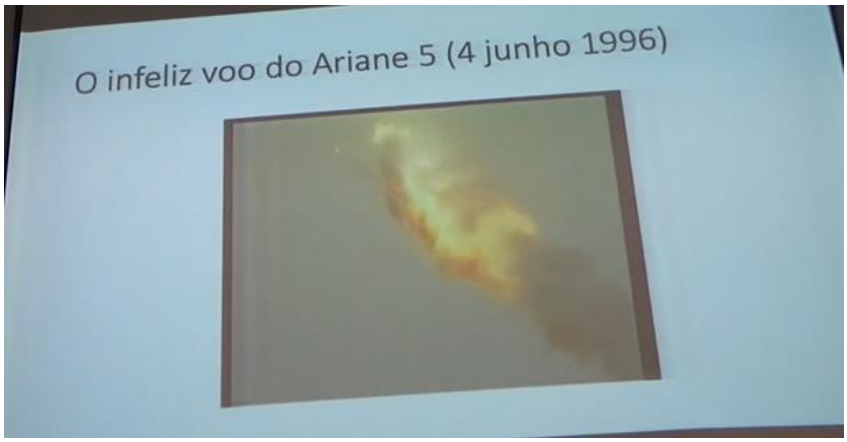


Figura 1 – Acidente no voo do Ariane 5, a 4 de junho de 1996, provocado por um algoritmo mal concebido.

Albuquerque (2022), por seu turno referencia Alan Turing (1912-1954) como sendo o criador de uma definição de números computáveis - números reais cuja representação decimal é possível obter recorrendo a meios finitos resultantes da aplicação de um algoritmo, dando inclusive o exemplo de um número definível. Turing é igualmente conhecido por ter desempenhado um papel essencial na descodificação de mensagens intercetadas entre os alemães na 2.ª guerra mundial permitindo às forças aliadas sair da guerra como vencedores.

As tabelas de Copérnico e de Pedro Nunes, segundo Albuquerque (2022), são bons exemplos de criações em que se trabalha com dados em larga escala, de forma a interpretar fenómenos astronómicos. Este este autor refere os matemáticos Newton, Gauss e Sebastião e Silva, como tendo participado no desenvolvimento de métodos iterativos que permitem, respetivamente, obter soluções aproximadas para zeros de funções polinomiais, ou soluções para um determinado sistema de equações lineares, ao que hoje designamos por método de eliminação de Gauss, que historicamente serviu para resolver problemas de mínimos quadrados relacionados com geodesia, ou ainda a “convergência de métodos numéricos” (Albuquerque, 2022, p. 33) como forma de resolver equações algébricas para uma dada equação. Tudo

situações que têm na base processos de abstração e que compõem aquilo a que hoje se dá o nome de pensamento computacional.

Segundo Carvalho e Silva (2022a), a introdução sistemática dos algoritmos e do pensamento computacional, com a ajuda intensiva da tecnologia, pode ajudar a recentrar o ensino da matemática no local de onde nunca deveria ter saído, nomeadamente a resolução de problemas reais concretos, usando os esquemas lógicos da Matemática.

A primeira linguagem de programação, criada com o objetivo de ser utilizada em sintonia com a Matemática, de seu nome Logo, remonta a 1967. Foi desenvolvida no Massachusetts Institute of Technology (MIT), por uma equipa liderada pelo matemático Seymour Papert, e da qual faziam igualmente parte Cynthia Solomon e Wally Feurzeige.

O Logo foi criado tendo por base uma filosofia construtivista, alicerçada em teorias de desenvolvimento cognitivo, ou não tivesse Papert trabalhado com Jean Piaget (1896-1980). Papert (1980), argumenta que quando uma criança aprende a programar o "processo de aprendizagem transforma-se" (p. 21). As aprendizagens dos alunos, ainda segundo Papert, são mais ativas, pessoais e direcionadas.

Na linguagem Logo, a generalidade dos comandos utilizados permitem desenhar e pintar e o resultado é mostrado imediatamente após a execução do comando, pelo que o aluno aprende por tentativa e erro. Quando um aluno tem um determinado raciocínio não adequado na utilização desta linguagem este transparece no écran, permitindo assim que os erros sejam corrigidos de forma quase imediata.

Um exemplo de programa com instruções que permitem construir um triângulo equilátero encontra-se na figura 2.

```
to triangulo
repeat 3 [forward 100 right 120]
end
```

Figura 2 – Programa em linguagem Logo que permite desenhar um triângulo equilátero.

O resultado da execução do programa descrito na figura 2 encontra-se na figura 3, em que o objetivo é desenhar um triângulo equilátero.

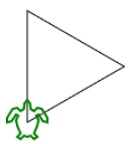


Figura 3 – Tartaruga da linguagem Logo a desenhar um triângulo equilátero.

Apesar da expressão PC ser atribuída a Wing, esta designação já tinha aparecido na obra de Papert (1980), sem que tivesse, contudo, uma definição explícita. Igualmente termos semelhantes aparecem na obra de Papert, nomeadamente, "pensamento procedimental"[1] (1980, p. 155), ou "pensamento combinatório"[2] (1980, p. 225).

A ideia apresentada por Papert de "PC para todos" (Resnick, 2008) encontrava-se à data da sua apresentação um pouco fora do seu tempo devido a dois fatores:

1) Dificuldades existentes na aprendizagem de linguagens de programação.
2) Inexistência de tarefas promotoras da aprendizagem do PC nas crianças, não refletindo assim os interesses dos alunos.

Com o passar dos anos, aliado a uma certa falta de objetividade na utilização da linguagem em contextos educativos e à existência de falhas na formação de professores para a sua utilização, a linguagem Logo foi desaparecendo das escolas e dos mecanismos de formação inicial de professores.

Nos dias que correm e atendendo ao facto de os computadores terem uma utilização mais generalizada, estes começaram a ser encarados de novo como essenciais para o ensino e aprendizagem da Matemática, existindo inclusive na Internet diversos programas gratuitos que permitem uma fácil utilização de linguagens de programação como o *Python*, quer para executar programas pré-concebidos, quer mesmo para a criação e desenvolvimento de programas em computadores pessoais.

2.1.2 Raciocínio matemático e possíveis desenvolvimentos em aula.

O raciocínio matemático, encarado como uma das variações do pensamento matemático, e o PC estão de tal forma interligados que os Princípios da Ciência para a Próxima Geração[3] (NGSS, 2013), referem como ações para desenvolver o raciocínio matemático nos alunos, ao nível do ensino básico, as seguintes:

- O uso das ferramentas digitais (por exemplo, computadores) para analisar bases de dados no reconhecimento de padrões e identificação de regularidades.
- O uso de representações para descrever e obter conclusões matemáticas.
- A criação de algoritmos para resolver um determinado problema.
- A aplicação de conceitos e processos matemáticos (por exemplo, taxas, rácios, percentagens, operações básicas, álgebra simples) para obter respostas para problemas e questões relacionadas com aspetos científicos e de engenharia.

[1] "Procedural thinking", no original.
[2] "Combinatorial thinking", no original.
[3] Next Generation Science Standards, no original.

- O uso de ferramentas digitais, de conceitos e argumentos matemáticos de forma a testar e comparar soluções propostas para um determinado problema de engenharia. (NGSS, 2013)

Na realidade é usual confundir PC com raciocínio matemático tal é a sua interligação e pontos de contacto. O PC assenta muito na utilização da tecnologia de forma a ajudar na obtenção e visualização de dados e na análise de problemas. O raciocínio matemático por sua vez centra-se mais nas competências de modo a conseguir-se resolver determinado tipo de tarefas, normalmente relacionadas com Álgebra e Geometria (Gilchrist et al., 2021).

Albuquerque (2022), estabelece uma relação direta entre a criação de algoritmos e o pensamento lógico referindo que: “Os algoritmos incluem naturalmente testes lógicos e uma evolução diferenciada em função da avaliação de condições lógicas, pelo que é essencial o uso de cálculo lógico” (p. 35).

Por outro lado, considera que “Na modelação matemática está presente a abstração (…)” (Albuquerque, 2022, p. 35), quando se negligenciam os aspetos menos essenciais para a resolução de um determinado problema, destacando como exemplos a descrição dos astros para Copérnico, em que são enfatizados aspetos numéricos em detrimento de aspetos físicos, ou o problema das pontes de Konigsberg (figura 4), em que Euler reduz a questão à teoria dos grafos.

Figura 4 – Pontes de Konigsberg.

Albuquerque (2022) menciona ainda Descartes e o seu Discurso do Método, sendo que para René Descartes (1596-1650) um dos processos que permite resolver um problema complexo passa por dividir esse problema em partes, juntando-se posteriormente os diversos constituintes, depois de se garantir a integridade de cada parte e a ligação coerente entre estas.

Estabelece igualmente uma comparação direta deste processo com a estrutura axiomática da Matemática em que, para demonstrar um determinado resultado, se parte de um conjunto de axiomas, efetuando a analogia com a criação de funções na programação (def() em

linguagem *Python*, por exemplo), com um determinado padrão ou regularidade, como forma de decompor uma determinada rotina em rotinas mais simples.

Por último, referindo-se à depuração[4], Albuquerque (2022), considera que obter muitos erros na resolução de um determinado problema, quer estes sejam devido a uma sintaxe deficiente, ou por erros de semântica, pode ser inicialmente desmotivador, no entanto, tal caraterística de utilização das máquinas é a "que as torna mais úteis por serem muito fiáveis" (p. 36). Sugerindo como solução para a resolução desses erros uma maior persistência por parte do utilizador, ou o recurso a alguém mais experiente, ou mesmo recorrendo a um motor de busca na Internet.

Efetivamente PC e raciocínio matemático, embora se possam confundir, não são a mesma coisa. Ponte, Quaresma e Pereira (2020), consideram que *raciocinar* tem um significado mais restrito que *pensar* (figura 5). Para Ponte et al. (2020), "raciocinar é realizar inferências de forma fundamentada, ou seja, partir de informação dada para obter nova informação através de um processo justificado" (p. 7).

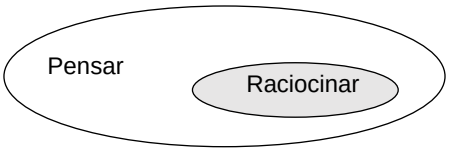


Figura 5 – Raciocinar e pensar (Ponte et al., 2020, p. 7)

Para estes autores, a descrição de um objeto, o relato de um acontecimento, a expressão de um sentimento, ou a formulação de um desejo, envolvem pensamento, mas podem não envolver qualquer tipo de raciocínio.

Ponte et al. (2020), destacam como fazendo parte da Matemática três tipos de raciocínio, a saber: dedutivo, indutivo e abdutivo. Assim,

- O raciocínio dedutivo resulta da aplicação de um conjunto de regras de inferência a axiomas ou postulados, permitindo assim a obtenção de novas afirmações verdadeiras apelidadas de teoremas (Ponte et al, 2020);
- O raciocínio indutivo expressa-se pela inferência de uma regra através da observação de regularidades ou padrões em casos particulares, tendo sido um raciocínio muito utilizado por George Pólya (1887-1985);
- O raciocínio abdutivo é um processo de inferência em que se parte de um facto invulgar ou insólito e que permite a criação de uma conjetura.

[4] Do termo inglês "debbuging"

Para Ponte et al. (2020), “O facto insólito ou invulgar para o qual se formula uma conjetura (raciocínio abdutivo), pode levar à identificação de casos particulares que permitem a observação de uma regra geral (raciocínio indutivo)” (p. 7). Estes dois raciocínios em conjunto permitem a criação de um novo conhecimento, sendo que o raciocínio dedutivo possibilita a validação desse conhecimento.

Numa aula é importante promover nos alunos estes três tipos de raciocínio, sendo que as tarefas a desenvolver devem permitir o estabelecimento de momentos em que coexistam o trabalho autónomo com a discussão coletiva e nesse sentido o trabalho em três fases torna-se relevante, sendo estas fases as seguintes:

1) Lançamento da tarefa, com uma pequena discussão para promover o envolvimento dos alunos;
2) Trabalho autónomo dos alunos, realizado em pares, em grupos ou em modo individual;
3) Discussão coletiva, com apresentação e confronto de resoluções e síntese final. (Ponte et al., 2020, p. 8)

Tal como Ponte et al. (2020) referem, a utilização destas três fases permite colocar em prática uma abordagem de ensino exploratório, em que as tarefas dadas aos alunos não têm uma resolução imediata. Para que os alunos consigam executar em pleno as tarefas propostas têm que utilizar estratégias próprias utilizando conhecimentos prévios.

A resolução de problemas em aberto, sob a forma de tarefas, permite incentivar os alunos na utilização dos três tipos de raciocínio considerados, pois possibilita que estes assumam um papel relevante na interpretação das questões, na representação da informação fornecida e na conceção e concretização de estratégias de resolução (Ponte et al, 2020).

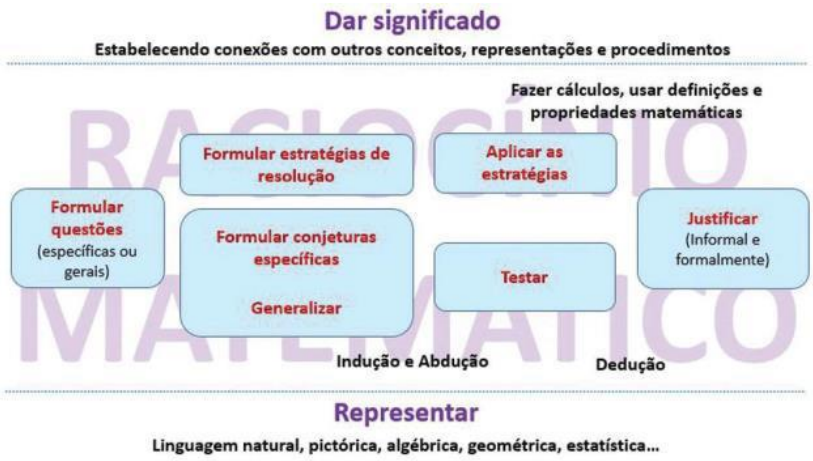


Figura 6 – Modelo do raciocínio matemático (Ponte et al., 2020, p. 10)

Estes autores apresentam um modelo teórico (figura 6) em que enquadram o raciocínio matemático em dois processos, a saber: o de representar e o de dar significado, sendo que o raciocínio matemático não pode existir se não houver representações dos objetos matemáticos, de preferência várias, e se não for atribuído qualquer tipo de significado a esses objetos e às ações desenvolvidas.

O modelo apresentado na figura 6 tem por base um processo de realização de uma investigação ou a resolução de um problema matemático, em que começando pela formulação de questões se passa posteriormente para o estabelecimento de conjeturas e para o desenvolvimento de estratégias de resolução (que podem passar por generalização), aplicando em seguida as estratégias e testando, as quais podem ou não confirmar as conjeturas, concluindo-se com o processo de validação (por justificação).

Tal como é referido por Ponte et al. (2020),

> A abordagem exploratória permite enfatizar a construção de conceitos, a modelação de situações e também a utilização de definições e propriedades de objetos matemáticos para os alunos realizarem raciocínios – nomeadamente para conjeturar, generalizar e justificar. Deste modo, presta atenção aos aspetos computacionais da Matemática, mas valoriza também os aspetos concetuais – ou seja, considera importante obter resultados, mas mais importante ainda compreender a estratégia que foi usada e a sua justificação. (pp. 10-11)

Para que exista um empenhamento profundo e concreto dos alunos tem que se atender a dois fatores muito importantes, como sejam: a escolha criteriosa das tarefas a serem executadas pelos alunos e a criação de um ambiente estimulante, em particular ao nível da comunicação, quer entre alunos, quer entre alunos e professor (Christiansen & al., 1986; Martins, 2016; Martins & al., 2018).

Esta metodologia de trabalho em aula permite que os alunos desenvolvam processos de raciocínio matemático e que os liguem com os seus conhecimentos e capacidades, possibilitando igualmente que os discentes reflitam sobre as suas práticas, aumentando assim os seus níveis de autonomia, de autoestima e de sentido crítico.

2.2 Utilização de tecnologia no ensino

O uso de tecnologia no ensino tem registado bastantes alterações desde meados do século XX, não só pelo aumento da capacidade do *hardware* nos computadores, mas também devido a uma intensificação dos programas/plataformas vocacionados exclusivamente para o ensino, como é o caso do *Moodle* ou do *Microsoft Teams*, ou de aplicações matemáticas como

o *Wolfram Alpha*, a que não tem sido alheio o incremento na Internet de motores de busca bastante poderosos como seja, por exemplo, o *Google*.

A utilização dos computadores e seus aplicativos em aula ainda hoje não é totalmente pacífica, quer por os professores não se sentirem totalmente confiantes para os utilizar, quer por as escolas enfrentarem problemas de envelhecimento dos equipamentos informáticos e de falta de Internet para se conseguir efetuar um trabalho convincente e satisfatório.

Oficialmente, a utilização de tecnologia nos programas de Matemática está prevista desde os programas de 1991, sendo que o Nacional Council of Teachers of Mathematics (NCTM), nos seus *Princípios para a Ação*, define o uso da tecnologia como um elemento essencial para a Matemática escolar, referindo que:

> A tecnologia inclui quadros interativos e um largo espectro de objetos portáteis, *tablets*, computadores e aparelhos concebidos com aplicações informáticas correntes que podem ser utilizadas para ajudar os alunos a darem sentido à matemática, a raciocinarem e a comunicarem matematicamente. (NCTM, 2017, p. 79)

A utilização de toda a tecnologia que se pode aplicar em aula pressupõe um desenvolvimento profissional sustentável, de forma que os professores se possam sentir confortáveis e livres de experimentar diferentes conceções e metodologias em aula com os seus alunos.

2.2.1 Tecnologia e Educação Matemática

Com a introdução da tecnologia em aula o professor "muda o seu papel de expositor e fiscalizador de exercícios para o de organizador de tarefas, conselheiro, recurso de informação, gestor, explicador e colega mais velho, enquanto que os alunos se envolvem em actividades exploratórias de aprendizagem (…)." (Fey, 1991, p. 46)

Este entendimento da utilização da tecnologia no ensino é reforçado por Papert (1980), quando refere que ser matemático, tal como ser um poeta, um compositor ou um engenheiro, significa muito mais do que simplesmente conhecer ou compreender.

Segundo Fey (1991), existem duas grandes linhas de transformação da Matemática escolar com a introdução da tecnologia na Educação Matemática, a saber: 1) Objetivos de Conteúdos/Processos, quer pela diminuição da atenção dada a determinados aspetos do trabalho efetuado em Matemática que pode ser substituído pelas máquinas, quer pela ampliação do currículo atual a ideias matemáticas e a aplicações de complexidade maior do que as que se encontram acessíveis aos alunos usando métodos tradicionais; 2) Estilos de ensino e aprendizagem sendo que os objetos, as relações e as operações matemáticas podem ser

representados eletronicamente sob formas que permitem uma maior manipulação exploratória controlada pelo professor.

Ponte, Nunes e Veloso (1991), apresentam um conjunto de 16 estudos de caso, que incluem experiências de cariz exploratório com computadores e diversos aplicativos informáticos, alguns feitos em Portugal, com professores e investigadores portugueses, outros elaborados noutros países da União Europeia, nomeadamente, Holanda, Reino Unido, Dinamarca, Bélgica, Itália, França, Alemanha e Espanha, pois o estudo inseria-se num estudo internacional centrado na Comissão das Comunidades Europeias (CEE).

Este estudo pretendia à altura relatar possíveis utilizações da informática, quer ao nível da sala de aula, quer em laboratórios de Matemática. Assim, são apresentados estudos de caso divididos em 4 grandes grupos. No primeiro grupo de estudos de caso encontramos várias experiências centradas na utilização do Logo e sob o tema da Geometria, com alunos do ensino básico e do ensino secundário; e, no uso de jogos educativos em que o tema eram vetores e translações.

Num segundo grupo de estudos de caso o trabalho esteve centrado na estatística e probabilidades, nomeadamente a utilização do computador na simulação de acontecimentos aleatórios; na utilização do programa Trinca-espinhas, em que foram trabalhados os conceitos de número, divisor e primo, no âmbito de um projeto que teve uma grande divulgação em Portugal, o projeto de desenvolvimento curricular MAT_{789}; e, no trabalho com bases de dados em que o objetivo era o manuseamento de dados e a sua análise em termos estatísticos.

Um terceiro grupo de estudos de caso analisados teve por foco o trabalho com funções e sua representação gráfica, em que foram usados um programa tutorial sobre funções trigonométricas; num outro estudo foram trabalhados conceitos relacionados com o cálculo infinitesimal, usando um programa específico criado para o estudo de funções; outra experiência centrou-se no trabalho com sucessões, noção de limite, funções algébricas e trigonométricas e derivadas, com alunos do 11.º ano, usando essencialmente uma folha de cálculo e fichas de trabalho; e, um outro estudo de caso na utilização de calculadoras gráficas pretendendo-se explorar a sua utilização na programação e na identificação de caraterísticas gráficas de funções.

Um quarto e último grupo de estudos de caso pretendeu explorar situações ligadas a novos temas curriculares e formas de abordagem, sendo que neste grupo foram analisados dois casos com atividades de modelação e outro que combinou conceitos matemáticos e ciências da computação.

Os autores referem-se igualmente às dinâmicas de trabalho utilizadas pelos alunos em sala de aula ou nos laboratórios de Matemática, sendo que na generalidade dos estudos de caso analisados os alunos trabalharam em grupo, de dois ou três elementos, com autonomia variável.

Ponte. Nunes e Veloso (1991) reforçam que:

> Em todos os estudos, vemos o computador trazer uma contribuição positiva ao processo de ensino/aprendizagem. O professor adquire novas funções e modifica a sua relação com os alunos. Estes, por seu lado, tendem a ficar muito envolvidos, a desenvolver novas capacidades e atitudes, a comprometer-se em trabalho de cooperação e refletir mais. (pp. 12 e 13)

O uso da tecnologia não elimina a necessidade de o aluno aprender algoritmos. Tal como refere NCTM (1991), é igualmente importante que este tenha alguma proficiência nos algoritmos de cálculo com papel e lápis, mas tal conhecimento deve partir de situações problemáticas em que o aluno esteja consciente que existem vários métodos possíveis que pode escolher, adotando o processo que se afigure mais adequado à resolução do problema. Isto é, por vezes pode ser mais relevante optar por uma estimativa, quando a resposta se pretende aproximada, ou cálculo mental, ou papel e lápis, ou o uso da calculadora, ou ainda o uso do computador.

Na figura 7 encontra-se um esquema, adaptado de NCTM (1991, p. 10), sobre decisões acerca dos procedimentos de cálculo em problemas numéricos.

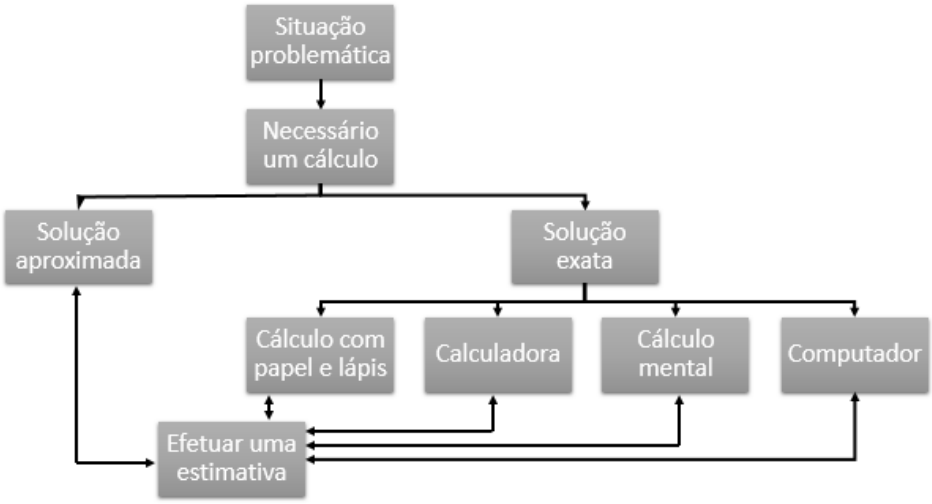


Figura 7 – Decisões sobre os procedimentos de cálculo a adotar em problemas numéricos (adaptado de NCTM ,1991, p. 10).

Este esquema pretende sintetizar quais as possíveis ações a desenvolver por um aluno quando este pretende resolver uma situação problemática e tem que efetuar um determinado cálculo, o qual pode ter uma solução exata ou aproximada.

Ao aluno exige-se que seja um *coletor* de conhecimento matemático e que, dentro do seu nível etário a das suas competências, seja capaz de efetuar uma escolha do processo e emitir uma resposta em conformidade com o problema sugerido.

Quando se comparam os estudos de caso relatados por Ponte et al. (1991), com as experiências efetuadas nos dias mais recentes verificamos que pouco ou nada tem mudado nas salas de aula e que ainda se continuam a utilizar os mesmos processos de ensino e aprendizagem que eram utilizados há mais de 30 anos.

O NCTM (2017), reforça que computadores, *tablets*, *smartphones* e calculadoras gráficas colocam ao dispor dos alunos, todos eles, uma variedade de aplicações que ajudam a explorar a Matemática, a tornar mais significativos determinados conceitos e procedimentos e a envolver os alunos em raciocínios matemáticos.

Destaca, em particular, o papel das aplicações gráficas como permitindo a visualização de gráficos, tabelas e expressões simbólicas; das folhas de cálculo que possibilitam, de forma rápida, o acesso a cálculos repetidos e a gerar tabelas, 'forçando' os alunos a estabelecer relações matemáticas; dos sistemas algébricos de computação (*Computer Algebra Systems - CAS*) que desencadeiam o trabalho com expressões algébricas; das aplicações de geometria dinâmica que permitem a exploração de conjeturas geométricas; das ferramentas de modelação muito utilizadas para explorar objetos tridimensionais; das aplicações de análise de dados, algumas bastante intuitivas, e que podem ser aplicadas a alunos mais novos até alunos no ensino secundário.

Assim, e tendo em vista a utilização da tecnologia com os alunos, o NCTM (2017) recomenda, o seguinte princípio, que apelida de Ferramentas e Tecnologia:

> - Implementar lições onde se utilize tecnologia em investigações, que procedam ou acompanhem o desenvolvimento de capacidades de papel e lápis;
> - Garantir que os alunos veem o poder mas também as limitações da tecnologia e contar que eles examinem as respostas, segundo a sua razoabilidade e aplicabilidade aos contextos, e que sabem escolher as ferramentas adequadas para a tarefa em questão;
> - Incorporar ferramentas matemáticas e tecnologia como componente diária das aulas de Matemática, reconhecendo que os alunos devem experimentar "tecnologias de ação matemática" e material na sala de aula para assegurar que a compreensão e o raciocínio dos alunos são reforçados. (p. 118)

Tal como podemos observar a tecnologia está bem patente no ensino e, em particular, no ensino da Matemática escolar. Existem bastantes exemplos que demonstram uma boa utilização da tecnologia, quer essas tecnologias sejam de ação matemática em que são produzidas respostas automáticas tendo por base os dados introduzidos pelo utilizador, permitindo o testar de conjeturas sobre relações matemáticas, quer sejam tecnologias não matemáticas, como sejam, por exemplo, um processador de texto, um software de apresentação ou aplicações de comunicação, que permitem que os alunos interajam matematicamente.

2.2.2 A tecnologia nos programas de Matemática do Ensino Secundário.

Em Portugal, o programa de Matemática A do Ensino Secundário (Bivar et al., 2014), que teve o seu início de implementação em 2015, no 10.º Ano, e términus em 2021, refere:

> Os professores e os alunos têm ao seu dispor, por exemplo, um vasto conjunto de recursos que facilitam o cálculo, as representações geométricas e a representação gráfica de funções, mas importa que os alunos adquiram capacidade crítica para reconhecer as situações em que a tecnologia não permite só por si justificar a adequação dos resultados encontrados ao problema proposto ou ilustrar devidamente os conceitos e procedimentos matemáticos envolvidos. (p. 28)

Neste mesmo programa, pode ler-se que:

> A utilização da tecnologia não pode (…) substituir a compreensão conceptual, a proficiência no cálculo e a capacidade de resolver problemas. Assim, os alunos devem dominar procedimentos como operar com polinómios, efetuar representações de gráficos de funções, resolver equações, calcular limites e derivadas sem necessitarem de utilizar recursos tecnológicos (calculadoras, computadores, etc.) que substituam algumas das capacidades matemáticas inerentes a esses procedimentos. Apenas a memorização e a compreensão cumulativa de conceitos, técnicas e relações matemáticas permitem alcançar conhecimentos progressivamente mais complexos e resolver problemas progressivamente mais exigentes. (Bivar et al., 2014, p. 28)

Como se verifica, o uso da tecnologia foi objeto de recomendação no programa de 2014, no entanto, e ao contrário do que é afirmado em NCTM (2017), tal utilização não pode servir, por exemplo, para atividades de natureza exploratória, pois neste programa o estudo dos conceitos deverá preceder a sua aplicação pelos alunos.

Este princípio encontra-se em total desacordo com o que foi afirmado no programa de Matemática A para o 10.º Ano, da autoria de Carvalho e Silva, Fonseca, Martins, Fonseca e Lopes (2001), em que explicitamente se aconselha:

- O uso de calculadoras gráficas, por permitirem desenvolver a capacidade de lidar com elementos de que apenas uma parte se pode determinar de forma exata; desenvolvendo assim a capacidade de resolver problemas de aplicações da matemática e a capacidade de analisar modelos matemáticos;
- O uso do computador, pelas suas potencialidades, nomeadamente nos domínios da geometria dinâmica, da representação gráfica de funções e da simulação, permite trabalhar com atividades de exploração, pesquisa, recuperação e desenvolvimento de conteúdos;
- O uso da Internet, como forma de criação de uma boa imagem da matemática, através da consulta de páginas que possibilitem a participação em projetos internacionais ou em acontecimentos de diversa índole relacionados com matemática.

Em Carvalho e Silva et al. (2001) é referido igualmente, numa coluna de indicações metodológicas, que:

> É preciso ter presente que a "tecnologia" em si não está em causa como conteúdo de ensino, mas que são as aprendizagens que ela pode proporcionar que justificam o seu uso. O recurso à tecnologia pode auxiliar os estudantes na compreensão de conceitos matemáticos e prepará-los para usar a matemática num mundo cada vez mais tecnológico.(…) Um estudante deveria registar por escrito, com os comentários julgados adequados, as observações que fizer ao usar a calculadora gráfica, o computador ou outro material, descrevendo com cuidado as propriedades constatadas e justificando devidamente as suas conclusões relativamente aos resultados esperados (desenvolvendo-se assim tanto o espírito crítico como a capacidade de comunicação matemática). (p. 22)

Neste programa de 2001, além de orientações genéricas relativas à utilização da tecnologia, são igualmente referidas indicações metodológicas para o uso dessa mesma tecnologia, nomeadamente o professor deverá encaminhar o aluno no sentido que este estabeleça algumas conclusões por escrito relacionadas com o que observa quando investiga determinadas propriedades.

Já nas Aprendizagens Essenciais de Matemática A para o Ensino Secundário (ME, 2018a; 2018b; 2018c), que se encontram atualmente em vigor, pode ler-se:

> Desde o início do ensino secundário, a tecnologia deve ser usada de forma crítica e inteligente contribuindo para o desenvolvimento de novas competências associadas à área da programação que, nalguns países, estão já integradas nos programas de Matemática.
> A tecnologia é uma ferramenta cada vez mais presente na sociedade e no mercado de trabalho e também um recurso essencial no ensino, ajudando os alunos a perceber as ideias matemáticas, a raciocinar, a resolver problemas e a comunicar. Assim, a tecnologia gráfica deve estar presente, quer em contexto de sala de aula, quer em contexto de avaliação externa (p. 3).

Referindo igualmente na coluna relativa às práticas essenciais da aprendizagem em ME (2018a, 2018b, 2018c) que:

- Utilizar a tecnologia para fazer verificações e resolver problemas numericamente, mas também para fazer investigações, descobertas, sustentar ou refutar conjeturas.
- Utilizar a tecnologia gráfica, geometria dinâmica e folhas de cálculo, no estudo de funções, de geometria e números complexos. (p. 5-6)

E ainda em ME (2018b):

> A Estatística deve ser trabalhada de forma não formal, usando tecnologia (calculadora, folha de cálculo) partindo de pequenos projetos, com dados reais e de forma a permitir a compreensão do processo estatístico e a avaliação crítica e conhecedora das múltiplas informações estatísticas com que os alunos são confrontados no dia a dia. (p. 7)

Como se pode observar, estas indicações embora sendo algo escassas estão perfeitamente em linha com o que é preconizado em NCTM (2017), incentivando, de forma indireta, os professores a praticarem um ensino baseado em tarefas de natureza exploratória.

Carvalho e Silva et al. (2020), baseados no estudo *Trends in International Mathematics and Science Study – Advanced 2015* (TIMSS *Advanced*), conduzido pela International *Association for the Evaluation of Educational Achievement* (IEA) e do qual foi produzido um relatório[5] por Mullis, Martin, Foy e Hooper, publicado em 2016, referem que existe um trabalho continuado acerca do papel da tecnologia na educação e, particularmente, nas aulas de Matemática, verifica-se que a disponibilidade de acesso à tecnologia (computadores, tablets, calculadoras e telemóveis) se cifra em Portugal nos 78%, apresentando um valor idêntico à média dos países que fizeram parte deste estudo, num total de nove países.

Destacam igualmente não existirem diferenças significativas entre os resultados nos testes do TIMSS *Advanced* dos alunos portugueses com acesso à tecnologia nas aulas de Matemática A e aqueles que não têm acesso a tecnologia (485 versus 477 pontos).

Os autores Carvalho e Silva et al. (2020) referem igualmente, tendo por base o mesmo estudo (Mullis et al. de 2016) que a tecnologia é usada pelos professores para:

i) Fazer representações de gráficos de funções (média internacional 68%, Portugal 77%);
ii) Resolver equações (média internacional 63%, Portugal 73%);
iii) Processar e analisar dados (média internacional 59%, Portugal o mesmo valor);
iv) Modelar e fazer simulações (média internacional 57%, Portugal 72%). (p. 267)

Num mundo em que desde os finais do século XX se constata que a tecnologia está em todo o lado: nas transações comerciais, nos transportes, nas diversas atividades sociais e de recreio, a educação não poderia ficar de fora desta evolução e permanentemente o ensino

[5] TIMSS *Advanced* 2015 International Results in Advanced Mathematics and Physics e que pode ser obtido de http://timssandpirls.bc.edu/timss2015/international-results/advanced/

confronta-se com novas ferramentas e novos conceitos que possibilitam a adoção de novas metodologias e processos para o ensino e aprendizagem dos nossos alunos.

2.3 Utilizando a linguagem *Python* na disciplina de Matemática no Ensino Secundário

Para Fey (1991), embora os computadores realizem operações numéricas, gráficas, simbólicas e lógicas, a caraterística que os torna verdadeiramente únicos é o facto de poderem ser programados.

Estamos perante uma situação de PC quando pegamos num problema e o formalizamos, isto é, quando o transformamos em algo que objetivamente conseguimos determinar utilizando um número finito e sequencial de passos criando aquilo a que usualmente se designa por algoritmo.

Embora se possam criar diferentes algoritmos para resolver um mesmo problema, uns mais complexos do que outros, a verdade é que a suposta perfeição vem com a prática que conduz necessariamente a um aumento da competência para desenvolver algoritmos.

Na maioria dos casos, quando se cria um determinado algoritmo, não se pensa numa linguagem de programação específica, sendo que, apesar de cada linguagem de programação ter as suas particularidades, é possível adaptar esse algoritmo à linguagem de programação que se vai utilizar de forma a resolver o problema.

Sendo o algoritmo um modelo do problema real que se pretende resolver, por vezes este modelo é muito abstrato devido à necessidade em eliminar fatores *distratores*, obtendo-se assim um modelo teórico que pode ser a solução para variadíssimos problemas com contextos diferentes.

Se é verdade que muitos académicos das ciências de computação consomem muito tempo a pensar em novas soluções para um problema complexo, a generalidade dos programadores profissionais, que programam diariamente, são muito práticos e quando pretendem resolver um dado problema já possuem previamente rotinas que resolvem partes do problema colocado, bastando para tal construir algumas sequências novas e encaixar de forma adequada as pré-existentes de forma a constituir um todo uniforme que na prática resulta como solução do problema proposto.

De qualquer das formas, quando se constrói um dado algoritmo, a ideia genérica é sempre a mesma, decompõe-se o problema inicial em várias tarefas mais pequenas que sabemos resolver, ou temos uma ideia de como resolver, até se conseguir resolver o problema por

completo. Tal como refere Mailund (2021), “esta é a essência do pensamento computacional.” (p. 6)

Quando se trabalha em programação é necessário implementar e desenvolver esses programas numa determinada linguagem de programação. Várias são as linguagens de programação que podem ser utilizadas em contexto de sala de aula, no entanto, talvez a mais adequada seja a linguagem de programação *Python*, pois além de estar disponível em *open source*, é das linguagens mais difundidas e utilizadas em todo o mundo, devido a ser uma linguagem concisa e simples, permitindo uma fácil leitura do seu código, e necessitar de poucas linhas de programação para executar uma tarefa quando comparada com outras linguagens.

O *Python* foi concebido pelo matemático Guido von Rossum, na Holanda, no final de 1989, devendo o seu nome aos humoristas ingleses *Monty Python*, criadores da série *Monty Python´s Flying Circus*, transmitida pela *British Broadcasting Corporation* (BBC) entre 1969 e 1974.

A linguagem *Python* pode ser utilizada em diferentes aplicativos/plataformas, nomeadamente em computadores e *tablets*, podendo igualmente ser instalada em diversos modelos de calculadoras gráficas e aplicações para *smartphones*.

É uma linguagem de programação que pode ser utilizada quer por programadores menos experientes, quer por especialistas, sendo usada essencialmente para processamento de texto, análise de dados e criação de páginas para a Internet. Possui igualmente uma grande quantidade de módulos ou bibliotecas, que podem ser facilmente carregados, e que são bastante úteis no trabalho com matemática, como sejam, por exemplo, bibliotecas para efetuar operações matemáticas específicas ou para efetuar gráficos e desenhos no écran.

Como refere Carvalho e Silva (2022), a propósito dos programas de Matemática em França, a linguagem Python foi a eleita “por ser concisa, grandemente utilizada e disponível numa variedade de plataformas.” (p. 53). Segundo este autor “os alunos devem passar da linguagem natural à linguagem Python e vice-versa” (p. 53).

Na figura 8 encontra-se o algoritmo e o correspondente programa em *Python* que permite determinar a espessura de uma folha de papel ao efetuar sucessivas dobragens.

Algoritmo:

```
função dob
  ler n
  e = 0.1
  Para i de 1 até n :
     e = 2e
  devolver e
```

Ficheiro dobragens.py:

```
def dob(n) :
    e = 0.1
    for i in range(1,n+1):
         e = 2 * e
    return(e)
```

Figura 8 – Algoritmo e programa para resolver o problema das dobragens de uma folha de papel.

O algoritmo/programa apresentados na figura 8 constitui um exemplo simples e eficaz do que pode ser a aplicação do PC na Matemática pois permite o reconhecimento da abstração, por permitir reduzir a complexidade da tarefa; da decomposição, ao dividir a tarefa em tarefas menores; do reconhecimento de padrões, ao reconhecer regularidades e relações; da algoritmia, ao permitir obter uma solução passo a passo para o problema; e da depuração, ao procurar corrigir eventuais erros que surjam na construção do programa, tudo caraterísticas deste tipo de pensamento (Espadeiro, 2021).

3

METODOLOGIA E DESENHO GERAL DA INVESTIGAÇÃO

Nesta secção descreve-se a metodologia utilizada na investigação, sendo esta essencialmente uma metodologia qualitativa com um cunho interpretativo, assente na recolha de dados de elementos produzidos pelos alunos, quer estes fossem expressos sob a forma verbal, quer na forma escrita, por meio de papel-e-lápis ou digitalmente.

É feita igualmente uma caraterização do que são estudos de caso em educação e, são descritas neste capítulo as técnicas utilizadas na recolha de dados, nomeadamente para a realização das entrevistas aos alunos, no decurso do trabalho de campo, e para a obtenção dos elementos documentais produzidos pelos alunos.

Por último, é apresentado o plano metodológico que permitiu levar à prática a investigação.

3.1 Investigação qualitativa

A metodologia adotada para o desenvolvimento desta investigação ação foi uma metodologia de natureza qualitativa em que o investigador foi um observador participante do trabalho de campo.

Para Bogdan e Biklen (1994), a expressão *investigação qualitativa* é "um termo genérico que agrupa diversas estratégias de investigação que partilham determinadas características" (p. 16), tendo surgido no final do século XIX e início do século XX, atingindo o seu ponto máximo nas décadas de 1960 e 1970.

Estes autores referem que os dados recolhidos são qualitativos, quando são ricos em pormenores descritivos dos quais constam pessoas, locais e conversas, sendo o seu tratamento estatístico algo complexo e de difícil execução. As questões que se pretendem investigar não dependem da operacionalização de variáveis, sendo que o principal objetivo de uma investigação deste tipo é a análise de todo o processo, na sua plenitude e complexidade,

atendendo ao contexto natural em que se desencadeiam os fenómenos. Compreende a observação sistemática dos processos, quer informais, como seja a entrevista, quer formais, como os questionários e os dados documentais.

Ainda segundo os mesmos autores, na investigação qualitativa em educação, o investigador comporta-se como um viajante que não planeia tudo de forma meticulosa, embora seja rigoroso nos seus processos de análise.

Ao contrário de uma investigação do tipo quantitativo, em que o investigador utiliza dados de natureza numérica que lhe permitem provar relações entre variáveis, a investigação qualitativa utiliza principalmente metodologias que lhe permitam criar dados descritivos de forma a que observe o modo de pensar dos participantes, não estando contemplada a resposta a questões prévias ou o teste de hipóteses de forma numérica.

Para Bogdan e Biklen (1994), uma investigação qualitativa comporta cinco caraterísticas, sendo que nem todas têm que transparecer de igual modo, é tudo uma questão de grau. A tabela 1 apresenta essas cinco caraterísticas.

Tabela 1 – Caraterísticas de uma investigação qualitativa (Bogdan e Biklen, 1994).

Caraterísticas	**Definição**
A fonte direta dos dados é o ambiente natural e o investigador constitui-se como o instrumento chave	O contexto em que os dados são recolhidos é relevante e por isso o investigador tende a recolher os dados no terreno. As formas de registo dos dados: gravações áudio ou vídeo, bloco de notas. lápis, etc, não é relevante. O que é importante é o cenário onde se desenrola a ação.
Descritiva	Os dados recolhidos tomam a forma de narrativas ou de imagens e raramente expressam quantidades. Os dados apresentados incluem extratos de entrevistas, notas de campo, imagens reais, vídeos, documentos pessoais e registos diversos que servem para suportar a apresentação dos dados. A palavra escrita é relevante e é utilizada para descrever os detalhes.
O processo é mais importante do que os resultados ou produtos	Foca-se na forma como se formam as definições, quer sejam as definições que o professor tem dos seus alunos, por exemplo, quer a forma como os alunos se vêm uns aos outros.
Os dados são analisados de forma indutiva	O processo é lento e o investigador toma consciência do processo quando consegue agrupar os dados recolhidos construindo abstrações. Parte da análise de situações abertas para obter conclusões específicas, como se agisse como um funil.
O significado tem uma importância vital	O investigador interessa-se sobre as diversas dinâmicas internas que estão em jogo e que são, por vezes, impercetíveis a um investigador exterior ao processo e que constituem perspetivas dos participantes.

Para Merriam (1988), nas investigações qualitativas os respetivos intervenientes não são reduzidos a variáveis isoladas, mas observados como parte integrante de um todo no seu contexto natural. É de reforçar que, reduzindo pessoas a dados estatísticos, existem algumas características do comportamento humano que são negligenciadas. Esta autora refere que para se conhecer melhor o pensamento do ser humano deverão ser utilizados dados descritivos, provenientes de registos e de anotações pessoais de comportamentos observados.

Bogdan e Taylor (1984) referem que nas investigações qualitativas o investigador deve estar envolvido no campo de ação dos investigados, sendo que, na sua essência, este método de investigação baseia-se principalmente em conversar, ouvir e deixar que os participantes se sintam livres para se exprimir. Para estes autores, numa investigação qualitativa, ao possibilitar que o investigador seja subjetivo na busca do conhecimento, permite que exista uma maior diversificação nos procedimentos metodológicos utilizados na investigação.

Tendo por objetivo analisar o desenvolvimento do PC em alunos do ensino secundário, nomeadamente caraterizar e identificar processos utilizados pelos alunos no trabalho de programação em *Python*, a utilização de uma metodologia de natureza qualitativa é a que se afigura mais adequada para responder às questões e objetivos fixados, tanto mais que o ambiente em que ocorreu a investigação foi no ambiente natural da escola.

A busca de significado foi uma das principais preocupações do investigador na análise e interpretação dos dados, tendo em vista a compreensão das ideias e as conceções dos participantes no estudo, baseado num processo eminentemente gradual e indutivo.

3.2 Estudos de caso

Para Yin (2012), um estudo de caso é uma investigação que se baseia fundamentalmente no trabalho de campo, quer estudando uma pessoa, um programa ou uma instituição no seu contexto, utilizando para tal, entrevistas semiestruturadas, observação, documentos, questionários e artefactos. O estudo de caso é muito utilizado quando não se consegue controlar os acontecimentos e, portanto, não é de todo possível manipular as causas do comportamento dos participantes (Yin, 2012).

Ludke e André (1986) referem que o interesse do estudo de caso incide naquilo que ele tem de único, mesmo que posteriormente fiquem evidentes certas semelhanças com outros casos ou situações. Estes autores sugerem que se deve escolher este tipo de estudo quando se pretende estudar algo singular, que tenha um valor intrínseco.

Para caracterizar o estudo de caso, Ludke e André (1986) referem sete caraterísticas numa investigação qualitativa:

1) Visam a descoberta, na medida em que podem surgir, em qualquer momento, novos elementos e aspetos importantes para a investigação, além dos pressupostos do enquadramento teórico inicial;
2) Enfatizam a interpretação em contexto, pois todo o estudo desta natureza tem que ter em conta as características da escola, o meio social em que está inserida, os recursos materiais e humanos, entre outros aspetos;
3) Retratam a realidade de forma completa e profunda;
4) Usam uma variedade de fontes de informação;
5) Permitem generalizações naturalistas;
6) Procuram representar as diferentes perspetivas presentes numa situação social;
7) Utilizam uma linguagem e uma forma mais acessível do que outros métodos de investigação.

Um estudo de caso deve utilizar-se quando se pretende observar e descrever detalhada e aprofundadamente um determinado fenómeno (Merriam, 1988). Para Bogdan e Biklen (1994) o estudo de caso pode ser representado como um funil em que o início do estudo é a parte mais larga. Estes autores referem igualmente que nos estudos de caso, a melhor técnica de recolha de dados consiste na observação participante sendo o foco de estudo uma organização particular.

Como principais vantagens deste tipo de investigação é o facto de se constituir como o método ideal para caraterizar um indivíduo e a circunstância de o investigador poder, a qualquer momento da investigação, alterar os métodos da recolha de dados e estruturar novas questões de investigação.

Num estudo de caso, o investigador, depois de recolher todos os dados de cariz qualitativo, tem poucos caminhos percetíveis no sentido de analisar os dados obtidos, portanto, torna-se fundamental conhecer o modo de pensar dos alunos e compreender o seu ponto de vista para tentar perceber o significado que os alunos atribuem às diferentes situações propostas pelo investigador.

Para Yin (2012) a qualidade de um estudo de caso está relacionada com critérios de validade e fiabilidade, a saber:

- A Validade de Construto que permite verificar até que ponto uma medida utilizada num estudo de caso é adequada aos conceitos a serem estudados;
-.A Validade Interna que avalia em que medida o investigador demonstrou a relação causal entre dois fenómenos observados;
- A Validade Externa que mostra até que ponto as conclusões de um estudo de caso podem ser generalizáveis a outras investigações de casos semelhantes.

A fiabilidade de um estudo de caso permite mostrar em que medida outros investigadores chegariam a resultados idênticos, utilizando iguais metodologias na mesma investigação.

No que diz respeito à generalização das conclusões e resultados de um estudo de caso, é fundamental perceber que este tipo de metodologia de investigação não tem por finalidade generalizar os resultados obtidos mas sim levar ao conhecimento profundo de casos concretos e particulares (Merriam, 1988 e Yin, 2012).

Tesch (1990), refere a existência três tipos de análise de dados de estudos de caso (tabela 2).

Tabela 2 – Tipos de análise de dados de estudos de caso (Tesch, 1990).

Tipos	Definição
Interpretativa	Todos os dados são analisados ao pormenor e são recolhidos com o objetivo de serem organizados e classificados em categorias de forma que se consiga explorar e explicar o fenómeno em estudo.
Estrutural	Analisam-se os dados com o objetivo de encontrar padrões ou regularidades que permitam explicar a situação em estudo.
Reflexiva	Permite interpretar ou avaliar o fenómeno em estudo, por norma, por julgamento ou intuição do investigador.

A presente investigação constitui-se como um estudo de caso interpretativo na medida em que decorreu no ambiente natural da sala de aula, onde, a cada momento, surgiram novos aspetos importantes para investigar. O investigador foi o principal agente de recolha de dados recorrendo à observação direta e tendo por base a interação entre e com os alunos, através de conversas informais. Os métodos de recolha de dados, essencialmente descritivos, foram evoluindo e pretenderam identificar quais as reações dos alunos durante a sua interação com o computador e com os guiões para a descoberta da linguagem *Python*.

3.3 Recolha de dados

Uma das caraterísticas dos estudos de caso é a possibilidade de se obter informação recorrendo a múltiplas fontes de dados. O estudo de caso faz recurso a uma diversidade de formas de recolha de informação, dependente da natureza do caso e tendo por finalidade, possibilitar o cruzamento de diversas visões do estudo ou de análise (Hamel, 1997).

Entre os instrumentos de recolha de informação encontram-se preferencialmente o questionário, as fontes documentais, a entrevista, o diário, e outros registos, como sejam o

recurso a *e-mails*, *WhatsApp*, plataformas eletrónicas como, por exemplo, o *Microsoft Teams*, etc.

O questionário, enquanto técnica de recolha de dados, presta um serviço relevante à investigação qualitativa. Esta técnica comporta a criação por norma de um formulário, previamente elaborado.

O recurso a fontes documentais é outra das estratégias usuais num estudo de caso. Podem suportar-se em relatórios, instrumentos escritos em papel-e-lápis e sob a forma digital, planificações, comunicados, dossiers, etc. A informação recolhida pelo investigador pode servir para dar consistência ao caso, acrescentar informação ou para validar evidências.

A entrevista é uma das fontes de informação mais importantes e essenciais nos estudos de caso (Yin, 2005). A entrevista é um instrumento que permite captar uma diversidade de descrições e interpretações que os elementos em estudo têm sobre a realidade. A entrevista permite uma interação verbal entre, no mínimo, duas pessoas: o entrevistado, que fornece respostas, e o entrevistador, que solicita informação para, tendo por base alguma sistematização e interpretação adequada, extrair conclusões sobre o estudo em causa.

As entrevistas semiestruturadas têm suscitado, segundo Flick (2004), bastante interesse e têm sido de utilização frequente. Este interesse está associado com a expectativa de que é mais provável que os sujeitos entrevistados expressem os seus pontos de vista numa situação de entrevista desenhada de forma relativamente aberta do que numa entrevista normalizada ou num questionário. Neste tipo de entrevista, o entrevistador estabelece os limites de inquirição sobre os quais incidem as perguntas.

A entrevista semiestruturada não segue uma ordem pré-estabelecida na formulação das questões, deixando maior flexibilidade para colocar essas perguntas no momento mais apropriado, conforme o encadeamento das respostas do entrevistado

O diário é um bom instrumento para registo dos processos e procedimentos de investigação como preconizam Bogdan e Biklen (1994) e atendendo à perenidade da memória de qualquer investigador é o local onde devem permanecem os dados, os sentimentos e as experiências da investigação (Vásquez & Angulo, 2003). O diário constitui-se deste modo um instrumento reflexivo e de análise, onde são registadas as notas do trabalho de campo e reflexões sobre o que o investigador vê e ouve. É assim um registo da observação direta, podendo igualmente haver grelhas de observação, onde os registos são feitos de forma mais sistematizada.

Os registos eletrónicos têm surgido igualmente como uma fonte essencial de dados para análise. A utilização destes registos, como fonte de informação, é algo relativamente recente

e que decorre da utilização da tecnologia. Por exemplo, a informação registada por plataformas eletrónicas é quase infindável e requer, normalmente por parte do investigador, a seleção da informação relevante para o caso em estudo. Entre os registos eletrónicos encontram-se os *e-mails*, as mensagens em redes sociais como, por exemplo, o *WhatsApp*, e os ficheiros e registos deixados em plataformas digitais como, por exemplo, o *Microsoft Teams*.

Nesta investigação a recolha de dados teve em conta cinco procedimentos fundamentais: os apontamentos efetuados pelos alunos em papel-e-lápis e digitalmente; os programas que foram efetuados por estes na execução dos guiões fornecidos; as entrevistas semiestruturadas no decurso do trabalho de campo; os trabalhos efetuados no final, nomeadamente os programas, os relatórios e os vídeos; e, as respostas dadas pelos alunos no questionário final. Foi igualmente realizado um diário, como preconizam Bogdan e Biklen (1994) e onde foram relatados aspetos diferenciados do projeto, dificuldades sentidas na sua aplicação e pequenas notas do trabalho de campo.

Os apontamentos elaborados pelos alunos decorreram da apresentação de guiões em aula, sendo que aos alunos do ensino secundário, do 12.º ano, do Curso de Ciências e Tecnologia, foi solicitado no final que concretizassem um projeto, de tema livre, tendo sido pedida em concreto a criação ou adaptação de programas em *Python*, sendo que para tal os alunos deveriam utilizar conceitos e processos aprendidos na disciplina de Matemática, preferencialmente do 12.º ano.

O trabalho proposto aos alunos implicou igualmente a criação de um relatório escrito em suporte digital e de um vídeo em que estes relataram qual era o objetivo do programa e as suas principais caraterísticas, como forma de desenvolver um sentido crítico do trabalho realizado e permitir igualmente a apresentação do trabalho efetuado por cada um dos grupos, aumentando desta forma os níveis de autoestima e de confiança neles próprios.

3.4 Plano metodológico

Antes de efetuar a recolha de dados no terreno, foi necessário obter previamente as devidas autorizações por escrito (Bogdan & Biklen, 1994), as quais foram solicitadas primeiramente à direção da escola onde decorreu o estudo, explicitando os objetivos, métodos de recolha de dados, duração da intervenção e potenciais benefícios. Foram igualmente informados o coordenador de departamento e a representante de grupo da intenção de conduzir esta investigação em aula. Por último, foram solicitadas as autorizações aos encarregados de educação para a participação dos respetivos educandos no estudo, garantindo em todos os momentos a confidencialidade das informações recolhidas (anexo I).

As técnicas que foram utilizadas para a recolha de dados consistiram na elaboração de documentos pelos alunos, em resultado da realização dos guiões em aula (anexo III); na análise direta da conversação entre os alunos, e entre alunos e professores; na observação participante da aplicação dessas tarefas aos alunos; na análise dos registos criados pelos alunos; e na análise dos questionários aplicados aos alunos. Foi criado igualmente um diário de bordo, com notas de campo e grelhas de observação (Bogdan & Biklen, 1994) que permitiu monitorizar todo o processo relativo à investigação.

Nos vídeos produzidos pelos alunos, que se constituíram desta forma como estudos de caso, foram analisados não só os programas elaborados, mas também o seu conteúdo matemático e as dificuldades e constrangimentos sentidos por estes na resolução da tarefa final.

No sentido de obter uma adequada análise dos dados tornou-se necessário que, ao longo do todo o processo de recolha, se encontrassem formas de categorização desses mesmos dados (Wiersma, 1995). Essas categorias foram estabelecidas tendo em atenção a procura de regularidades, através da análise de padrões de pensamento ou comportamento, palavras ou frases, que contivessem algo em comum. A análise de conteúdo foi efetuada com o estabelecimento de inferências que emergiram das regularidades encontradas (Bogdan & Biklen, 1994; Wiersma, 1995).

Tendo por base a revisão de literatura, foram desenvolvidos critérios de análise que foram tidos em conta ao longo de todo o estudo. Assim, e após efetuar a recolha de dados, através da pesquisa de documentação, foi construído um referencial que permitiu a análise de conteúdo.

Após a análise dos vídeos, relatórios, programas efetuados por cada grupo de dois alunos e questionários finais, houve o cuidado de verificar se havia coerência entre todo o trabalho desenvolvido pelos 24 alunos que efetuaram o trabalho final (4 alunos não apresentaram o programa final solicitado), tendo sido estes analisados de acordo com a revisão da literatura, de forma a se poder estabelecer conclusões no final.

Assim, e de forma a que aprendessem como programar em *Python*, linguagem desconhecida da generalidade destes alunos, foi-lhes solicitado pelo professor da disciplina de Aplicações Informáticas B que, em três aulas de 90 minutos da sua disciplina, analisassem 3 guiões de trabalho (anexo III).

Com a presença dos dois professores das duas disciplinas, Aplicações Informáticas B e Matemática A, nas três aulas de Aplicações Informáticas B, os alunos aprenderam os rudimentos de como programar em *Python*.

A turma, constituída por 28 elementos, já se encontrava dividida em 14 grupos de dois elementos e essas dinâmicas de sala de aula foram mantidas e acalentadas neste projeto. Foi igualmente criada na plataforma *Microsoft Teams* da escola uma *equipa* para o desenvolvimento desta investigação onde os alunos definiram uma diretoria para cada grupo de dois alunos e colocaram os trabalhos que foram efetuando no decurso das três aulas. O esclarecimento de quaisquer dúvidas que tiveram foi efetuado por mensagens na plataforma, por *e-mail* e/ou por *WhatsApp*.

Deu-se início ao trabalho de campo na aula que decorreu no dia 4 de março de 2022, solicitando-se aos alunos que respondessem a um questionário (anexo II) e que analisassem e efetuassem as propostas elencadas no guião 1 (Plataformas/Dados/Operadores/Funções – anexo III). Na aula de dia 11 de março foi analisado o guião 2 (Algoritmo/Função condicional – anexo III).. Por último, na aula de dia 18 de março, foi fornecido o guião 3 (Estruturas de repetição – Ciclos FOR e WHILE – anexo III), que foi trabalhado nessa aula.

No final desta última sessão, e no seguimento do guião 3 (anexo III), foi proposto a cada aluno que, em conjunto com o seu colega de grupo, criassem um programa em *Python*, tendo por base um tema matemático estudado no ensino secundário. Deveriam igualmente elaborar um relatório sobre o trabalho efetuado e um vídeo, com a duração máxima de 5 minutos, explicando o programa desenvolvido e mostrando exemplos de aplicação. Para a execução deste trabalho – elaboração do programa em *Python*, de um vídeo e do relatório – foi definida como data-limite de entrega o dia 31 de março, tendo sido prolongado posteriormente este prazo de entrega até dia 5 de abril.

Refira-se a propósito que os elementos solicitados aos alunos para o projeto foram objeto de avaliação nas duas disciplinas, estando previstos tais procedimentos nos critérios de avaliação dos grupos disciplinares respetivos, na componente de trabalho colaborativo.

Atendendo ao facto de haver dois grupos que não apresentaram o trabalho final proposto, e de forma a manter o anonimato dos alunos, foram constituídos 12 estudos de caso, com 11 alunos do género feminino e 13 do género masculino, sendo que os elementos de cada grupo foram designados, no decurso deste trabalho, pelas letras *F* e *H*, respetivamente, com índices, isto é, F_k ($k \in \{1, \dots, 11\}$) e H_i ($i \in \{1, \dots, 13\}$).

Assim, consideraram-se os seguintes pares para cada um dos estudos de caso: 1) F_1/F_2; 2) H_1/H_2; 3) F_3/F_4; 4) H_3/H_4; 5) H_5/H_6; 6) H_7/F_5; 7) F_6/F_7; 8) F_8/H_8; 9) H_9/H_{10}; 10) H_{11}/H_{12}; 11) F_9/F_{10}; e,12) F_{11}/H_{13}.

Posteriormente, já no 3.º período, no dia 31 de maio, foi distribuído um questionário aos alunos para que refletissem sobre a aplicação dos guiões em aula e sobre o produto final apresentado (anexo IV).

4

CONTEXTO EDUCATIVO, INTERVENIENTES E IMPLEMENTAÇÃO

Nesta secção descreve-se a escola onde decorreu a experiência, os diversos intervenientes envolvidos e a implementação no terreno das tarefas propostas. São igualmente descritos os elementos que fizeram parte dos estudos de caso.

4.1 A escola onde decorreu a investigação

O trabalho de campo decorreu no 2.º período de 2021/22, numa escola da área da grande Lisboa em que o seu patrono é uma personalidade das ciências e das artes.

É uma escola que tem uma arquitetura original, sendo simultaneamente luminosa e sóbria, funcional e confortável, em permanente diálogo com os jardins e o espaço envolvente. Foi requalificada em finais da década de 2010, pela Parque Escolar, EPE, que lhe restituiu o brilho original, enquanto obra de arquitetura e como lugar de ensino, dotando-a de bons espaços formais e informais propícios a um ensino de qualidade e de convívio.

É uma escola que se encontra numa freguesia de Lisboa, densamente povoada, extensa em termos geográficos e com uma grande diversidade urbanística, tendo como caraterística principal o facto de possuir na sua vizinhança zonas degradadas, habitadas por famílias de classe baixa e média baixa, e outras zonas, de construção recente, onde habitam famílias com rendimentos acima da média nacional.

A escola é caraterizada por ser uma escola tipicamente urbana, representativa da realidade a nível nacional, sendo uma escola em que os alunos obtêm resultados escolares francamente positivos, sendo constituída maioritariamente por cursos científico-humanísticos e ainda por alguns cursos profissionais.

A equipa diretiva do agrupamento de escolas onde esta se insere é constituída por 5 elementos, que zelam pela qualidade do ensino ministrado, bem como pelo bem estar físico e social dos alunos que a frequentam, a ponto de não se notar nos espaços físicos da escola

qualquer tipo de degradação. Estes elementos caraterizam a escola como sendo uma instituição que se propõe formar jovens de forma que sejam no futuro adultos íntegros, socialmente solidários, cientificamente competentes e culturalmente evoluídos.

4.2 Os professores intervenientes no estudo

Tendo em vista a condução do trabalho de campo, o investigador assistiu a aulas de Aplicações Informáticas B lecionadas pelo professor da disciplina. O professor em causa tem vários anos de experiência no ensino secundário, conhecendo bem o programa da disciplina que leciona e a forma como a generalidade dos alunos reage neste nível de ensino.

O professor que se predispôs a acompanhar o projeto tem ainda a vantagem de se sentir perfeitamente à-vontade na utilização das tecnologias que ensina em contexto de sala de aula. Aceitou integrar este grupo de trabalho após solicitação do investigador para que participasse, tendo manifestado grande vontade em fazê-lo e, quando decorreu a experiência, tinha várias turmas ao seu cuidado.

A escolha da turma teve em conta o facto de os alunos apresentarem um perfil adequado ao que se pretendia para esta investigação, nomeadamente apresentarem um bom ambiente em sala de aula e estarem predispostos a participar.

4.3 Os alunos do estudo

O estudo incidiu numa turma do Curso Científico-Humanístico de Ciências e Tecnologias, com 28 alunos, que apelidaremos de turma A, 17 do género masculino e 11 do género feminino, que se encontravam a frequentar as disciplinas de Matemática A e de Aplicações Informáticas B, no início do 2.º período de 2021/22. A média de idades rondou, no princípio do ano letivo, os 17 anos.

Os alunos que foram objeto do estudo encontravam-se a frequentar a disciplina de Matemática A no 12.º ano pela primeira vez e tinham, na totalidade, permanecido durante todo o ensino secundário na escola onde decorreu a experiência,

As classificações obtidas pelos alunos nas disciplinas de Português e de Matemática A, no final do 10.º e 11.º anos de escolaridade, encontram-se refletidas na tabela 3. Analisando a tabela apresentada verifica-se que a disciplina de Matemática A teve um insucesso de 7%, no 10.º ano, e de 4% no 11.º ano, perfeitamente em linha com os resultados obtidos por estes alunos na disciplina de Português.

No que respeita à qualidade do sucesso, isto é, alunos com classificações superiores ou iguais a 14, denota-se que 71% dos alunos apresentam uma classificação desse tipo na disciplina de Matemática A, no 10.º ano, e de 82% no 11.º ano, igualmente em linha com os resultados obtidos na disciplina de Português para os anos de escolaridade observados.

Efetuando uma análise comparativa entre as classificações obtidas nas disciplinas de Português e de Matemática A, verifica-se que existem mais alunos com classificações superiores ou iguais a 18 valores na disciplina de Matemática A.

Tabela 3 – Classificações obtidas pelos alunos da turma que foi objeto de estudo, no final do 10.º e 11.º anos, nas disciplinas de Português e de Matemática A

Classificações obtidas	Português		Matemática A	
	10.º	11.º	10.º	11.º
$8-9$	1	1	1	0
$10-13$	6	6	7	5
$14-17$	15	12	11	11
$18-20$	6	9	9	12
Totais	28			

Atendendo ao questionário efetuado (anexo II), para a maior parte dos alunos inquiridos, estudar Matemática significa somente efetuar exercícios (39%), para outros significa uma forma de compreender melhor o mundo (29%) e outros ainda têm uma perspetiva mais utilitarista, isto é, serve para ter uma boa nota no acesso ao ensino superior (14%), um aluno refere mesmo que "é uma obrigação escolar". Os alunos restantes referem que serve para colocar o raciocínio à prova (11%) ou obter mais agilidade mental (7%).

Relativamente à pergunta sobre o que gostam mais e o que gostam menos de fazer em Matemática, o tema funções foi simultaneamente o mais indicado como sendo o preferido pelos alunos (22%) e o menos preferido (29%), Esta última escolha talvez se deva às dificuldades sentidas por alguns destes alunos no cálculo. As restantes respostas no que gostam menos de fazer estão todas isoladas e referem temas como a geometria, o cálculo combinatório ou as probabilidades e mesmo a estatística, esta última por ser "pouco motivadora" refere um aluno. Destacam igualmente os processos utilizados como sendo, por vezes, algo complicados e desnecessários, isto é, muito formais.

No que concerne à pergunta sobre as atividades matemáticas mais favoritas e menos favoritas de realizar, os alunos referem que gostam mais de tarefas que envolvam a resolução de exercícios ditos rotineiros (50% dos inquiridos) e menos os exercícios que tenham alguma componente espacial ou cálculo combinatório.

Já em relação às perguntas sobre se gostam de programar e se o fazem de forma eficiente, a maioria referiu que gosta (68%), sendo que 64% refere que não se considera eficiente a programar.

Por último, na pergunta que permitia verificar se os alunos estabelecem alguma relação entre a programação e a Matemática, verifica-se que na amostra considerada de 28 alunos, 18 alunos consideram que sim, por facilitar essencialmente o raciocínio lógico, e os restantes referem o contrário, a maioria por considerar que ainda tem da Matemática um conhecimento muito elementar.

4.4 Estudos de caso

A forma de seleção dos alunos para os estudos de caso atendeu simplesmente ao facto de estes terem ou não completado as tarefas propostas. Dos 28 alunos que foram objeto do trabalho de campo efetuaram as tarefas finais solicitadas (guião 3, anexo III) vinte e quatro alunos no total utilizando uma taxonomia referida no capítulo anterior.

Todos estes vinte e quatro alunos foram escolhidos no final do 2.º período do ano letivo de 2021/22 para constituir a amostra de estudos de caso, tendo todos eles aceite sem qualquer contestação a tarefa de serem elementos informantes e respondido, no final de maio de 2022, a um questionário que pretendia obter as suas impressões sobre a aplicação dos guiões que foram aplicados no trabalho de campo e sobre o trabalho final que resolveram (anexo IV).

4.5 As aulas do estudo

A implementação do projeto na escola foi realizada através de três linhas de atuação: nas aulas de introdução à linguagem *Python*; nas entrevistas semiestruturadas que decorreram no decurso da aplicação do trabalho de campo, e, na aplicação dos questionários já depois de analisado o conteúdo dos trabalhos finais apresentados pelos alunos para estabelecer possíveis constrangimentos tidos e sucessos obtidos na elaboração dos programas e no decurso da aplicação dos guiões.

A preparação para a implementação deste projeto na escola foi iniciada em setembro de 2021, dando-se por terminado o projeto em maio de 2022, no final do 3.º período, com análise das respostas dos alunos ao questionário final.

Para o efetivo desenvolvimento do trabalho de campo, nomeadamente a aplicação dos guiões de trabalho sobre a linguagem *Python* a alunos do 12.º ano, optou-se por utilizar os

computadores da escola em detrimento das calculadoras gráficas, em três aulas de Aplicações Informáticas B, essencialmente por três ordens de razão:

1) Os alunos tinham a disciplina de Aplicações Informáticas B e tal facto constituiu uma oportunidade para efetuar interdisciplinaridade entre as disciplinas de Matemática A e de Aplicações Informáticas B.

2) O *Python* inserido nas calculadoras gráficas, embora seja bastante poderoso, permite correr somente alguns programas que recorram a bibliotecas específicas, enquanto que o leque de bibliotecas disponíveis para computadores pessoais é muito maior, nomeadamente quando se pretende recorrer a bibliotecas com potencialidades mais gráficas.

3) Por último, no 12.º ano de escolaridade, ano terminal do Ensino Secundário, a generalidade dos alunos são sujeitos a avaliação externa na disciplina de Matemática A, o que causa normalmente alguma ansiedade e desconforto nas famílias, e com maioria de razão nos alunos. Estes, por norma, costumam trabalhar bastante para conseguir uma boa classificação no exame da disciplina de Matemática A para poderem assim ingressar em cursos de ciências, como, por exemplo, cursos das áreas de Engenharia, em geral com classificações mínimas de acesso bastante elevadas. Ao efetuar esta experiência maioritariamente nas aulas de Aplicações Informáticas B, foi possível desligar o trabalho de campo da execução de exames na disciplina de Matemática A do 12.º ano sem causar muitos mais constrangimentos nos alunos e respetivas famílias, que aceitaram prontamente colaborar neste trabalho.

Nestas aulas foi monitorizada a evolução dos alunos enquanto percorriam os procedimentos associados à linguagem de programação *Python* e executavam as diversas tarefas solicitadas nos guiões de trabalho, quer questionando-os diretamente, quer observando localmente as suas prestações e interações em grupo de dois alunos ou no grupo turma, quer ainda com os professores em aula.

No que concerne à matemática requerida, e de forma a desenvolver o PC nos alunos, foram escolhidos para a realização deste estudo programas (anexo III) que permitiram obter:

- a distância entre dois pontos no plano dadas as suas coordenadas;
- o declive e a ordenada na origem da equação reduzida de uma reta dados dois pontos;
- as imagens de uma dada função real de variável real utilizando ramos;
- as soluções de uma equação do 2.º grau na forma canónica, recorrendo à fórmula resolvente;
- os termos de uma sucessão por meio da utilização do termo geral de uma progressão geométrica tendo como contexto as dobragens de uma folha de papel;
- os zeros de uma função num dado intervalo recorrendo aos métodos da bissecção e de *Newton-Raphson*.

As aulas decorreram nas datas descritas anteriormente, das 8 horas às 9 horas e trinta minutos da manhã, sempre com um desenvolvimento animado, mas ainda assim sereno e calmo, sob a orientação dos professores de Aplicações Informáticas B e de Matemática A

que, no início de cada uma das três sessões, forneciam os guiões de trabalho e introduziam os objetivos de trabalho para cada uma das sessões.

Os alunos seguiram naturalmente os guiões fornecidos e quando o resultado não foi o desejado abordavam a professora da disciplina de Aplicações Informáticas B ou, em alguns casos, o professor de Matemática A, e perguntavam o que se passava para ocorrer tal erro tentando assim depurar o erro detetado. Sempre que tinham dúvidas adotaram uma atitude de comentar com o outro elemento do mesmo grupo os diversos aspetos das tarefas propostas.

No final das 1.ª e 2.ª sessões, e após cada grupo executar o último exercício proposto e inseri-lo na sua diretoria da plataforma *Microsoft TEAMS*®, na equipa criada para o efeito (*Python* – 12.º Ano), foram fornecidas propostas de soluções para esses mesmos exercícios, tendo assim os alunos a possibilidade de confrontar a sua solução com a fornecida.

5

RESULTADOS – APLICAÇÃO DOS GUIÕES

Nas subsecções seguintes são apresentados os procedimentos e instruções que foram efetuados nas aulas de introdução ao *Python*, nomeadamente quais são os tipos de dados e operadores reconhecidos por esta linguagem; como importar um módulo ou biblioteca; como definir funções e condições causais; e, como trabalhar com estruturas de repetição, nomeadamente os ciclos *FOR* e *WHILE*, tendo por base os diversos guiões fornecidos (anexo III).

Na figura 9 encontra-se uma imagem da sala onde decorreu o trabalho de campo.

Figura 9 – Sala onde decorreu o trabalho de campo.

Seguindo o guião 1 (anexo III), os alunos aprenderam que ao criar e utilizar um programa em *Python* é possível distinguir dois tipos de ambientes: o modo de edição, onde são escritas as sequências de instruções/comandos que se pretende que o programa execute, e o modo *Shell* ou Interpretador, que se encontra diretamente ligado ao modo de edição, e que possibilita a execução das sequências de comandos criados. Na figura 10 encontra-se o Editor do programa Exemplo1.py, constituído unicamente pela instrução print(123), executado na plataforma IDLE, versão 3.10.1, e que os alunos inseriram no computador.

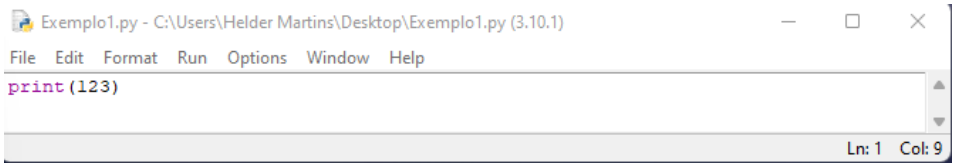


Figura 10 – Editor do programa Exemplo1.py

A figura 11 apresenta o resultado da execução do programa Exemplo1.py.

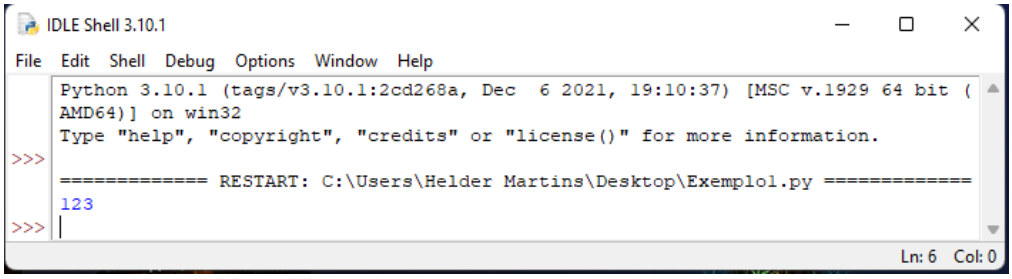


Figura 11 – Resultado da execução da função *print*() no Interpretador.

Pretende-se com estas subsecções descrever como as aulas decorreram e que tipos de comandos e procedimentos foram efetuados pelos alunos na introdução à linguagem *Python*, dando assim uma ideia de como um professor ou um aluno neste nível de ensino podem iniciar o seu trabalho com esta linguagem de programação.

Os exemplos de programas apresentados podem facilmente ser substituídos por outros de igual eficácia que se adaptem ao ano de escolaridade em que se encontre o aluno.

5.1 Dados, operadores, importação de bibliotecas e funções em *Python*

A professora de Aplicações Informáticas B (AI) deu início à primeira aula deste projeto, no dia 4 de março de 2022, referindo a razão pela qual nos guiões estarem sempre duas imagens associadas, nomeadamente:

> Professora: Meninos, no exercício que vocês vão fazer, não têm o editor e o compilador em documentos separados [referindo-se ao guião 1]. O que está no vosso lado esquerdo é o que têm de escrever. O que está no lado direito é o que o computador vos vai devolver. (4_3_2022)

Seguindo o guião 1 (anexo III), passou-se para a análise dos diversos tipos de dados possíveis de trabalhar com a linguagem *Python*. Os mais usados nesta linguagem são os

números, inteiros (*int*), sem parte decimal, ou reais (*float*), com parte decimal[6]; os valores lógicos, verdadeiro (*True*) ou falso (*False*); e, as cadeias de texto (*strings*), sequências de carateres entre aspas. Para armazenar estes dados em memória utilizam-se variáveis.

Podem igualmente considerar-se operadores, que permitem comparar diversos valores como, por exemplo, '==', '<' ou '>='. Ao comparar a com b, como, por exemplo, $a < b$, os termos são comparados respeitando a ordem. Ao efetuar uma comparação é criado um valor lógico (verdadeiro ou falso). Os operadores booleanos: *and* (e) e *or* (ou), seguem igualmente este princípio.

Na figura 12 encontram-se exemplos de números, de valores lógicos e de cadeias de texto, bem como de operadores, sugeridos no guião 1 (anexo III) aos alunos. Para visualizar o valor das variáveis no Interpretador utiliza-se a função *print*().

Editor:

```
a=2
x,y=3,4.02
nome1="Olá"
nome2=" Rui"
print("a =",a)
print(nome1+nome2)
print(y==4 or x*y<=12)
```

Interpretador:

```
a = 2
Olá Rui
False
```

Figura 12 – Editor e Interpretador com a introdução de números e cadeias de texto em Python.

No interpretador encontram-se a apresentação do número 2; a junção das palavras *Olá* e *Rui* numa única frase; e, por último, o valor lógico retornado pela instrução *print*(y==4 or x*y≤12), que é falsa.

Na figura 13, está a sequência efetuada pelo grupo dos alunos F_8 e H_8 e o resultado obtido, depois de gravar e correr o programa.

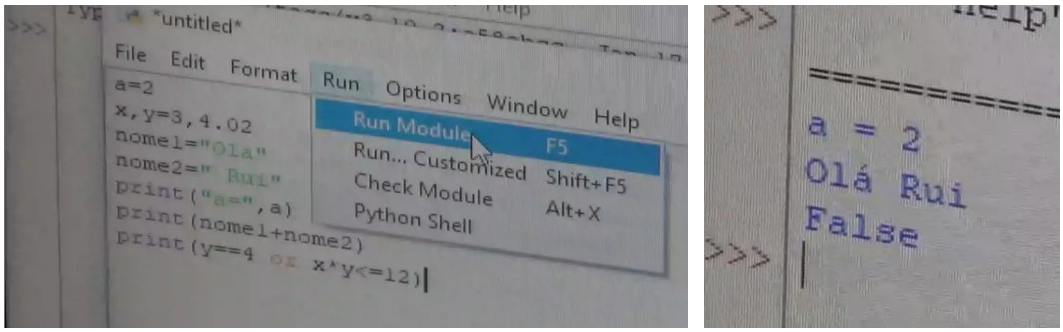


Figura 13 – Sequência introduzida pelos alunos F_8 e H_8 e resultado obtido.

[6] O símbolo da vírgula (',') nos números com parte decimal, é substituída por ponto ('.').

Alguns grupos levantaram nesta fase algumas questões de génese instrumental, nomeadamente ao não introduzir o espaço no início da palavra 'Rui', a sequência das palavras 'Olá' e 'Rui' vêm juntas. A esses alunos foi-lhes transmitido que teriam que efetuar essa alteração no programa e executá-lo de novo. A professora de AI relembrou que no *Visual Basic* acontece o mesmo.

A linguagem *Python* contém igualmente algumas funções na sua raiz que permitem realizar diversos tipos de operações, quer estas sejam aritméticas, quer de outro tipo matemático. Assim, é possível obter: o valor absoluto de x (*abs*(x)); o simétrico de x ($-x$); converter uma variável x num número inteiro, retirando a parte decimal (*int*(x)); converter x num número real (*float*(x)); arredondar x com n casas decimais (*round*(x, n)); e, obter o menor ou o maior valor entre x e y, *min*(x, y) e *max*(x, y), respetivamente.

É igualmente possível operacionalizar x com y, nomeadamente para a adição, subtração, multiplicação, divisão e potenciação, $x + y$, $x - y$, $x * y$, x/y, $x ** y$, respetivamente. Consegue-se ainda obter o quociente da divisão inteira de x por y e o resto dessa divisão ($x//y$ e $x\%y$, respetivamente), como ilustra a figura 14 com alguns exemplos.

Editor:

```
print(abs(-1.23))
print(int(-12.35))
print(float(7))
print(round(2.587,1))
print(min(1.4,2.5))
print(5*3)
print(11//3)
print(17%4)
```

Interpretador:

```
1.23
-12
7.0
2.6
1.4
15
3
1
```

Figura 14 – A apresentação no Editor e no Interpretador de operações unárias/binárias.

Na figura 15 encontra-se a sequência de instruções escritas pelas alunas F_9 e F_{10} no compilador (interpretador), antes de as copiar e importar para um programa próprio.

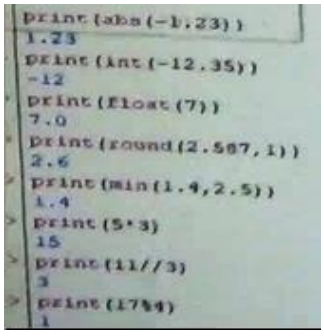


Figura 15 – Pormenor das instruções introduzidas pelo grupo das alunas F_9 e F_{10} antes de as importar para o Editor de um programa.

A linguagem *Python* permite igualmente efetuar a formatação de números quando se utiliza a instrução *print*(), evitando assim que os números apresentados no Interpretador tenham um número excessivo de casas decimais, possibilitando que as variáveis definidas apareçam numa determinada posição de uma frase que se pretende colocar no Interpretador. A figura 16 apresenta os casos dados no guião 1 (anexo III).

Editor:

```
a,b=2.357,1.2
print("O valor de a é {}.".format(a))
print("Os valores de a e b são {} e {}.".format(a,b))
print("O valor de a arredondado com uma casa decimal é {:.1f}".format(a))
```

Interpretador:

```
O valor de a é 2.357.
Os valores de a e b são 2.357 e 1.2.
O valor de a arredondado com uma casa decimal é 2.4
```

Figura 16 – Exemplos de utilização da instrução *format*() no *print*().

Por vezes, quando se trabalha com cadeias de texto, é também útil saber o número de carateres da sequência x (*len*(x)); ou o carater da sequência x que está na ordem k+1 (x[k]) (figura 17).

Editor:

```
a="Python"
print(len(a))
print(a[3])
```

Interpretador:

```
6
h
```

Figura 17 – Exemplos de utilização de cadeias de texto.

Uma das caraterísticas da linguagem que é muito apreciada pelos seus utilizadores, pois permite que um dado programa ao ser executado seja bastante rápido a 'correr', é a possibilidade que se tem de importar bibliotecas ou módulos para que o programa consiga utilizar determinado tipo de funções. Nestes casos utiliza-se a instrução *import* *

Um dos módulos mais utilizados é a biblioteca *math*, que contem algumas funções importantes, como sejam a raíz quadrada (*sqrt*()); as funções trigonométricas (*sen*(), *cos*() ou *tan*(), por exemplo); ou constantes matemáticas como, por exemplo, o pi (π). Na figura 18 encontram-se os exemplos de utilização desta biblioteca introduzidos no computador pelos alunos e que são sugeridos no guião 1 (anexo III).

Editor:

```
from math import *
print(sqrt(3))
print(cos(pi/6))
```

Interpretador:

```
1.7320508075688772
0.8660254037844387
```

Figura 18 – Exemplos de utilização da biblioteca *math*.

A leitura de dados de forma interativa no Interpretador implica a introdução no Editor da função *input*(). Esta instrução permite a criação de uma variável pela leitura de um símbolo ou sequência de símbolos, introduzido(s) pelo utilizador, que poderá(ão) ser convertida(os) em número, inteiro ou real, ou permanecer como cadeia de texto, conforme a instrução do Editor (figura 19).

Editor:

```
nome = input("nome? ")
n = int(input("n= "))
pi = float(input("pi= "))
```

Interpretador:

```
nome? Ana
n= 6
pi= 3.14
```

Figura 19 – Leitura interativa de dados e respetivo comportamento nos ambientes Editor e Interpretador.

Outra das grandes vantagens de utilizar o *Python* reside na utilização de funções, que não são mais do que estruturas que agrupam sequências de instruções.

Para caraterizar uma função em *Python* utiliza-se a instrução *def* seguida do nome da função, separada da palavra *def* por um espaço, e o(s) argumento(s) entre parenteses, separados por vírgulas. Esta linha termina com dois pontos (':'), 'forçando' a indentação[7] (um avanço para a direita) nas linhas seguintes.

Por último, e nas linhas seguintes, tem de se incluir a sequência de instruções necessárias para atingir o objetivo definido (figura 20).

```
def nome(argumento1,argumento2,…) :
    sequência de instruções
```

Figura 20 – Esquema de utilização do comando *def*.

Na figura 21 encontra-se o exemplo de um programa definido por uma função que permite obter a distância entre dois pontos no plano, dadas as suas coordenadas e que foi introduzido pelos alunos no computador.

Editor:

```
from math import *
def dist(xA,yA,xB,yB) :
    d=sqrt((xA-xB)**2+(yA-yB)**2)
    return(d)
```

Interpretador:

```
dist(2,3,-1,7)
5.0
```

Figura 21 – Editor e Interpretador do programa distância.py.

[7] A indentação em *Python* tem de ser respeitada sob prejuízo do programa não funcionar.

A criação de uma função possibilita a decomposição do problema inicialmente proposto em subcategorias e assim evitar a repetição de instruções. Uma dada função pode ser 'chamada' durante o programa tantas vezes quanto for necessário, podendo não ter argumentos, ou então também pode ser 'chamada' noutro programa, bastando para tal inserir uma instrução da função com valores nos argumentos no 2.º programa.

Na figura 22 encontra-se o enunciado do exercício proposto no final do guião 1 (anexo III).

Declive e ordenada na origem da equação reduzida de uma reta:
Escreva um programa em Python que permita devolver o declive e a ordenada na origem da equação reduzida de uma reta, dadas as coordenadas de dois pontos no plano.
A resposta deverá conter duas linhas, uma com o declive e outra com a ordenada na origem.
Nome do ficheiro: reta.py

Figura 22 – Exercício para obter o declive e a ordenada na origem da equação reduzida de uma reta.

A figura 23 apresenta o programa efetuado a *papel-e-lápis*, pelos alunos H_9 e H_{10}, do exercício final do guião 1 (anexo III), apresentado na figura 22.

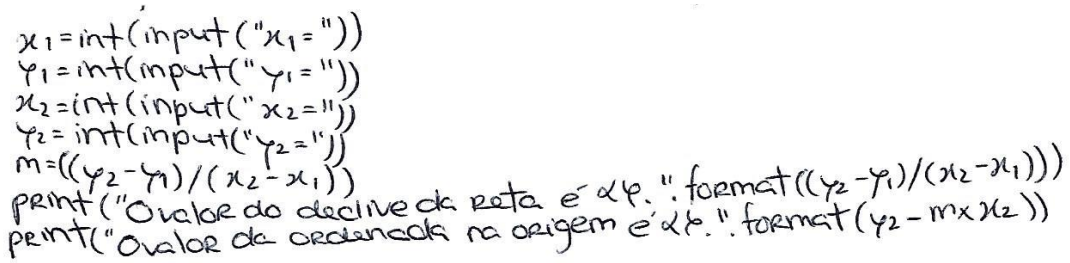


Figura 23 – Algoritmo apresentado pelos alunos H_9 e H_{10} para obter o declive e a ordenada na origem da equação reduzida de uma reta.

Posteriormente, os alunos H_9 e H_{10}, efetuaram algumas alterações à sua proposta e testaram o programa definido na figura 24, em que se utiliza a instrução *def* tal como proposto no programa da distância entre dois pontos, apresentado na figura 21.

```
File  Edit  Format  Run  Options  Window  Help
def ret(xA,yA,xB,yB) :
    m=(yA-yB)/(xA-xB)
    b=yA-m*xA
    print("Declive:{:.2f}".format(m))
    print("Ordenada na origem:{:.2f}".format(b))
```

Figura 24 – Programa para obter o declive e a ordenada na origem da equação reduzida de uma reta dadas as coordenadas de dois pontos..

Ambos os programas, quer o apresentado na figura 23, quer o apresentado na figura 24, cumprem todos os requisitos solicitados no último exercício do guião 1 (anexo III). A generalidade dos alunos construiu e executou o programa solicitado, uns construindo uma

função *def* semelhante à descrita na Figura 24, outros sem a definirem, como ilustrado na Figura 23.

As dificuldades sentidas pelos alunos centraram-se nas regras específicas da linguagem, como por exemplo a falta do sinal ':', no final da primeira linha da instrução *def*, e na utilização da instrução *format()*.

5.2 Função condicional (IF-ELIF-ELSE)

Na aula de dia 11 de março de 2022 a professora de AI começou por relembrar o que se entende por um algoritmo, tomando como exemplo o programa que os alunos tinham efetuado na sessão anterior para determinação do declive e da ordenada na origem de uma reta.

Após este momento inicial passou à descrição do que se entende por uma função condicional em *Python* em que tem que se considerar uma condição e conforme esta se torne numa proposição verdadeira ou numa proposição falsa, mediante o teste de um determinado valor, assim o programa executa uma instrução A, uma instrução B, ou uma instrução C. A sintaxe geral encontra-se na figura 25.

```
If condição1:
    Instrução A
elif condição2:
    Instrução B
else:
    Instrução C
```

Figura 25 – Estrutura de uma função condicional.

Seguindo o guião 2 (anexo III), e para ilustrar este comando, os alunos consideraram a função f que se encontra definida na figura 26, sendo o algoritmo de aplicação da função e o respetivo programa definido na figura 27. O objetivo é o de calcular a imagem de um determinado objeto desta função real de variável real.

$$f(x) = \begin{cases} 2x - 1 & se\ x < 0 \\ \sqrt{2 + x} & se\ 0 \leq x < 2 \\ x^2 + 3 & se\ 2 \leq x < 4 \\ \dfrac{1}{x} & se\ x \geq 4 \end{cases}$$

Figura 26 – Função real de variável real definida por ramos.

Algoritmo:

função f
  ler x
  Se $x < 0$ então
    $y = 2x - 1$
  Se $(x \geq 0\ e\ x < 2)$ então
    $y = \sqrt{2 + x}$
  Se $(x \geq 2\ e\ x < 4)$ então
    $y = x^2 + 3$
  Se $x \geq 4$ então
    $y = 1/x$
  devolver y

Programa funcao:

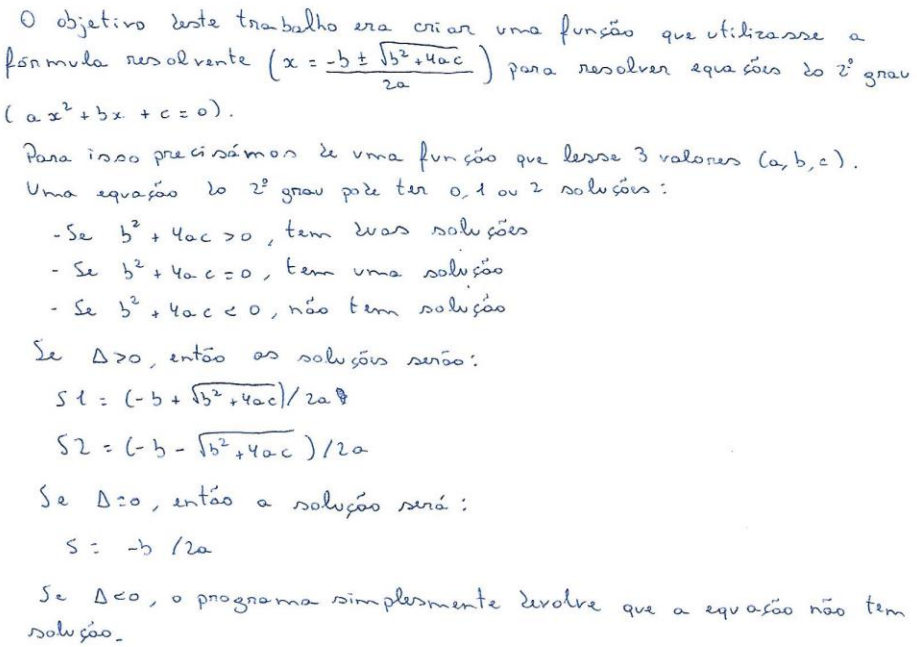
```
def f(x):
 if x<0:
    y=2*x-1
 elif (x>=0 and x<2):
    y=(2+x)**(1/2)
 elif (x>=2 and x<4):
    y=x**2+3
 else
    y=1/x
 return(y)
```

Figura 27 – Algoritmo e programa função para a função f, usando uma função condicional.

Após a inserção do programa função, todos os alunos testaram alguns valores, sendo que para obter, por exemplo, a imagem de 0, basta introduzir no interpretador, f(0).

No seguimento da aula e tal como consta do guião 2 (anexo III), foi solicitado aos alunos que efetuassem um algoritmo e um programa que permitisse determinar as soluções de uma equação do 2.º grau dada na forma canónica.

Na figura 28 encontra-se o relatório com o algoritmo proposto pelo grupo de alunos *F8* e *H8* do programa sobre a fórmula resolvente do 2.º grau apresentado no final do guião 2 (anexo III).



O objetivo deste trabalho era criar uma função que utilizasse a fórmula resolvente $\left(x = \frac{-b \pm \sqrt{b^2+4ac}}{2a}\right)$ para resolver equações do 2º grau ($ax^2 + bx + c = 0$).

Para isso precisámos de uma função que lesse 3 valores (a, b, c).

Uma equação do 2º grau pode ter 0, 1 ou 2 soluções:

- Se $b^2 + 4ac > 0$, tem duas soluções
- Se $b^2 + 4ac = 0$, tem uma solução
- Se $b^2 + 4ac < 0$, não tem solução

Se $\Delta > 0$, então as soluções serão:

$S1 = (-b + \sqrt{b^2+4ac})/2a$

$S2 = (-b - \sqrt{b^2+4ac})/2a$

Se $\Delta = 0$, então a solução será:

$S = -b/2a$

Se $\Delta < 0$, o programa simplesmente devolve que a equação não tem solução.

Figura 28 – Relatório com o algoritmo proposto pelos alunos *F8* e *H8* para o programa da fórmula resolvente.

O algoritmo descrito pelos alunos contém os diversos passos ao qual o programa deve obedecer para que se consigam obter as diversas soluções de uma equação do 2.º grau, dados os coeficientes da equação na forma canónica.

Os alunos F_3 e F_4, propuseram a solução que se encontra na figura 29 e cuja apresentação, testando algumas condições, está na figura 30.

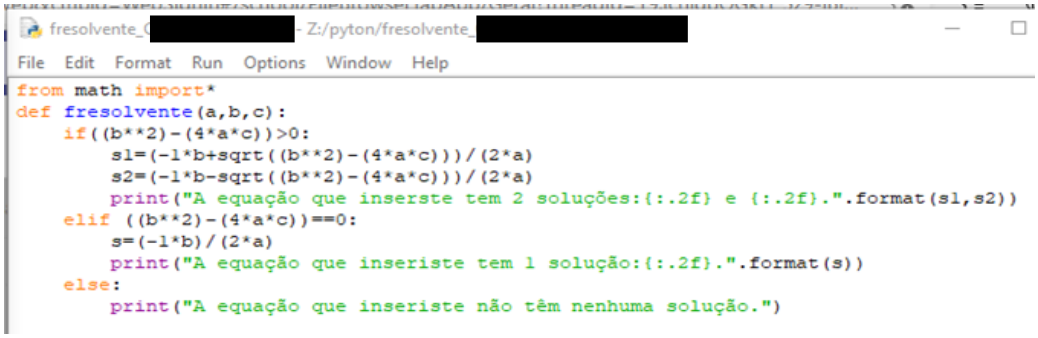


Figura 29 – Programa da fórmula resolvente elaborado pelas alunas F_3 e F_4

O programa apresentado na figura 29 recorre à biblioteca *math* e estabelece uma função baseada numa instrução if-elif-else que abarca todos os casos possíveis.

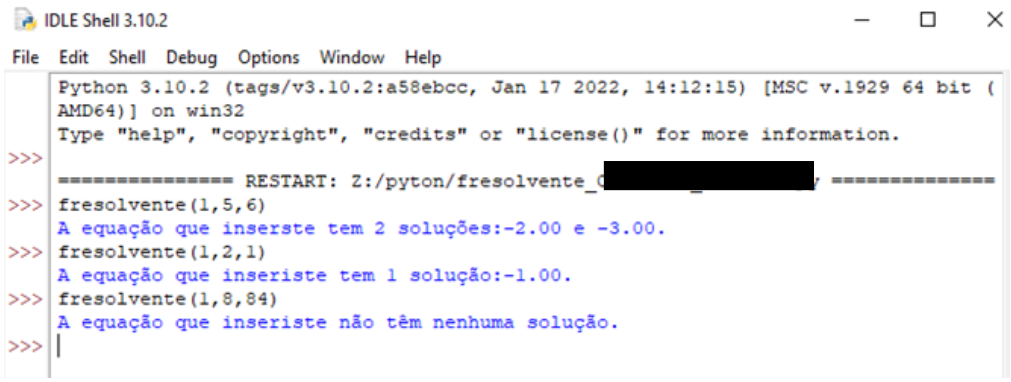


Figura 30 – Testagem do programa da fórmula resolvente, efetuado pelas alunas F_3 e F_4, para três tipos de equações.

Assinale-se que os alunos F_8 e H_8 apresentaram um programa bastante semelhante ao desenvolvido pelas alunas F_3 e F_4, ao contrário do grupo de alunos H_3 e H_4 que apresentou um código algo diferente e que se encontra na figura 31.

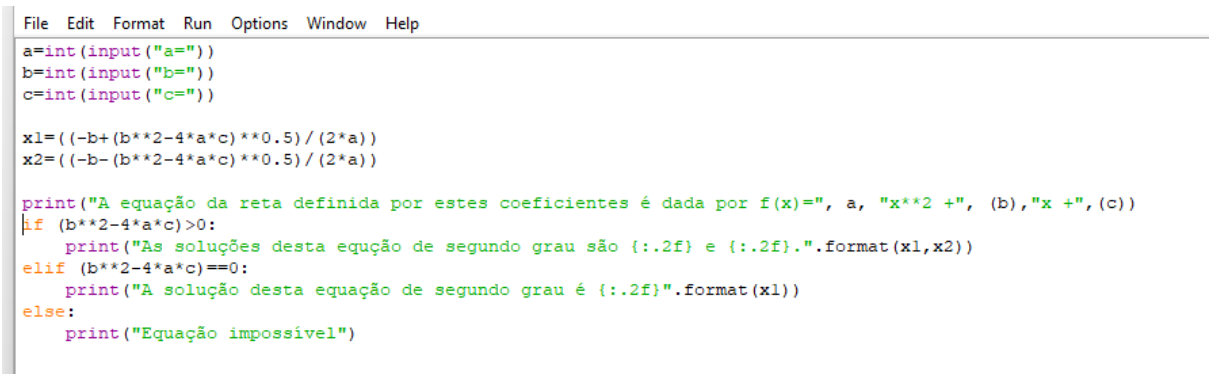


Figura 31 – Programa com a fórmula resolvente de equações do 2.º grau do grupo de alunos H_3 e H_4,

Como se pode constatar, o programa apresentado pelos alunos H_3 e H_4 não recorre à definição de uma função utilizando o comando *def*, permitindo a introdução dos coeficientes da equação do 2.º grau de uma forma mais interativa, e, por outro lado, define dois valores iniciais (x1 e x2), que podem existir ou não, mas que só são apresentados se forem satisfeitas determinadas condições.

Estes dois programas permitem ambos determinar as soluções de uma equação do 2.º grau caso estas existam. Temos aqui um exemplo concreto, apresentado por estes dois grupos de alunos, em que a criação de um programa, aparentemente simples, pode incentivar o gosto pela Matemática, sendo que os alunos envolvidos, sabendo que um resultado semelhante pode ser obtido usando dois ou mais tipos de código diferentes, os incentiva a serem mais criativos e a demonstrar autoconfiança na procura de procedimentos matemáticos adequados.

Também nesta 2.ª sessão, e à semelhança do que já havia acontecido na 1.ª aula embora com menor incidência, até porque o programa solicitado era mais simples, verificou-se que os alunos percorreram na generalidade as cinco práticas identificativas do PC para elaborar o programa solicitado, pois para resolver o 'seu problema' verificaram a sua abstração, pela tentativa de diminuírem a complexidade do problema inicial; decomposição, por terem dividido o problema em partes mais pequenas, reconhecimento de padrões, pelo reconhecimento de regularidades; algoritmia, por terem que respeitar determinados passos e eventualmente a depuração, por os alunos terem muito provavelmente de corrigir erros ou imprecisões para conseguir obter o programa pretendido (Albuquerque, 2022; Espadeiro, 2022; Carvalho e Silva, 2022).

Igualmente se encontram os três tipos de raciocínio apontados por Ponte et al. (2020), nomeadamente o dedutivo, pelas inferências que vão sendo estabelecidas através da consciência que o sinal do binómio descriminante determina o número de soluções da equação sendo que os alunos têm que introduzir no código funções condicionais, e os indutivo e abdutivo em conjunto, por os alunos criarem a noção, através da experimentação com casos particulares, que existe uma determinada regra entre os parâmetros que determina o número de soluções,

5.3 Estruturas de repetição

Na terceira e última aula, que decorreu no dia 18 de março de 2022, foram analisadas pelos alunos as estruturas de repetição, nomeadamente os ciclos *FOR* e *WHILE*.

Assim, a professora de AI deu início aos trabalhos esclarecendo os alunos como poderiam fazer para incrementar uma variável em *Python*, definindo em seguida o que se entende por estruturas de repetição. Esclareceu que para a execução de certos programas é útil repetir uma ou várias instruções um determinado número finito de vezes. Se soubermos antecipadamente o número de repetições que é necessário efetuar, pode-se utilizar um ciclo *FOR*. A sintaxe geral encontra-se na figura 32.

```
for variável in range():
    Instrução A
    Instrução B
    …
```

Figura 32 – Estrutura de um ciclo *FOR*.

Ressalve-se que em *Python*, o contador do ciclo *FOR* se inicia, por defeito, em 0, e termina em $b-1$, com passo igual a 1, caso a instrução seja *range(b)*. Caso a instrução dada seja *range(a,b)*, o contador inicia-se em a e percorre todos os números inteiros até $b-1$, sendo que no caso de a instrução dada ter a sintaxe *range(a,b,c)*, altera-se o passo e em vez de ser de 1 em 1, é de c em c. Na figura 33 encontra-se o enunciado de um problema cuja solução pode ser encontrada elaborando um programa em *Python* com um ciclo *FOR*.

Dobragens de uma folha de papel:
Pretende-se determinar a espessura do aglomerado de folhas de papel que se obtêm ao dobrar sucessivamente um determinado número de vezes uma folha de papel com uma espessura inicial de 0,1 mm. Em particular, qual é a espessura das folhas de papel que se obtêm quando se efetuam 10 dobragens? Qual é o número mínimo de dobragens que serão necessárias para que se obtenha uma espessura superior à distância da Terra à Lua (384 400 km ou 384 400 000 000 mm)?

Figura 33 – Problema das dobragens de uma folha de papel.

O algoritmo e um programa em *Python* que permite resolver o problema apresentado na figura 33 encontra-se na figura seguinte (figura 34).

Algoritmo:

```
função dob
  ler n
  e = 0.1
  Para i de 1 até n :
     e = 2e
  devolver e
```

Ficheiro dobragens.py:

```
def dob(n) :
   e = 0.1
   for i in range(1,n+1):
        e = 2 * e
   return(e)
```

Figura 34 – Algoritmo e programa para resolver o problema das dobragens de uma folha de papel.

Correndo o programa dobragens.py, descrito na figura 34, os alunos verificaram que com 10 dobragens se obtém uma espessura de 1,024 dm e que são necessárias 42

dobragens para obter uma espessura superior à distância da Terra à Lua, atingindo uma espessura de 439804651110,4 mm.

Caso se pretenda repetir uma ou várias instruções, um número indeterminado de vezes, é aconselhável utilizar um ciclo não limitado *WHILE* (enquanto) cuja sintaxe geral se encontra na figura 35.

```
while condição:
    Instrução A
    Instrução B
    …
```

Figura 35 – Estrutura de um ciclo *WHILE*.

Como se pode constatar pela análise da sintaxe do comando *WHILE*, não existe nenhuma instrução para indicar o fim de ciclo. O fim obtém-se através da indentação de instruções. Na figura 36 encontra-se o enunciado de um problema que pode ser resolvido em *Python* usando um ciclo *WHILE*.

Método da bissecção:
Considere-se a função f definida em $[-1,2]$ definida por $f(x) = -x^3 + x + 1$.
Utilize a calculadora gráfica para traçar uma representação gráfica de f .
Determine a solução da equação $f(x) = 0$ utilizando o algoritmo conhecido por "método da bissecção".

Figura 36 – Proposta de problema para resolver a equação $-x\text{^}3 + x + 1 = 0$.

A utilização do método da bissecção para determinar os zeros da função f, isto é, determinar as soluções da equação $f(x) = 0$, baseia-se no Corolário do Teorema de Bolzano-Cauchy e, sendo assim, pressupõe-se que a função dada, f, é contínua em $[a, b]$ e que se tem que $f(a) * f(b) < 0$. Como tal, admite-se que a solução, c, se encontra em $]a, b[$ e determina-se um valor para c com um erro máximo que podemos apelidar de $erro$.

O processo passa por pedir ao computador para calcular $f(a)$ e $f\left(\frac{a+b}{2}\right)$, verificando o sinal de $f(a) \times f\left(\frac{a+b}{2}\right)$. Se este for negativo, toma-se $b = \frac{a+b}{2}$, caso contrário, toma-se $a = \frac{a+b}{2}$ e assim sucessivamente até $b - a < erro$.

O programa permite obter um intervalo de amplitude igual ou inferior a $erro$ onde se encontra a solução pretendida. Um possível algoritmo e o respetivo programa que resolve o problema proposto na figura 36 encontra-se na figura 37.

Algoritmo:

função f
  devolver $-x^3 + x + 1$

função biss
  ler $a, b, erro$
  Enquanto $(b - a) > erro$
    $c = (a + b)/2$
    Se $f(a) * f(c) < 0$:
      $b = c$
    Senão:
      $a = c$
  devolver a, b

Ficheiro bissecção.py:

```
def f(x) :
  return(-x ** 3 + x + 1)

def biss(a,b,erro)
  while (b - a) > erro:
    c = (a + b)/2
    if f(a) * f(c) < 0:
        b = c
     else:
        a = c
  return(a,b)
```

Figura 37 – Algoritmo e programa para a resolução da equação $-x\text{^}3 + x + 1 = 0$.

Tendo por base o programa bissecção e escrevendo biss(-1,2,0.001) (valor mínimo do intervalo $-1{,}2$, valor máximo 0 e erro máximo de 0,001) no Interpretador, tem-se que o intervalo onde se encontra a solução com amplitude 0,001 é $[1{,}32470703125; 1{,}325439453125]$ (a solução obtida na calculadora é 1,324717957).

Por último, e na continuação da aula, foi proposto aos alunos que investigassem na internet o que é o método de Newton-Raphson e que escrevessem um programa em Python que permitisse devolver valores aproximados para os zeros da função f, considerando que a expressão analítica da função f é dada por: $f(x) = -x^3 + x + 1$ em $[-1{,}2]$. Deveriam dar ao programa assim construído o nome de newton.py

A generalidade dos grupos conseguiu elaborar um programa nas condições solicitadas. Na figura 38 encontra-se o programa obtido pelos alunos F_8 e H_8 .

```
def F(x) :
    return((-x**3)+x+1)
def f(x) :
    return(-3*(x**2)+1)
def newton(x,erro) :
    x1=x-(F(x)/f(x))
    while abs(x1-x)>=erro:
        x=x1
        x1=x1-(F(x1)/f(x1))
    return(x1)
```

Figura 38 – Programa elaborado pelos alunos F_8 e H_8 e baseado no método de *Newton-Raphson* para obtenção dos zeros da função f(x)=-x^3 + x + 1.

Introduzindo a instrução ‘ newton(-1,0.001)’ no compilador, obtém-se como valor da solução 1.3247187886152572 a qual está correta, como se pode comprovar facilmente recorrendo, por exemplo, a uma calculadora gráfica.

No final do guião 3 (anexo III) é apresentada uma proposta de programa newton.py para resolver o problema apresentado para que aqueles alunos que obtiveram uma solução possam comparar com a sua e simultaneamente dar aos alunos que não conseguiram elaborar um programa tenham a possibilidade de testarem posteriormente uma solução..

```
from math import *
def f(x) :
   return(−x ** 3 + x + 1)
def df(x)
   return(−3 * x ** 2 + 1)
erro=10**-10
n=100
x0=0.5
x1=x0-f(x0)/df(x0)
i=1
while  (abs(x1-x0)>=erro  and
i<n):
   x0=x1
   x1=x1-f(x1)/df(x1)
   i=i+1
print(x1)
```

Figura 39 – Proposta de programa newton.py apresentada no guião 3.

O programa apresentado na figura 38, da autoria dos alunos F_8 e H_8, permite obter a mesma solução final que o programa que consta do guião 3 (anexo III), apresentado na figura 39, embora seja ligeiramente diferente, nomeadamente permite a introdução de um erro específico e não contem um contador.

Este problema constitui-se como uma tarefa de investigação em que os alunos desenvolvem procedimentos matemáticos para que consigam obter uma solução do problema utilizando a linguagem *Python*. No entanto, permite igualmente que os alunos desenvolvam o PC nas suas diversas vertentes, bem como apliquem os diversos tipos de raciocínio analisados anteriormente, com maior incidência nos raciocínios dedutivo e indutivo.

O facto de os alunos terem que utilizar instruções e uma simbologia adequada, em que, por exemplo, a não colocação de dois pontos nos locais adequados faz toda a diferença, permite que os alunos deem um maior enfoque a aspetos formais da linguagem, pois sem a sua correta compreensão os programas não funcionam possibilitando assim que interiorizem aspetos ligados à Matemática e à sua formalização.

Quando se constrói uma condição num programa sem que se perceba bem o contexto, tem de se compreender igualmente o papel dos conectivos lógicos e das expressões

proposicionais e nesse sentido é fundamental que os alunos percebam as operações com condições, nomeadamente as noções de conjunção e de disjunção.

O facto de diferentes alunos resolverem um mesmo problema de maneiras distintas permite que desenvolvam diferentes competências, sendo fundamental que todos tomem contacto e reflitam sobre as diferentes abordagens possíveis o que efetivamente foi proporcionado ao disponibilizar na equipa do *Microsoft Teams*® todos os programas efetuados pelos diferentes alunos que participaram nesta experiência.

No seguimento das sessões poder-se-iam apresentar mais instruções úteis para efetuar em *Python* que seriam igualmente importantes para utilizar com a Matemática, no entanto, as instruções apresentadas nas três aulas efetuadas são o que basta para que um aluno minimamente curioso possa desenvolver programas bastante interessantes e motivadores de forma a adquirir competências relacionadas com o PC e desenvolver o raciocínio, bem como outras atitudes igualmente importantes como sejam, por exemplo, o sentido crítico, a autoconfiança e a autoestima.

6

RESULTADOS – ESTUDOS DE CASO

Como referido anteriormente, a seleção dos alunos que serviram de estudos de caso para esta investigação atendeu ao facto de estes terem completado as tarefas propostas nos guiões de trabalho. Dos 28 alunos que se propuseram no início ser observados no trabalho de campo, 24 alunos (11 do género feminino e 13 do género masculino) efetuaram o trabalho final (guião 3, anexo III), descrito igualmente na figura 40.

Exercício final:
Trabalhando com o colega de trabalho com o qual executaste estes guiões, cria um programa em linguagem *Python*, tendo por base um tema matemático aprendido no secundário, aplicando conteúdos tratados nos guiões. No final devem elaborar um vídeo, com uma duração máxima de 5 minutos, explicando o programa desenvolvido e mostrando exemplos de aplicação. Devem ainda efetuar um relatório com um máximo de 3 páginas, sem contar com a capa e eventuais anexos, onde descrevam o que foi feito. Este relatório deve ser entregue até dia 31 de março.

Figura 40 – Proposta de exercício final.

Os diversos elementos que constituem a amostra dos estudos de caso, designámos, como referido anteriormente, pelas letras *F* e *H*, respetivamente, com índices, isto é, F_k ($k \in \{1, \dots, 11\}$) e H_i ($i \in \{1, \dots, 13\}$). Foram assim considerados 12 pares para cada um dos estudos de caso, a saber: 1) F_1/F_2; 2) H_1/H_2; 3) F_3/F_4; 4) H_3/H_4; 5) H_5/H_6; 6) H_7/F_5; 7) F_6/F_7; 8) F_8/H_8; 9) H_9/H_{10}; 10) H_{11}/H_{12}; 11) F_9/F_{10}; e,12) F_{11}/H_{13}.

Nesta secção encontram-se descritas as prestações de cada um desses 12 estudos de caso, seguindo uma taxonomia própria definida de forma a relatar o trabalho efetuado por cada grupo. Em cada um dos casos é descrito o objetivo do trabalho realizado; apresentado o programa efetuado; analisados os sucessos obtidos e constrangimentos encontrados, tendo por base não só o relatório apresentado por cada grupo, mas igualmente os vídeos elaborados pelos alunos e as respostas dadas ao questionário final, o qual permitia que os alunos refletissem sobre a forma como os guiões foram aplicados no trabalho de campo e sobre os procedimentos adotados pelos alunos na elaboração do trabalho final (anexo IV).

Os estudos de caso contemplam trabalhos sobre: estudo de funções recorrendo às derivadas, juros bancários, triângulo de Pascal, Binómio de Newton, método de *Newton-Raphson*, interseções reta/reta, reta/plano e plano/plano e cálculo das médias do ensino secundário.

6.1 Estudo de caso 1 – Derivada

As duas alunas deste grupo, F_1 e F_2 , elaboraram um programa que consistia na derivação de funções polinomiais até ao 4.º grau e na apresentação gráfica das primeira e segunda derivadas. A escolha deste tema, segundo as alunas, ficou a dever-se ao fato de ser um tema bastante abordado ao longo do 12.º ano, na disciplina de Matemática A, e devido à sua "importância no estudo completo de funções, permitindo o estudo da monotonia e das concavidades dos seus gráficos", conforme as próprias referem no seu relatório.

O objetivo do seu trabalho permitiu consolidar os conhecimentos obtidos durante as aulas sobre a linguagem *Python* assim como aprofundar as suas aplicações ao nível da Matemática.

6.1.1 Desempenho escolar das alunas

Em termos de rendimento escolar as duas alunas obtiveram resultados bastante assinaláveis, sendo que na disciplina de Matemática A, a aluna F_1, obteve no final de 12.º ano, 20 valores, ficando com uma média final do ensino secundário na disciplina de 20, e a Aplicações Informáticas B (AI), a classificação obtida foi de 19 valores. A aluna F_2, obteve 17 valores na disciplina de Matemática A, no 12.º ano, ficando com uma média final de 16 na disciplina, e a AI, obteve igualmente 19, à semelhança da colega de grupo.

No exame nacional de Matemática A, a aluna F_1 obteve 198 pontos e a aluna F_2 obteve 148 pontos. São duas alunas bastante aplicadas e determinadas, com resultados nas diversas disciplinas do secundário ao nível de um quadro de excelência. Ambas pertencem inclusive ao quadro de mérito da escola.

6.1.2 Análise e objetivos do programa

O programa final efetuado (Derivada, anexo V) começa por solicitar ao utilizador os coeficientes da função polinomial pretendida, por ordem decrescente, do maior grau para o menor, atribuindo as variáveis num4, num3, num2, num1 e num0.

Utilizando o comando *If*, o programa analisa os seus valores comparando-os com zero. Caso seja diferente de zero (!=0), aplica a regra de derivação, ou seja, multiplica o coeficiente pelo monómio de grau inferior, isto é, considerando o expoente como *n* então a derivada terá expoente *n-1.* Por exemplo, a derivada de $16x^3$ é $48x^2$.

Caso o coeficiente seja zero (introduzindo a instrução *else*), a derivada do monómio toma também o valor zero, não aparecendo, portanto, na expressão da derivada. Este processo é repetido para os diferentes coeficientes inseridos pelo utilizador exceto o último, cuja derivada é sempre zero. As alunas apresentam aqui de forma consistente terem utilizado o raciocínio dedutivo.

De seguida, o programa possibilita a apresentação da expressão da derivada obtida anteriormente sob a forma de: derx3.x³+derx2.x²+derx1.x+derx0 sendo que derx3=4.num4, derx2=3.num3, derx1=2.num2 e derx0=num1. O código utiliza então a função apresentada para criar um gráfico de f', em [-10,10]. A partir da análise feita pelo programa nos intervalos em que f'(x)≤0, o programa deduz os intervalos de monotonia da função f (utiliza a biblioteca *numpy*, que recorre ao que vulgarmente se designa por CAS – cálculo algébrico simbólico).

Repetindo o processo anterior, o programa deduz uma expressão para a segunda derivada sob a forma de der2x2.x**2+der2x1.x+der2x0 sendo que der2x2=derx3.3, der2x1=derx2.2 e der2x0=derx1. Apresenta, em seguida, o gráfico da segunda derivada de f, em [-10,10]. Finalmente, através de uma análise idêntica à anterior, aparecem os intervalos do sentido das concavidades do gráfico da função f.

Por exemplo, considerando a função f, definida por: $f(x) = 4x^4 + 5x^3 - 2x^2 - 8x + 9$, basta escrever fx para indicar a função e os respetivos coeficientes entre parênteses, separados por virgulas, como se pode observar na figura 41.

```
Insira os coeficientes de uma função polinomial até
ao grau 4 de forma decrescente de maior grau:
Exemplo: fx(-5,4,-6,2,-3)
fx(4,5,-2,-8,9)
```

Figura 41 – Introdução no programa efetuado dos coeficientes da função polinomial..

Em seguida, carregando em *Enter,* obtém-se a derivada da função na linha seguinte e é aberta uma nova janela com a sua representação gráfica (figura 42).

```
Insira os coeficientes de uma função polinomial até
ao grau 4 de forma decrescente de maior grau:
Exemplo: fx(-5,4,-6,2,-3)
fx(4,5,-2,-8,9)
f'(x)= 16 x^3 + 15 x^2 -4 x -8
Para calcular a segunda derivada clique na cruz da
janela com o gráfico
```

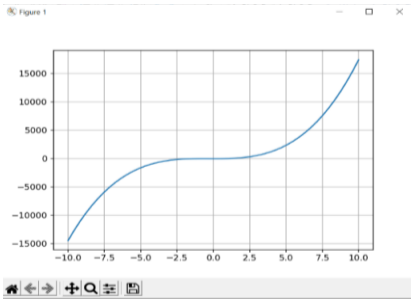


Figura 42 – Derivada da função introduzida e respetiva representação gráfica..

Carregando na cruz da nova janela com o gráfico, surgem os intervalos de monotonia da função f, a expressão da segunda derivada e o respetivo gráfico (figura 43).

```
Insira os coeficientes de uma função polinomial até ao grau
4 de forma decrescente de maior grau:
Exemplo: fx(-5,4,-6,2,-3)
fx(4,5,-2,-8,9)
f'(x)= 16 x^3 + 15 x^2 -4 x -8
Para calcular a segunda derivada clique na cruz da janela c
om o gráfico
A função é descrescente em Interval(-oo, 0.646307176604227)
A função é crescente em Interval(0.646307176604227, oo)
f''(x)= 48 x^2 + 30 x -4
```

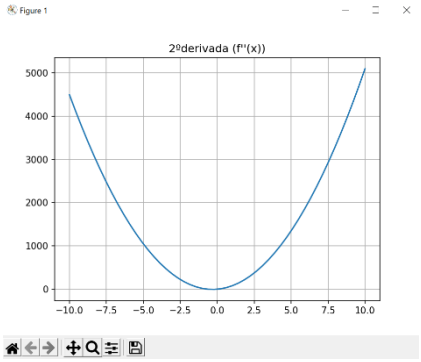


Figura 43 – Intervalos de monotonia e segunda derivada da função introduzida, bem como respetiva representação gráfica..

Por último, carregando na cruz da nova janela, com o gráfico da segunda derivada, surgem analisados os intervalos do sentido das concavidades do gráfico da função (figura 44)..

```
Insira os coeficientes de uma função polinomial até ao grau 4 de forma decres
cente de maior grau:
Exemplo: fx(-5,4,-6,2,-3)
fx(4,5,-2,-8,9)
f'(x)= 16 x^3 + 15 x^2 -4 x -8
Para calcular a segunda derivada clique na cruz da janela com o gráfico
A função é descrescente em Interval(-oo, 0.646307176604227)
A função é crescente em Interval(0.646307176604227, oo)
f''(x)= 48 x^2 + 30 x -4
O gráfico da função tem a concavidade voltada para cima em Union(Interval(-oo
, -0.737928705347128), Interval(0.112928705347128, oo))
O gráfico da função tem a concavidade voltada para baixo em Interval(-0.73792
8705347128, 0.112928705347128)
```

Figura 44 – Concavidades do gráfico da função introduzida.

Como se pode constatar é um programa que permite analisar conteúdos das Aprendizagens Essenciais da disciplina de Matemática A relacionados com o cálculo diferencial, sendo aplicado exclusivamente a funções polinomiais de grau inferior ou igual a 4.

6.1.3 Processos utilizados e fontes procuradas

Tal como as autoras do programa referem no questionário final, pensaram num processo semelhante ao de encontrar as raízes de uma equação do 2.º grau, calculando em seguida os intervalos de monotonia da função e da concavidade do gráfico da função considerada, adicionando os respetivos gráficos da 1.ª e 2.ª derivadas utilizando para tal a biblioteca *sympy*.

Para a construção dos diversos programas solicitados, basearam-se nos guiões fornecidos, bem como recorreram a conhecimentos básicos de matemática, como, por exemplo, à fórmula resolvente de equações do 2.º grau, e no que concerne à aplicação do método de Newton-Raphson, recorreram à internet, como sugerido no guião respetivo.

6.1.4 Dificuldades sentidas e raciocínio matemático envolvido

Para a execução do programa final os alunos consideram que tiveram algumas dificuldades em encontrar a biblioteca que possibilitou a criação dos gráficos, e em particular do seu descarregamento. Sentiram igualmente dificuldades na determinação dos intervalos de monotonia e do sentido das concavidades, nomeadamente em obter aproximações dos valores dos intervalos, isto é, dos zeros da 1.ª e 2.ª derivadas.

Relativamente à concretização dos guiões consideram que, quer o guião 1, quer o guião 2, permitem uma fácil compreensão das instruções básicas da linguagem *Python*, nomeadamente destacam os exemplos fornecidos pela sua clareza e facilidade de compreensão, no entanto, relativamente ao guião 3, consideram que foi dado pouco tempo para a pesquisa e execução do método de *Newton-Raphson*.

Aparentemente, as dificuldades sentidas por estas alunas na construção do programa solicitado que revelasse a aplicação do método citado ficou a dever-se à complexidade dos conceitos matemáticos envolvidos, que fez com que não chegassem a apresentar um processo definitivo que permitisse calcular os zeros de uma função tendo por base este método

A tarefa mais fácil de executar nos guiões foi a relacionada com a obtenção de um programa de aplicação da fórmula resolvente do 2.º grau e a mais complexa foi a relativa à aplicação do método de *Newton-Raphson*, essencialmente por não terem percebido muito bem como aplicar este método à programação em *Python*, especialmente como obter as sucessivas aproximações.

Segundo estas alunas a programação em *Python* ajuda a desenvolver o raciocínio matemático, pois por mais simples que sejam os programas a executar têm por base a lógica matemática. Esta afirmação é comprovada por uma análise adequada do código efetuado

pelas alunas no seu projeto (Derivada, anexo V), nomeadamente quando constroem a 1.ª e 2.ª derivadas.

Tal como as alunas referem e é visível no seu trabalho, utilizaram raciocínio matemático na construção do seu projeto, nomeadamente os raciocínios dedutivo e indutivo, nomeadamente quando constroem as duas funções condicionais para construção das derivadas.

Consideram que o trabalho solicitado foi útil, quer por permitir uma consolidação da matéria na disciplina de Matemática, além de possibilitar encontrar novos métodos para resolver o mesmo problema como seja a utilização do método de *Newton-Raphson* em oposição ao método da bisseção, quer por ter possibilitado aprender a linguagem *Python*.

Em termos metodológicos não consideraram que fosse útil o facto de terem que efetuar um vídeo e um relatório sobre o programa, pois segundo as alunas um dos dois instrumentos bastaria para explicar o funcionamento do programa. Em sua opinião o vídeo seria mais útil pois possibilita a exploração simultânea do código e do programa final efetuado

Referem como conclusão do seu trabalho o seguinte:

> Neste trabalho foram abordados aspetos básicos e fundamentais da linguagem de programação *Python* assim como da sua aplicação na matemática, concretamente no estudo de funções. Deste modo, o nosso projeto permitiu-nos perceber a importância desta linguagem de programação já que por ser uma linguagem de fácil leitura e que precisa de poucas linhas de programação, pode ser aplicada por qualquer um de nós no futuro (Extraído do relatório das alunas F_1 e F_2 sobre o trabalho solicitado).

As alunas enfatizaram igualmente o facto de que sendo um projeto essencialmente prático conseguirem aplicar conhecimentos de outras formas sem recorrer a métodos convencionais de avaliação.

6.1.5 Conclusões

Como se pode constatar pelo que as duas alunas referiram no questionário final, a aplicação dos guiões e, em particular, a elaboração do trabalho final implicou que as alunas recorressem ao raciocínio matemático, nomeadamente ao raciocínio dedutivo, facto esse que se constata ao analisar o programa elaborado pelas alunas.

No entanto, e como já foi assinalado, este tipo de tarefa envolveu igualmente o reconhecimento de padrões, caraterística fundamental do raciocínio indutivo. Se juntarmos a este tipo de raciocínios a construção de algoritmos e a depuração de erros, temos na plenitude o desenvolvimento do PC, assente na formação e consolidação de conceitos matemáticos.

Por outro lado, o facto de se terem promovido este tipo de tarefas em trabalho de projeto, fez com que as alunas tivessem de recorrer a diferentes modos de raciocínio, interligando-os, desenvolvendo uma maior compreensão do que seria expetável com, por exemplo, a resolução de uma ficha exclusivamente com exercícios rotineiros de matemática.

O programa construído constituiu-se assim como algo muito rico, que embora envolvesse um esforço suplementar por parte das alunas, permitiu que estas adquirissem outro tipo de capacidades e competências que de outra forma seria de todo impossível desenvolverem, como sejam, por exemplo, o sentido de autocrítica e autoestima possibilitando assim uma maior e melhor compreensão conceptual.

6.2 Estudo de caso 2 – Bancos

Os dois alunos deste grupo, H_1 e H_2, efetuaram um programa a que apelidaram de Bancos no qual se pretendia saber qual o valor obtido com a aplicação de uma determinada quantia inicial num determinado período de tempo sob o efeito de juros compostos em dois bancos diferentes.

6.2.1 Os alunos

Os alunos deste estudo de caso têm rendimentos escolares ligeiramente diferentes. O aluno H_1 denota algumas dificuldades na disciplina de Matemática A, tendo decidido anular a disciplina no final do 3.º período do 12.º ano, após ter tido a classificação de 11 e 12 no 10.º e 11.º anos, respetivamente, e estar praticamente com o mesmo tipo de classificações no momento em que anulou. Já o aluno H_2 obteve no final do 12.º ano a classificação de 17 na disciplina de Matemática A, ficando com uma média dos 3 anos de 16 valores. No exame nacional de Matemática A, o aluno H_1 obteve 151 pontos e o aluno H_2 obteve 183 pontos. Na disciplina de AI obtiveram ambos 19 valores no final do 12.º ano.

O aluno H_2 consta do quadro de mérito da escola em que decorreu a experiência. Já o aluno H_1 recebeu nas diversas atas dos Conselhos de Turma de avaliação menções para o quadro de mérito da parte de todos os professores por se considerar por unanimidade que o aluno apresentou ao longo de todo o ano letivo um desempenho assinalável e digno de registo, embora as classificações nas diversas disciplinas não o denotassem de forma explicita.

6.2.2 Análise e objetivos do programa

O programa apresentado como trabalho final (Bancos, anexo V) apresenta a possibilidade de escolher dois tipos de bancos para aplicar uma determinada quantia de valores a que

os alunos apelidaram de ‘Banco Português’ e de ‘Banco Europeu’. A escolha do banco respetivo é feita através da escolha de duas opções: 1 ou 2 (figura 45).

No ‘Banco Português’, a taxa de juro anual é de 5% com uma capitalização semestral. No entanto, se o dinheiro estiver depositado mais de 3 anos, a taxa passa a ser de 4% no 4.º ano, 5% no 5.º ano e assim sucessivamente, com uma capitalização trimestral.

No ‘Banco Europeu’, a taxa de juro é de 10%, com uma capitalização anual, sendo que se o dinheiro estiver depositado mais de 4 anos, haverá mais 1 período de capitalização a cada 5 anos, isto é passados 4 anos passa a capitalização semestral; passados 9 anos a capitalização quadrimestral; e assim sucessivamente.

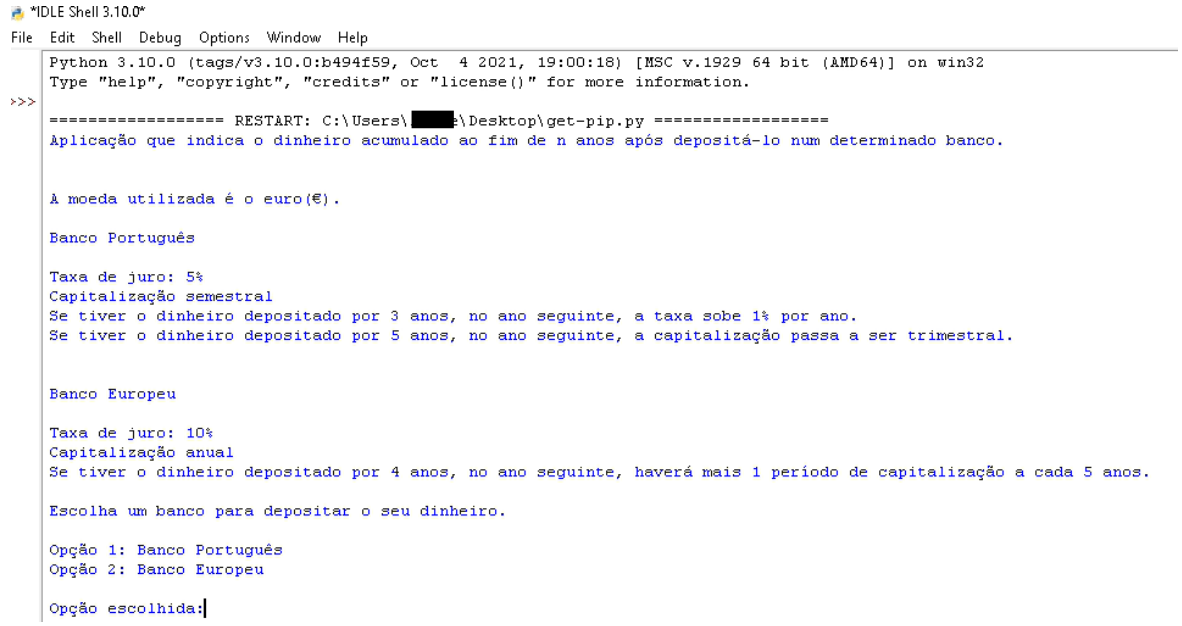


Figura 45 – Escolha das opções iniciais: 1 ou 2.

O programa permite introduzir o capital inicial a colocar no banco escolhido bem como o número de anos da aplicação. Faz igualmente um teste ao valor da quantia introduzida e ao número de anos, não permitindo que o utilizador introduza nem quantias, nem anos negativos.

Neste programa faz-se uso da fórmula $C_n(t) = c_0.\left(1 + \frac{r}{100n}\right)^{nt}$, para calcular o capital acumulado em função do capital inicial (c_0), da taxa de juro (r), do n.º de períodos de capitalização (n) e do n.º de anos (t).

Como as condições de cada banco variam conforme o n.º de anos, os alunos utilizaram uma função condicional (IF-ELIF-ELSE) para separar a função por ramos como tinham aprendido no guião 2 (anexo III), e que se encontra expresso na figura 46.

```
if t>=0 and t<=3:
   k=c0*(1.025**(2*t))
elif t>3 and t<=5:
   k=c0*((1+((2+t)/200))**(2*t))
else:
   k=c0*((1+((2+t)/400))**(4*t))
print()
print("O valor acumulado será {:.2f}€".format(k))
```

Figura 46 – Rotina do programa Bancos para saber o valor do dinheiro obtido com a capitalização no 'Banco Português'.

A expressão k=c0*(1.025**(2*t)) apresentada na figura 46, corresponde à simplificação da fórmula k=c0*((1+(5/(100*2)))**(2*t)). Por outro lado, considerando uma taxa inicial de 5% ao ano, com uma capitalização semestral, tem-se que após 3 anos, a taxa sobe 1% por ano, logo, r = 5+(t-3) = 2+t. Por último, para obter o resultado final arredondado com 2 casas decimais, escreve-se {:.2f}.

No caso do 'Banco Europeu' os alunos criaram uma variável, x, que indica o n.º de períodos de capitalização em função do n.º de anos. A Instrução criada tem a forma seguinte: 'x=(1+(t//5))'. Para este banco, se o dinheiro estiver depositado mais de 4 anos, haverá mais 1 período de capitalização a cada 5 anos, ou seja, usando a instrução '$t//5$', obtém-se a parte inteira da divisão do n.º de anos por 5, e somando 1 obtêm-se o n.º de períodos de capitalização para esse banco.

Utilizando várias simulações em que os alunos aplicavam em cada um dos bancos considerados 100 euros, durante vários anos, estes chegaram à conclusão que para um período até 7 anos, inclusive, era mais rentável colocar o dinheiro no 'Banco Europeu', o que está correto. No entanto, constataram igualmente que, para períodos de tempo superiores a 7 anos, já era mais rentável o 'Banco Português'.

O programa criado por esta equipa permite não só explorar a unidade de juros compostos que foi lecionada no 12.º ano, na disciplina de Matemática A, mas igualmente estudar extensões ao tema de forma dinâmica e útil, tomando em consideração casos concretos, e possibilitando assim o estudo da Matemática com significado e compreensão através do uso de diversas simulações.

6.2.3 Processos utilizados e fontes procuradas

Os alunos deste estudo de caso referem no questionário final que os guiões foram úteis "para obter alguma informação", nomeadamente para aprender em termos genéricos o modo de funcionamento da linguagem *Python*, tendo recorrido igualmente a pesquisas na internet.

Relativamente à utilização dos guiões consideram que deveriam ter mais exemplos para suportar os exercícios requeridos.

6.2.4 Dificuldades sentidas e raciocínio matemático envolvido

A tarefa mais fácil de executar nos guiões, segundo estes alunos, foi a relativa à utilização da fórmula resolvente para equações do 2.º grau, sendo a mais complexa a relativa à concretização do método de *Newton-Raphson*, como era expectável, uma vez que envolveu uma matéria que não tinha sido lecionada na disciplina de Matemática A, possibilitando-se assim a formação de novos conceitos matemáticos num ambiente mais informal e de forma a se conseguir integrar o PC na aprendizagem da Matemática.

A maior dificuldade centrou-se no facto de a linguagem *Python* ser nova para estes alunos o que dificultou em grande medida a concretização do trabalho. Consideram ainda que a linguagem *Python* requer uma grande concentração na sua utilização pois tem muitos detalhes ao nível da escrita simbólica.

Uma das grandes vantagens do trabalho realizado foi a possibilidade de os alunos poderem subdividir uma tarefa grande em tarefas mais diminutas, neste caso contextualizado em duas opções de escolha, sendo esta efetivamente uma das caraterísticas fundamentais da resolução de problemas matemáticos e do PC que contribui de forma decisiva para a criação de raciocínio matemático.

Os alunos deste estudo consideraram que o trabalho realizado foi útil na medida em que possibilitou que explorassem uma linguagem de programação nova, a qual lhes pode vir a ser útil no futuro. No que respeita à matemática envolvida, a abordagem utilizada foi interessante não só pelos conteúdos abordados ao longo dos guiões, mas também pelas metodologias consideradas, que como já foi referido possibilitaram a decomposição de uma tarefa em tarefas mais pequenas possibilitando assim o gosto pela Matemática.

6.2.5 Conclusões

Tal como referem os alunos nas suas considerações finais do relatório, é importante realizar este tipo de tarefas, pois permitem 'desmontar' certas matérias, de forma a entendê-las com mais compreensão. Referem o caso da utilização da fórmula resolvente em que tiveram que estudar todas as implicações do sinal do binómio discriminante e passar tal facto para a linguagem *Python,* utilizando essencialmente os raciocínios dedutivo e indutivo.

Em termos metodológicos consideram que a realização de um vídeo e de um relatório implicou uma certa sobrecarga de trabalho que consumiu muito tempo.

No que se refere ao raciocínio matemático utilizado por estes alunos existe uma predominância do raciocínio dedutivo e indutivo em detrimento do abdutivo, mais especificamente

quando criaram as diversas funções condicionais utilizadas nos diversos programas efetuados.

Em termos matemáticos, verificou-se que houve no seu trabalho de projeto um investimento maior em conceitos relacionados com os juros bancários, tema que foi objeto de estudo na disciplina de Matemática A de 12.º ano, sendo que o seu trabalho permitiu que aprofundassem e consolidassem melhor esse tema.

6.3 Estudo de caso 3 – Binómio de Newton

As duas alunas deste grupo elaboraram um programa no qual se pretendia obter o desenvolvimento de um binómio por aplicação da fórmula do Binómio de Newton.

6.3.1 As alunas

As alunas F_3 e F_4 são excelentes alunas, encontrando-se ambas no quadro de mérito da escola pelos resultados escolares alcançados ao longo do 12.º ano.

A aluna F_3, obteve no final de 12.º ano, 16 valores a Matemática A, ficando com uma média final do ensino secundário na disciplina de 16. A aluna F_4, é um pouco mais eficaz, tendo obtido 18 na disciplina e de média final. No exame nacional de Matemática A, a aluna F_3 obteve 148 pontos e a aluna F_4 obteve 190 pontos. Ambas obtiveram na disciplina de AI a classificação de 19 no final do ano letivo.

6.3.2 Análise e objetivos do programa

O programa final que foi desenvolvido por estas duas alunas (Binómio de Newton, anexo V) pretende aplicar os resultados lecionados em aula sobre a fórmula do Binómio de Newton, para o qual Álvaro de Campos, heterónimo de Pessoa (2006), referia: "O binómio de Newton é tão belo como a Vénus de Milo. O que há é pouca gente para dar por isso. óóóó – ó óóóóóóóó - ó óóóóóóóóóóóóó (O vento lá fora)." (p. 110)

Tal como as suas autoras referem no relatório apresentado, no objetivo do trabalho, pretende-se "(…) determinar o desenvolvimento do Binómio de Newton na forma $(a+b)^n$, em função de um dado valor de n".

Atendendo às regularidades observadas que permitem a representação do polinómio gerado tendo por base a potência de um binómio, as alunas, no seu relatório e no vídeo apresentado, estabelecem várias propriedades do desenvolvimento de potências de expoente inteiro não negativo.

Assim, e de modo a conseguir-se introduzir no programa, pelo utilizador, um valor para n, as alunas criaram um input para números inteiros (n=int(input(…))), em que escreveram uma mensagem que aparece no executável (linha 14, figura 47). Sabendo-se que n não pode tomar um valor negativo, criaram uma condição (n<0), isto é, se for inserido um valor para n negativo, o cálculo no desenvolvimento será interrompido. Explicam, então, que se deve dar um novo valor a n, em que n>=0 (if n<0: print("…")) (linhas 15 e 16, figura 47).

```
    def bnewton():
        msg('Binómio de Newton (a+b)^n')
        n = int(input("Escreva um valor para n: "))
15      if n < 0:
            print("Não, deve ser um número inteiro não negativo.")
        else:
            for i in range(0,n+1):
                print ('%i*a^(%i)*b^(%i)' %(binomialCoeff(n,i), n-i, i), end=" ")
20              if i != n:
                    print('+', end=" ")
```

Figura 47 – Linhas 12 a 21 do programa Binómio de Newton.

Para efetuar o cálculo do coeficiente binomial (nCk = n*(n-1) ...(n-k+1) / k*(k-1) *...*2*1) as alunas executaram um ciclo for, em que o i é iterado de 1 a k e em que para cada iteração, o resultado que foi definido antes como 1, é atualizado para resultado = resultado * (n-i+1) / i (linhas 3 a 5 da figura 48). Por exemplo, se n = 5 e k = 3 o resultado é 1. Na primeira iteração do ciclo for, o resultado é atualizado para resultado = resultado* n. Uma vez que o resultado estava definido como 1, ao fim da primeira iteração, fica como 5. Na segunda iteração é atualizado para resultado = resultado * 4/2. Ao fim da segunda iteração passa a 10. Na terceira iteração é atualizado para resultado = resultado * 3/3, ou seja, 5C3 = 10.

```
1   def binomialCoeff(n, k):
        result = 1
        for i in range(1, k+1):
            result = result * (n-i+1) / i
5       return int(result)
```

Figura 48 – Linhas 1 a 5 do programa Binómio de Newton.

Ao terminar este cálculo a função binomialCoeff irá devolver o valor do coeficiente binomial para o valor de n e k introduzidos. A partir dos valores de n e de nCp previamente calculados, calcula-se o desenvolvimento do binómio de expoente n. Criando um novo ciclo para os valores de p (representados por i) que se iniciaram em 0 e irão aumentar até que i tome valor de n. Assim, pode-se escrever: BinomialCoeff(n,i) *a^(n-i)*b^i. Que simplificado passa a: %i*a^(%i)*b^(%i), em que % corresponde, respetivamente, a binomialCoeff(n,i), n-i e i (linha 19, figura 47).

Na figura 49 encontra-se o executável do programa que se inicia com a execução da função binewton(), para $n = 23$ e a resposta dada pelo programa para $n = -2$.

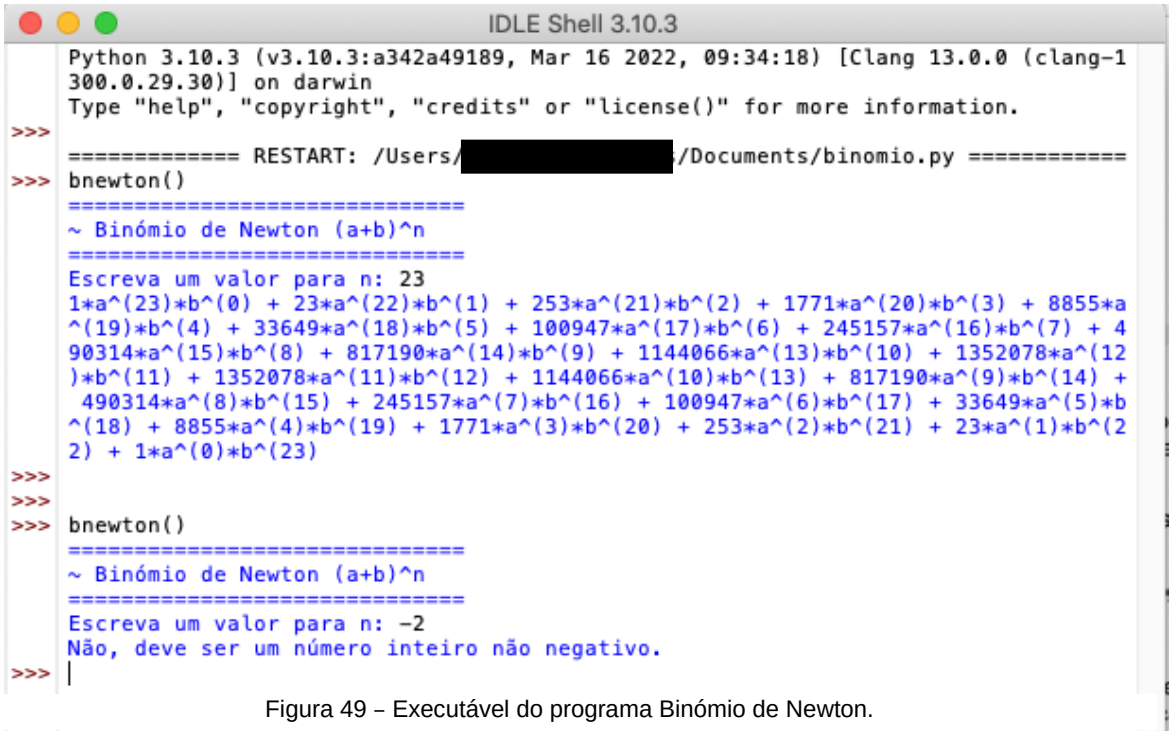


Figura 49 – Executável do programa Binómio de Newton.

Como se verifica, o programa apresentado funciona exclusivamente para variáveis '*a*' e '*b*', não podendo estas ser alteradas, o que limita um pouco a sua utilização, no entanto não apresenta falhas.

6.3.3 Processos utilizados e fontes procuradas

Para a execução deste projeto as alunas recorreram a pesquisas na internet[8], aos guiões fornecidos, a vídeos no youtube e a alguma bibliografia de referência[9].

Acharam os guiões fáceis de seguir e com explicações breves que permitiram aprender como programar em *Python*. Gostaram em especial dos exemplos de código dados para cada tarefa, com temas retirados da Matemática, por ajudarem a perceber como funciona esta linguagem de programação.

[8] https://docs.python.org/pt-br/3/reference/index.html; https://github.com/python/cpython/blob/3.10/Gramar/python.gram; https://pense-python.caravela.club/01-a-jornada-do-programa/06-linguagensformais-e-naturais.html

[9] Swokowski, E. W. (2008). *Algebra and Trigonometry with Analytic Geometry Classic 12th edition*. Books/Cole. ISBN 978-0-495-55971-9

6.3.4 Dificuldades sentidas e raciocínio matemático envolvido

As alunas destacaram como dificuldades sentidas as regras muito ‘apertadas’ da criação de ciclos como um obstáculo à prossecução do seu projeto que permitem o desenvolvimento dos raciocínios dedutivo e indutivo.

Para a resolução das tarefas propostas ao longo dos guiões recorreram, no primeiro guião, a conhecimentos relativos ao método utilizado para a escrita de uma equação reduzida da reta tendo por base dois pontos do plano, calculando o declive e a ordenada na origem. Esta foi para as alunas a tarefa mais fácil.

No segundo guião reconheceram a fórmula resolvente e, relativamente ao terceiro guião, procuraram saber como se processava o método de *Newton*. Este terceiro guião foi para as alunas o menos bem conseguido pois a aplicação do método de *Newton* deveria ter sido precedido de uma pequena explicação do que faz e de como funciona de forma a auxiliar na compreensão do que é pretendido na escrita do código.

Para estas alunas, a tarefa mais difícil foi levar à prática a tarefa final. Consideraram que este projeto as obrigou a raciocinar “fora da caixa”. O facto de terem tido que programar foi útil para desenvolver raciocínio matemático, pois "tal como uma operação matemática nos faz chegar a um resultado esperado, através de uma razão lógica e exata, também um algoritmo ou a escrita de um código permite chegar a um resultado esperado na programação". Remataram o seu raciocínio referindo que aprender *Python* as ajudou no desenvolvimento do raciocínio matemático. Efetivamente, e tal como se pode comprovar na Figura 47, existe aqui a evidência de terem utilizado os raciocínios dedutivo e indutivo ao recorrerem à formação de uma função ‘*bnewton()*’ a qual é construída tendo por base uma função condicional ‘if-elif-else’.

O trabalho efetuado foi útil uma vez que segundo as alunas as tornou mais colaborativas e motivadas, sentindo que além de efetuarem aprendizagens mais consolidadas adquiriram novas competências.

6.3.5 Conclusões

Com a concretização deste projeto, e tal como as alunas referem nas conclusões ao seu trabalho, a linguagem *Python* tem por caraterísticas principais as seguintes:

- É de fácil entendimento e aprendizagem, por ter uma sintaxe simples e amigável.
- É de altíssimo nível, por ser uma linguagem de script. A linguagem *Python* é de alto nível, ou seja, possui objetos complexos, como listas, facilitadores do desenvolvimento rápido de aplicações.

• É Orientada a objetos, fazendo um uso intensivo de objetos, como *strings*.

Por outro lado, consideram que o facto de terem abordado esta linguagem lhes será útil no futuro uma vez que podem encontrar aplicações de *Python* em sistemas de automação, administração de sistemas e redes, *hacking*, criptografia, aplicações científicas e eventualmente aplicações espaciais.

Em termos metodológicos, consideraram que ao efetuarem um relatório e um vídeo lhes foi benéfico pois estes dois recursos complementaram-se.

O raciocínio matemático aplicado por estas duas alunas foi essencialmente indutivo, embora também fosse dedutivo, como explicado anteriormente.

Em termos de avaliação do trabalho de projeto realizado considera-se que foi um trabalho muito útil inserindo-se nos conteúdos que são lecionados na disciplina de Matemática do 12.º ano de escolaridade permitindo assim o desenvolvimento de competências associadas à disciplina de Matemática.

6.4 Estudo de caso 4 – Juros

Os dois alunos deste grupo, H_3 *e* H_4, fizeram um programa a que apelidaram de Juros (Juros, anexo V) e em que ao utilizarem-no pretendiam saber qual o valor obtido com a aplicação de uma determinada quantia inicial num determinado período sob o efeito de juros compostos. Uma ideia parecida com a do estudo de caso 2, mas com várias opções de execução.

6.4.1 Os alunos

Ao contrário do estudo de caso 2, os alunos deste grupo apresentam rendimentos escolares ligeiramente superiores. O aluno H_3 obteve no final do 3.º período a classificação de 16 valores na disciplina de Matemática A, ficando com uma média final de 16 nos três anos do ensino secundário. O aluno H_4 é um aluno dedicado e muito atento na disciplina de Matemática e conseguiu em cada ano do ensino secundário a classificação de 20 na disciplina de Matemática. No exame nacional de Matemática A, o aluno H_3 obteve 166 pontos e o aluno H_4 198 pontos. Na disciplina de AI obtiveram ambos 19 valores no final do 12.º ano.

Ambos os alunos referidos constam do quadro de mérito da escola.

6.4.2 Análise e objetivos do programa

Os alunos deste estudo de caso desenvolveram uma aplicação para um banco *online* fictício, cujo programa permite efetuar 5 tipos de operações: 1) Determinar o valor do câmbio entre moedas; 2) Comparar diferentes moedas; 3) Obter o valor em euros dado o valor noutra moeda; 4) Calcular os valores a pagar por um empréstimo a prestações com um juro fixo; e, 5) Determinar o valor obtido ao fim de um determinado número de meses com o depósito de uma determinada quantia subordinada a uma taxa de juro fixa de periodicidade mensal (figura 50).

```
Olá, este é um site de um banco que te irá ajudar a realizar câmbio entre moedas
, comparar diferentes moedas e adquirir os teus produtos de sonho a taxas de jur
o imbatíveis!

ATENÇÃO- Dada a situação volátil que se tem verificado a nível global com a guer
ra na Ucrânia, algumas operações de câmbio estão inoperacionais!
Por favor, prima:

1 - Câmbio entre moedas
2 - Comparação de moedas
3 - Chega para comprar algo em euros?
4 - Pagamento a prestações
5 - Dinheiro acumulado ao fim de x meses

Escolha:
```

Figura 50 – Compilador visível após a execução do programa Juros.

A primeira coisa que pensaram é que o programa a construir teria que ter muitas linhas de código para que fosse interativo e funcional. Utilizando maioritariamente a função *print* construíram as mensagens necessárias para que o programa interagisse com o utilizador e de forma a que o algoritmo permitisse, após a escolha de uma determinada operação, que o utilizador voltasse ao início ou encerrasse o programa.

Assim, criaram uma função inicial que apelidaram de *função*, a qual, dependendo do número dado como *input*, encaminha para a operação pretendida de entre as 5 possíveis. A escolha da operação a executar pelo programa é armazenada neste por uma variável do tipo *funcn*, depois do utilizador escolher a opção e carregar em ENTER.

Segundo os alunos, a estratégia utilizada na escrita de código foi bastante intuitiva. Optaram pelo uso quase exclusivo de funções condicionais, *if, elif* e *else*, como se pode observar em Juros (anexo V), e criaram variáveis para conseguirem guardar os valores dados como resposta às perguntas e pedidos de decisão que apareceram ao executar o programa.

Nas duas primeiras opções (*func1* e *func2* são as variáveis no programa que armazenam estas duas opções), os alunos estabeleceram relações entre as diferentes moedas (utilizando os símbolos $>$, $<$ e $=$), com 5 casas decimais (à data de 3 de abril de 2022), de forma

a que ao selecionar uma moeda, se obtivesse o valor noutra arredondado às centésimas (figura 51).

```
Os valores de câmbio apresentados são relativos ao dia de hoje, com base na atua
l cotação das moedas apresentadas!
As opções de conversão de moeda são:

1 - EUR
2 - USD
3 - GBP
4 - JPY



Moeda a converter: 2

Quantia a converter: 200

Moeda para converter: 1


200.0 $ = 180.48 €

Deseja:

1 - Voltar à página inicial
2 - Sair do site


Escolha:
```

Figura 51 – Compilador visível após a execução da opção 1 do programa Juros.

Para aparecerem respostas aos comandos indicados pelos utilizadores, escreveram no programa a instrução: *a = int(input("Moeda a converter: "))*, fazendo a instrução *int* referência ao facto de o valor devolvido ter de ser um número inteiro, embora pudessem ser reais. Estes números são posteriormente arredondados com o número de casas decimais pretendido através da função *round*.

Na 3.ª opção (*func3* é a variável no programa que armazena esta opção), para permitir perceber se com um dado valor de uma moeda se consegue comprar algo com o valor de x euros, o programa começa por converter a moeda usada em euros e de seguida procede à divisão inteira do valor obtido pelo do valor do artigo. Se o resultado do quociente for inferior ao do artigo não se consegue comprá-lo, caso contrário o programa retorna o número de vezes que poderemos comprar o artigo (na figura 52 encontra-se um exemplo de escolha da 3.ª operação, para 100€, sendo que o produto custa 50 libras esterlinas).

```
1 - Câmbio entre moedas
2 - Comparação de moedas
3 - Chega para comprar algo em euros?
4 - Pagamento a prestações
5 - Dinheiro acumulado ao fim de x meses

Escolha: 3
Preço artigo desejado em euros: 100


Diga a moeda que quer usar:

1 - EUR
2 - USD
3 - GBP
4 - JPY


Escolha: 3

Quantia: 50


Infelizmente não tem dinheiro suficiente para comprar o seu artigo.
```

Figura 52 – Compilador visível após a execução da opção 3 do programa Juros.

Já na 4.ª opção (*func4*), o programa procede ao cálculo do valor pago mensalmente para a compra de um produto a prestações, dividindo o valor total a ser pago pelo número de meses e acrescentando o valor da taxa de juro fixa de 5%, que também será dividida pelo número de meses. Para isso escreveram no programa *round(j/k+j*0.05/k,2),* sendo j o preço total e *k* o número de meses. Também possibilita o cálculo do valor total a ser pago fruto de prestações idênticas a quantificar.

Por fim, na 5.ª opção (*func5*), procederam à determinação do capital acumulado ao fim de n meses através da expressão aprendida nas aulas de Matemática A, a saber:

$$C_n = C_0.\left(1+\frac{r}{100}\right)^n$$, sendo r os juros compostos mensais.

Na figura 53, encontra-se um exemplo de interface quando o utilizador escolhe esta opção para um capital inicial ,C_0 , de 1000€, durante 20 meses, com uma taxa de juro mensal de 2%.

```
Olá, este é um site de um banco que te irá ajudar a realizar câmbio entre moedas
, comparar diferentes moedas e adquirir os teus produtos de sonho a taxas de jur
o imbatíveis!

ATENÇÃO- Dada a situação volátil que se tem verificado a nível global com a guer
ra na Ucrânia, algumas operações de câmbio estão inoperacionais!
Por favor, prima:

1 - Câmbio entre moedas
2 - Comparação de moedas
3 - Chega para comprar algo em euros?
4 - Pagamento a prestações
5 - Dinheiro acumulado ao fim de x meses

Escolha: 5

Qual é o seu saldo atual? 1000

Quer saber quanto dinheiro terá daqui a quantos meses? 20

Qual é o seu juro composto mensal? 2
Daqui a 20  meses terá um saldo de 1485.95

Deseja:

1 - Voltar à página inicial
2 - Sair do site


Escolha:
```

Figura 53 – Informação visível após a execução da opção 5 do programa Juros.

Este programa, com 419 linhas de programação, é um pouco longo, e tem muitas rotinas/instruções repetidas que poderiam ser evitadas caso se tivessem utilizado funções definidas no programa que poderiam ser chamadas cada vez que fossem necessárias, além de que a fórmula das prestações constantes aparentemente está incorreta.. É verdade que o código é eficaz e não apresenta erros de aplicação, no entanto, isto constitui uma evidência que o PC nestes alunos não está ainda completamente desenvolvido pois havendo um padrão que tinha de ser identificado os alunos tinham que efetuar a abstração do que poderia ser incluído numa função *Python*.

O código idealizado e concretizado por estes alunos apresenta evidências de terem utilizado raciocínio dedutivo e indutivo, mais que não seja por permitir a escolha de uma opção de entre 5 através de uma função apelidada de *função()*, construída tendo por base funções condicionais. Estes alunos apresentam um conhecimento baseado num trabalho de abordagem exploratória em que é enfatizada “a construção de conceitos, a modelação de situações e também a utilização de definições e propriedades de objetos matemáticos” (Ponte et al., 2020).que aparentemente permitiu gerar diversos tipos de raciocínio que possibilitam a obtenção de conjeturas e de generalizações.

6.4.3 Processos utilizados e fontes procuradas

Os alunos deste grupo utilizaram a internet cada vez que os programas a executar conduziam a um erro, em particular para saber se os operadores que estavam a utilizar estavam corretos. No que respeita ao programa que tiveram de idealizar para executar o trabalho final o primeiro obstáculo que tiveram de ultrapassar foi decidir o que fazer e depois definir quais as operações que deveriam constar no banco *online*.

6.4.4 Dificuldades sentidas e raciocínio matemático envolvido

As principais dificuldades sentidas na execução do trabalho final prenderam-se com as demasiadas linhas de código que tiveram de descrever e na depuração do programa pois surgiram pequenos erros como, por exemplo, a falta de parenteses em determinadas linhas do código, sendo esta uma das práticas do PC referida por Wing (2006) e Espadeiro (2022)..

Relativamente à utilização dos guiões de trabalho na aprendizagem da linguagem *Python*, consideram que os dois primeiros se centram demasiado nos operadores lógicos e que o exercício final do 3.º guião, aplicação do método de *Newton-Raphson*, deveria ser mais fácil uma vez que por falta de tempo ou alguma inércia não o conseguiram executar. Este foi para estes alunos a exercício mais difícil, sendo que a tarefa mais fácil foi a relativa à determinação do declive e da ordenada na origem de uma reta, dadas as coordenadas de dois pontos.

Consideraram igualmente que “os guiões estavam bastante completos e tinham bastantes exercícios resolvidos que ajudavam a perceber a matéria”, como relataram na resposta ao questionário final.

Os alunos referem no questionário final que tiveram "que organizar o nosso pensamento numa sequência lógica de indicações simples a serem cumpridas pelo computador". Realmente, atendendo a esta afirmação dos alunos verifica-se que existe aqui a identificação da utilização de raciocínio dedutivo como um dos elementos fundamentais na construção dos programas solicitados nos guiões de trabalho.

Para os alunos H_3 *e* H_4, a linguagem *Python* "é claramente uma ferramenta desafiante" em que têm que optar por uma estratégia e decidir qual dos caminhos é mais eficiente. Consideram que o seu contributo para a compreensão especificamente da matéria não terá sido grande, no entanto, poderá ter ajudado na criação da organização do raciocínio e na escolha de processos mais eficientes.

Efetivamente o programa está bem estruturado embora pudesse ter sido elaborado utilizando funções as quais agilizariam os processos computacionais. No entanto, nada disso

leva a concluir que não tenha sido utilizados os diversos tipos de raciocínio, sendo que talvez o abdutivo é o menos presente pois não são apresentadas conjeturas claras às quais o programa permitisse justificar de forma clara.

Relativamente à realização deste tipo de tarefas, consideram que deveria ter sido mais dilatada no tempo. Este trabalho todo, para somente 3 aulas apenas, foi, segundo as suas próprias palavras, excessivo.

6.4.5 Conclusões

Em jeito de conclusão, os alunos referem que:

> (…) com este trabalho, mais do que termos aprendido a trabalhar com uma das linguagens de programação mais famosas e utilizadas, o Python, conseguimos perceber a utilidade que esta pode ter na realização das mais diversas atividades do nosso dia a dia. A própria prática de programar estimula tanto ou mais que a matemática a que estamos habituados graças à tentativa de criar raciocínios lógicos que "guiem" o computador na obtenção dos resultados esperados. Este trabalho mostrou-nos que fazer algo como um banco online com recurso à linguagem de programação Python é desafiante mas executável, pois sabendo as bases e buscando auxílio com o código desconhecido, é como a matemática, uma disciplina que usa normas específicas e a capacidade de raciocínio. (relato dos alunos apresentado no seu relatório)

Em relação à metodologia de trabalho sugerida em que aos alunos foi pedida a produção de um relatório e de um vídeo que fundamentasse o programa que idealizaram, estes referiram que um vídeo bastava, atendendo ao pormenor utilizado na descrição do programa.

O trabalho de projeto apresentado por estes alunos é indiciador da utilização de PC uma vez que são criadas várias funções que permitem decompor o problema inicial em cinco tipos de opções, assim como está bem patente a utilização da abstração na construção de todo o programa.

Depois, e como já foi referido, os alunos tiveram de depurar o programa pois encontraram na fase de concretização do código vários 'erros' que foram solucionando com recurso a pesquisas na internet. A linguagem *Python* serviu aqui de mote para que os alunos desenvolvessem o PC e potenciassem o seu raciocínio matemático.

6.5 Estudo de caso 5 – Triângulo de Pascal

Os dois alunos deste grupo, H_5 *e* H_6, elaboraram um programa que apelidaram de Pascal (Triângulo de Pascal, anexo V) e que devolve todos os valores de uma determinada linha do triângulo de Pascal, número de linha essa escolhida pelo operador.

6.5.1 Os alunos

Os alunos deste grupo são alunos que se podem considerar de sucesso. O aluno *H5* obteve no final do 12.º ano a classificação de 15 valores na disciplina de Matemática, ficando com uma média final de 15. O aluno *H6* é um aluno um pouco mais dedicado na disciplina de Matemática A, tendo conseguido 19 valores como média final dos três anos, tendo obtido 20 valores no 12.º ano. No exame nacional de Matemática A, o aluno *H5* obteve 151 pontos e o aluno *H6* obteve 197 pontos. Na disciplina de AI obtiveram ambos 19 valores.

Ambos os alunos referidos constam do quadro de mérito da escola.

6.5.2 Análise e objetivos do programa

O triângulo de Pascal é descrito pelos alunos no seu relatório como sendo um triângulo aritmético infinito onde são dispostos os coeficientes das expansões binomiais. Os números que compõem o triângulo apresentam diversas propriedades e relações, e são chamados de números binomiais ou coeficientes binomiais. São representados por C(n,k)=n!/k!*(n-k)!, sendo C(n,k) combinações simples de n elementos tomados k a k, com n e k números naturais (n ≥ k).

O triângulo de Pascal verifica as seguintes propriedades:

1.ª) O primeiro e o último elemento de cada linha é 1, isto é,
C(n,0)=C(n,n)=1, com n ∈ N_0.
2.ª) Elementos equidistantes dos extremos são iguais (lei da simetria), isto é,
C(n,p)=C(n,n-p), com n e p naturais e n ≥ p
3.ª) A soma de todos os elementos da linha n, com n natural, é igual a 2^n.
4.ª) Dados números naturais n e k (k<n)
C(n+1,k+1)=C(n,k)+C(n,k+1)
5.ª) A linha n do triângulo de Pascal, com n natural, tem n+1 elementos.

O programa que foi trabalhado pelos alunos tem como objetivo calcular e mostrar ao utilizador o triângulo de Pascal. Pretende-se fornecer informações possíveis de retirar do triângulo, tais como o somatório de todas as linhas anteriores e da soma de uma linha do triângulo.

Os alunos começam por definir a função *Pascal* para poderem 'correr' o programa, sendo que para dar "mais liberdade de escolha ao utilizador resolvemos pedir lhe um input de um número de uma da linha do triangulo de Pascal, para usarmos e devolver as informações fornecidas pelo nosso programa", como referem no seu relatório. Usando a função *int*()

passam o valor do *input*, que por defeito vem como uma *string*, para inteiro. Atribuíram a esse *input* a variável n, como se pode verificar na figura 54, linha 3 do programa *Pascal* elaborado,.

Pascal.py

File Edit Format Run Options Window Help

```
1  from math import *
2  def pascal():
3      n = int(input("Escreva o numero da linha de pascal: "))
4      print("\nAmostra do triângulo ate á linha " + str(n))
5      triângulo(n + 1)
6      soma = 2**(n)
7      print("\nA soma da linha " + str(n) + " é " + str(soma))
8      somatório = ((1 - 2**n)//(1 - 2)) * 1
9      print("\nA soma de todas linhas anteriores até á linha " + str(n) + " é " + str(somatório))
10
11 def triângulo(c):
12     for n in range(c):
13         for k in range(n + 1):
14             print(factorial(n)//(factorial(k)*factorial(n-k)), end = " ")
15         print()
16
17 pascal()
```

Figura 54 – Programa Pascal desenvolvido pelos alunos do estudo de caso 5.

Escreveram no corpo do programa uma função que apelidaram de *triangulo* que implica a introdução de um parâmetro que designaram por ‘c’. Na função *pascal* utilizam a função *triângulo* com parâmetro ‘n + 1’ (linha 5 do programa Pascal apresentado na figura 54), uma vez que o triangulo de Pascal começa em 0 (linha número 0). Assim, para que o programa execute o triângulo até ao número introduzido pelo utilizador executa “n+1” linhas, tal como os alunos referem no seu relatório, “por exemplo se pedirmos que o programa faça o triângulo até á linha 0 do triangulo se não tivermos o n + 1 o programa não iria fazer nenhuma linha, mas com o n + 1 já faria o total de 1 linha”.

Na definição da função *triângulo* usaram dois ciclos de repetição for: um primeiro para fazer sucessivamente as linhas do triângulo até à linha desejada e um segundo para calcular os elementos para cada posição de cada linha do triângulo (linhas 12 e 13 do programa Pascal apresentado na figura 54).

A razão pela qual os alunos utilizaram a biblioteca *math* (linha 1 do programa “Pascal” apresentado na figura 54) prende-se com o facto de o programa recorrer à função *factorial*, uma vez que para obter o valor das combinações necessita do fatorial de um número

O programa Pascal, ao ser executado, corre sucessivamente cada elemento da linha e só quando chega ao final de cada linha é que segue para a linha seguinte, percorrendo desta forma todas as linhas até ao número da linha definida pelo utilizador.

Por último o programa calcula ainda a soma de todos os elementos na linha escolhida e o somatório de todas as linhas anteriores á linha referida.

Na figura 55, encontra-se um exemplo de interface quando o utilizador introduz 8 para número da linha do Triângulo de Pascal.

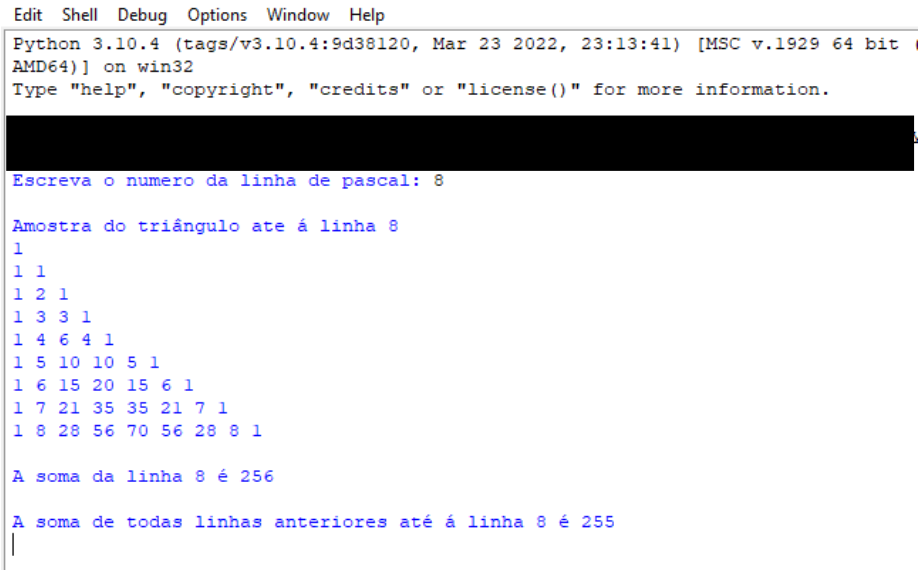


Figura 55 – Interface que se obtém introduzindo 8 no número de linha, após a execução do programa Pascal.

Este programa é um programa simples, que permite desenvolver todas as linhas até um determinado número natural escolhido pelo utilizador, e que possibilita que um aluno infira todas as propriedades do Triângulo de Pascal apresentadas na aula de Matemática A e outras que não tenham sido referidas no âmbito das Aprendizagens Essenciais em vigor.

6.5.3 Processos utilizados e fontes procuradas

Para a construção do programa final os alunos recorreram à biblioteca *math* e à internet pois, como referem no questionário final, “O trabalho final que efetuámos tinha bastantes fontes na internet as quais podemos analisar e fazer assim a tarefa final”.

Relativamente ao conhecimento que tinham de *Python* afirmaram que “a linguagem não era totalmente desconhecida, para além disso também pesquisámos na internet e no manual de matemática”.

6.5.4 Dificuldades sentidas e raciocínio matemático envolvido

Estes alunos afirmaram que não sentiram grandes dificuldades na elaboração do projeto, e segundo descrevem, conseguiram desenvolver a maior parte do programa sozinhos e "complementar depois com a ajuda da internet".

Relativamente aos guiões de trabalho propostos no trabalho de campo, e segundo o seu ponto de vista, "os guiões foram progressivamente aumentando a dificuldade, pois os exercícios pedidos eram cada vez mais exigentes". Acrescente-se que estes dois alunos não propuseram qualquer tipo de alteração aos guiões fornecidos e "acham os guiões bastante prestáveis, pois a partir deles era possível trabalhar com Python de forma simples e eficaz sem grande dificuldade".

Estes alunos consideram que a programação em *Python* é simples, no entanto, foi necessário trabalhar o raciocínio matemático de modo a resolver os exercícios propostos nos guiões fornecidos. Afirmaram, igualmente, no questionário final, que programar em *Python* quando ligado a temas intrínsecos à Matemática pode realmente potenciar este tipo de raciocínio. Esta afirmação proferida pelos alunos pode ser verificada nas linhas de código, quer pela inserção de ciclos de repetição, quer por o processo ser muito recursivo, recorrendo à definição da função *pascal*.

Segundo os alunos H_5 *e* H_6, "o trabalho desenvolvido foi útil, pois permitiu uma aprendizagem inicial que até então tínhamos trabalhado pouco, no entanto consideramos que apesar de promover o raciocínio matemático não contribuiu diretamente para uma melhor aprendizagem da matemática".

Esta última afirmação não deixa de ser estranha tanto mais que o tema escolhido por estes alunos potencia os raciocínios dedutivo e indutivo, Talvez os alunos não o tenham conseguido identificar por não terem noções claras do que é o raciocínio matemático.

Por outro lado, na construção do código apresentado foram alcançadas a abstração, embora pelos motivos já expostos anteriormente pudesse ser melhorada; a decomposição, por terem dividido a situação em estudo em partes menores; o reconhecimento de padrões, pelo reconhecimento das regularidades que constituem, em suma, as propriedades das combinações; a algoritmia, pela construção sequencial da solução, e eventualmente a depuração, para corrigir erros quer de sintaxe do *Python*, quer matemáticos.

6.5.5 Conclusões

Ambos os alunos consideram que este tipo de tarefas são importantes na disciplina de Matemática, não só como introdução para novas aprendizagens, mas igualmente para dinamizar de uma forma mais dinâmica as aulas.

Em relação à metodologia utilizada referiram que o facto de terem executado um relatório e um vídeo os incentivou a fazer um trabalho melhor "que conseguimos" e, por via disso, analisar e compreender melhor a linguagem de programação *Python*.

Estes alunos conseguiram construir um programa com apenas 15 linhas de programação e perfeitamente depurado, assente num tema que é lecionado no 12.º ano, cálculo combinatório. É, no entanto, um programa em que o uso da função fatorial para gerar todo o triângulo de Pascal é algo ineficiente em virtude de conter demasiadas operações para o resultado pretendido. O uso da 4.ª relação permitiria com poucas adições gerar ~mais facilmente o triângulo.

O programa, pela forma como está construído, permitiu desenvolveu nos alunos capacidades de programação em *Python* e possibilitou o tratamento do tema escolhido com compreensão, além de potenciar a utilização de raciocínio matemático nas suas diversas vertentes, mostrando a integração do PC na Matemática.

6.6 Estudo de caso 6 – Funções_Cálculo

Os alunos deste grupo misto, F_5 e H_7, efetuaram um programa a que apelidaram de Funçãoanálise (Funções_Cálculo, anexo V) no qual pretendem obter a caraterização de uma função polinomial de grau inferior ao 3.º, em termos de domínio; contradomínio; zeros; estudo do sinal, intervalos de monotonia; estudo do sentido das concavidades do seu gráfico; coordenadas do vértice, se aplicável, e uma expressão da derivada.

6.6.1 Os alunos

Este grupo é constituído por dois excelentes alunos, ambos do quadro de mérito da escola, sendo que o aluno H_7 obteve no final do 12.º ano a classificação de 20 valores na disciplina de Matemática, ficando com uma média final de 19 e a aluna F_5 obteve 16 valores, ficando com uma média final de 17. No exame nacional de Matemática A, a aluna F_5 obteve 198 pontos e o aluno H_7 obteve 199 pontos. Ambos obtiveram 19 valores na disciplina de AI

6.6.2 Análise e objetivos do programa

O trabalho efetuado por estes dois alunos, tal como é referido no seu relatório:

> (..) tem como objetivo consolidar o que foi aprendido em aula no que convém a conhecimentos de Python de forma a conciliar a matemática com a programação e também para entender e poder aplicar esta linguagem de programação, que certamente será útil para o nosso futuro, dada a modernização do trabalho e do mundo em geral. (relatório apresentado pelos alunos F_5 e H_7)

Estes alunos começaram por contextualizar a linguagem *Python* onde referem a propósito que: “o Python é uma linguagem de programação de alto nível orientada a objetos criada

em 1991 e que recentemente tem ganhado muita popularidade nos campos da ciência e das engenharias devido à sua eficiência na resolução de variados problemas e por ser uma linguagem clara e concisa", acrescentando que "o nosso projeto procedeu sem grandes dificuldades, dado que quando havia um erro, facilmente identificávamo-lo e corrigíamos".

Desenvolvendo o programa pretendido no Python IDLE, e mantendo os objetivos iniciais os alunos, segundo referem, ainda ponderaram incluir a representação gráfica da função, no entanto não o fizeram pois não conseguiram ser bem-sucedidos na instalação do módulo gráfico necessário para obter tal desígnio.

Partiram o código em várias partes, isto é, primeiro introduziram no programa uma função que permitisse ao utilizador introduzir os diferentes coeficientes da expressão do 1.º ou do 2.º grau, o que conduziu, respetivamente, a uma função afim ou quadrática, tal como consta da figura 56.

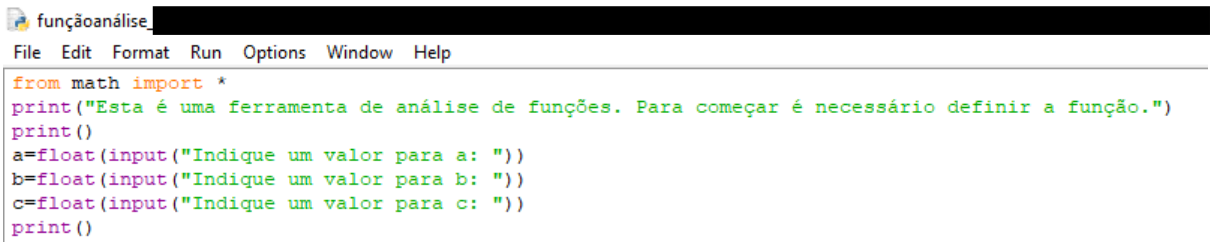


Figura 56 – As primeiras linhas do programa Funçãoanálise que permitem a introdução dos coeficientes das funções afim ou quadrática pelo utilizador.

Tendo por base os coeficientes introduzidos, e efetuando sucessivas verificações, utilizando para tal várias funções condicionais (*if-elif-else*), tal como consta no código apresentado em Funções_Cálculo (anexo V), o programa possibilita que surjam no écran diverso tipo de informações sobre a função inserida como, por exemplo, os intervalos de monotonia, o estudo do sinal e a expressão da derivada da função.

Na figura 57, encontra-se um exemplo de interface para a função definida pela expressão $y = -x + 2$

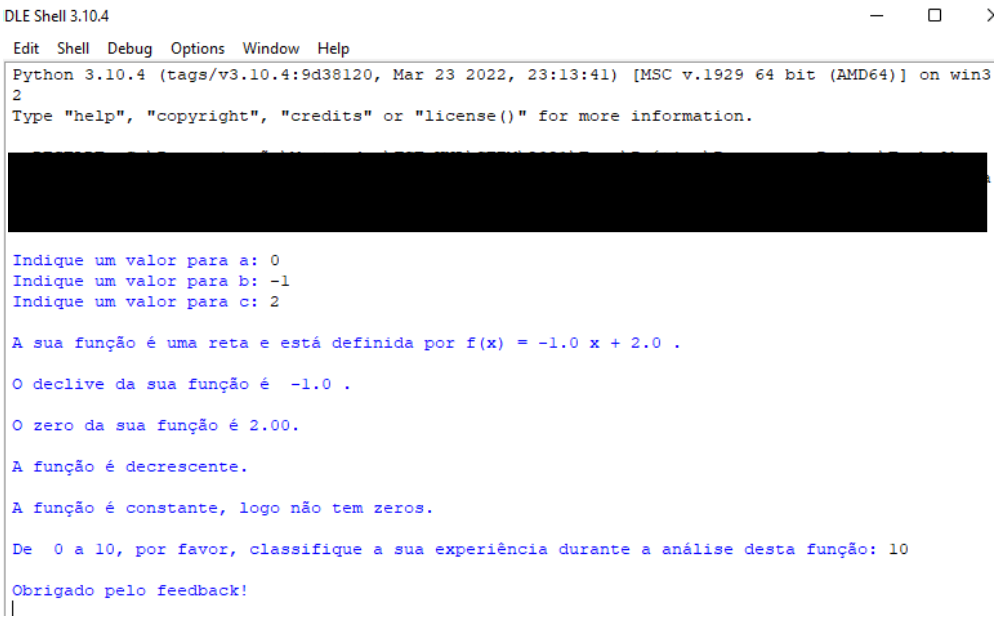


Figura 57 – Interface que se obtém depois de introduzir sucessivamente os coeficientes da função do 1.º grau, e após a execução do programa Funçãoanálise.

e na figura 58 para a função definida por $y = 12x^2 + 3{,}6x - 24{,}2$

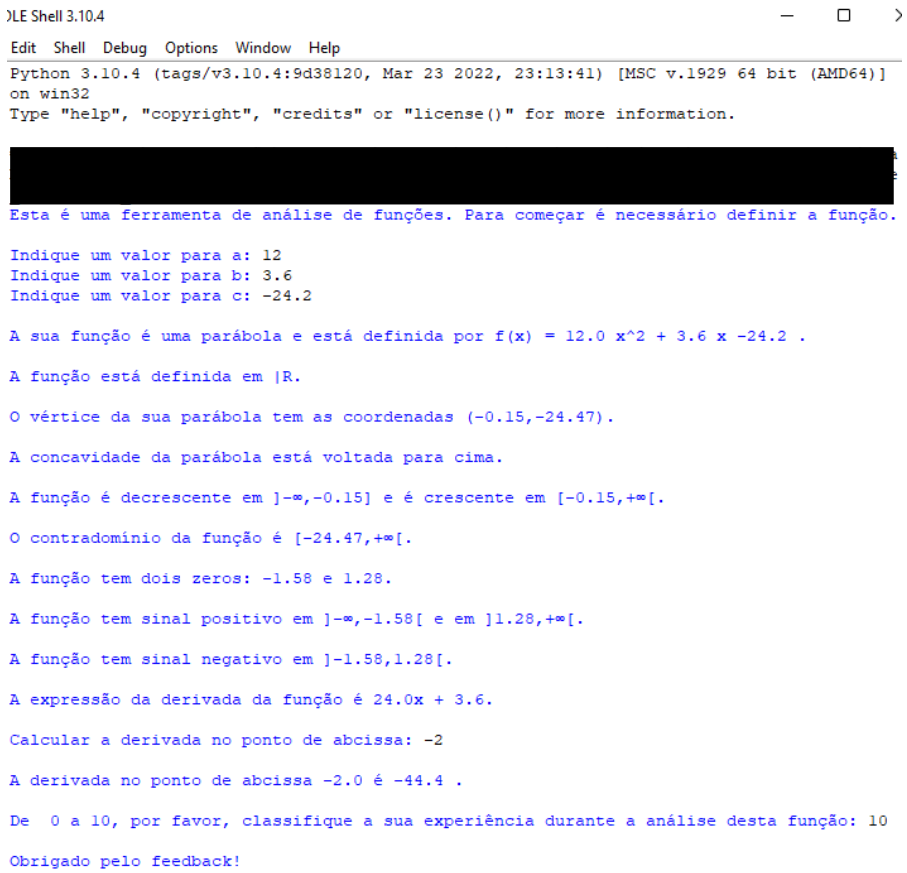


Figura 58 – Interface que se obtém depois de introduzir sucessivamente os coeficientes de uma função do 2.º grau, e após a execução do programa Funçãoanálise.

Este programa, com 121 linhas de programação, embora pareça algo básico, permite efetuar vários tipos de testes depois do utilizador inserir os diversos coeficientes da função a estudar, quer esta função seja do 1.º grau ou do 2.º, tendo sido integralmente construído pelos alunos. Esta conclusão foi obtida não só pela apresentação efetuada pelos alunos no relatório e no vídeo apresentados, mas também pelas perguntas que os alunos foram efetuando no decurso da sua construção.

Talvez a sua apresentação fosse melhorada se fossem utilizadas quer uma biblioteca gráfica, quer uma biblioteca de cálculo algébrico simbólico, no entanto está funcional e não apresenta erros de execução, sendo que faz tudo aquilo que os seus autores pretendiam que executasse.

6.6.3 Processos utilizados e fontes procuradas

Estes dois alunos decidiram fazer no seu projeto uma extensão do exercício final do guião 2 e consideraram, relativamente aos guiões, que "foram úteis, especialmente com os exemplos, que deram para compreender melhor as aplicações de cada componente do Python". Não propuseram qualquer alteração aos guiões, considerando-os “elucidativos”.

Recorreram essencialmente aos "conhecimentos adquiridos através dos guiões e das aulas práticas".

6.6.4 Dificuldades sentidas e raciocínio matemático envolvido

O maior problema que os alunos enfrentaram foi a escolha do tema para o trabalho final, sendo que decidiram "fazer uma extensão/desenvolvimento do exercício final do guião 2", como referem na sua resposta ao questionário final.

Em termos das tarefas solicitadas nos guiões consideram que a mais fácil foi a do exercício final do guião 1, pois segundo eles era a que envolvia menos conhecimentos de *Python* e servia como se fosse uma introdução. A tarefa que consideraram mais complexa foi a final pois “exigiu a junção de várias matérias aprendidas ao longo dos guiões, para além de um vídeo e um relatório”. Sugeriram que os guiões fossem impressos com frente e verso, tendo-os considerado “elucidativos”.

Para estes alunos, o facto de terem aprendido *Python* ajudou-os a pensar "mais objetivamente e com lógica". Referem igualmente que:

> (…) tivemos de aplicar várias áreas de matemática de forma a fazer o problema [final], como por exemplo o uso das derivadas para avaliar a monotonia de uma função e as concavidades do gráfico de uma função. (retirado do questionário final dos alunos F_5 e H_7)

O programa efetuado por F_5 e H_7 efetivamente obrigou ao reconhecimento de padrões e a raciocinar de forma lógica, potenciando assim o raciocínio dedutivo, o que aparentemente foi benéfico como é reportado por estes. Esta caraterística do raciocínio matemático é visível quando os alunos construíram as diversas funções condicionais utilizadas no código construído.

Para os alunos envolvidos neste projeto a resolução da tarefa proposta foi útil pois aplicaram "conhecimentos matemáticos, consolidando-os". Acharam que é igualmente importante para a disciplina de Matemática “porque nestas tarefas pegamos na matéria que damos na disciplina e aplicamos numa coisa que terá um uso útil para o futuro da nossa vida, como é o uso da programação em Python”.

6.6.5 Conclusões

Em jeito de conclusão, referiram no relatório apresentado que conseguiram atingir os objetivos definidos inicialmente, uma vez que efetuaram a caraterização completa de uma função quadrática/afim através de um projeto em linguagem *Python*. Entendem que, com este trabalho, ficaram mais familiarizados com as situações que aparecem no desenvolvimento de um projeto de programação, tanto as partes em que se desenvolve facilmente o código, como atender aos obstáculos que surgem na sua construção.

Por último, e em termos metodológicos, consideram que bastaria efetuar somente um vídeo ou um relatório para conseguirem caraterizar o seu projeto.

Da análise do programa elaborado por estes alunos constata-se que utilizaram essencialmente o raciocínio dedutivo, tendo igualmente utilizado as 5 práticas do PC.

6.7 Estudo de caso 7 – Média do secundário

As duas alunas deste estudo de caso, F_6 e F_7, efetuaram um programa (Média do secundário, Anexo V), a que apelidaram de Médiadosecundário, o qual devolve a média de um dado aluno no secundário inserindo-se para tal as classificações obtidas por esse aluno em cada uma das disciplinas frequentadas em cada ano de escolaridade.

6.7.1 As alunas

As alunas deste grupo têm um rendimento escolar que se pode apelidar de sucesso figurando ambas no quadro de mérito da escola. A aluna *F6* obteve no final do 12.º ano a classificação de 16 valores na disciplina de Matemática A, ficando com uma média final de 16. A aluna *F7* conseguiu um pouco mais de sucesso na disciplina de Matemática, tendo conseguido 17 valores, sendo a média final dos três anos de 17 valores. No exame nacional de Matemática A, a aluna *F6* obteve 183 pontos e a aluna *F7* obteve 164 pontos. Na disciplina de AI obtiveram ambas 19 valores.

6.7.2 Análise e objetivos do programa

Estas duas alunas efetuaram um "código que (…) pretende simplificar o cálculo da média do secundário", como referem no seu relatório, e desse modo ajudar todos os alunos.

Criaram, logo no início do programa uma função que apelidaram de *função* em que o parâmetro a introduzir aloca na variável 'a' o curso lecionado, ou '1' ou '2', conforme o curso seja, respetivamente, um curso científico humanístico, ou um curso profissional (a opção relativa ao curso profissional não se encontrava realizada no momento de entrega do trabalho) (figura 59).

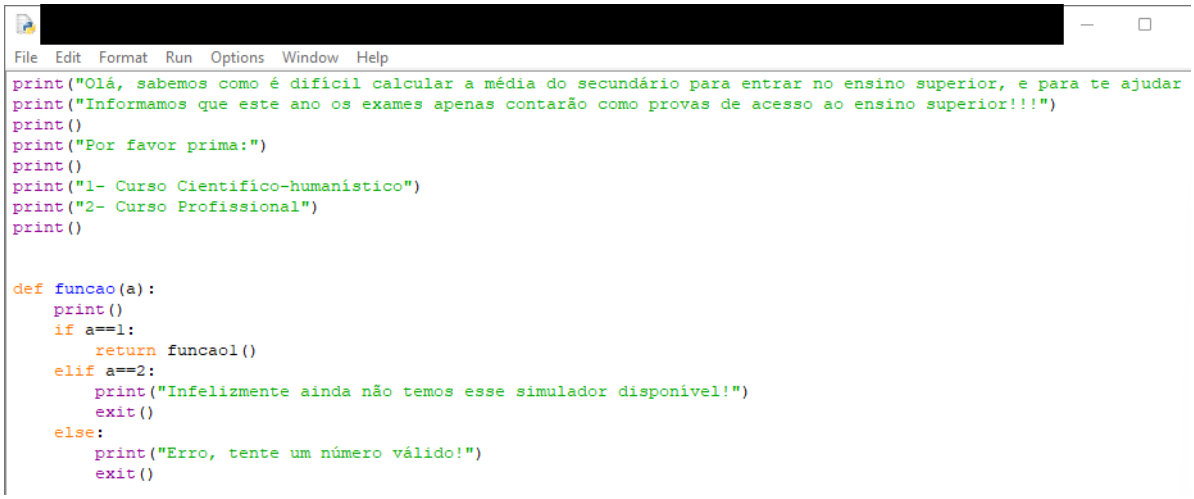


Figura 59 – Linhas iniciais de código do programa Médiadosecundário.

Em seguida, definem uma *função1* associada ao curso científico e a partir daí o programa pergunta sucessivamente a média do curso pretendido (variável 'b') e peso e as classificações obtidas nos Exames Nacionais do Ensino Secundário, precedida do número de disciplinas que o curso pretendido utiliza como provas de ingresso.(variável 'c').

Como as alunas referem no seu relatório, “Estes valores não podem ser menores que 0 ou maiores que 20, contudo cursos com média inferior a 10 são bastante raros. O número de exames não pode exceder os 3 e o seu peso está entre 35 e 50 por cento”. Na posse destes valores, o programa calcula uma média simples ((m+n+o)/3), tendo por base as variáveis relacionadas com cada valor em que ‘m’, ‘n’ e ‘o’ são as variáveis que definem as notas obtidas nos exames. Todas estas introduções são alvo de testes com funções condicionais para detetar se os valores introduzidos pelo utilizador fazem sentido (figura 60).

```
funcao1():
b=round(float(input("Qual é a média do curso que pretendes seguir? ")),2)
print()
if b>20 or b<0:
    print("Esse valor não é válido")
    exit()
elif b>0 and b<10:
    print("Duvido que existam cursos com médias tão fracas!")
print()
c=int(input("Quantos exames vais usar como provas de ingresso? "))
print()
if c>3 or c <0:
    print("Só podes usar até 3 exames!")
elif c==0:
    print("Tens de usar pelo menos um exame para qualquer curso!")
else:
    d=int(input("Qual é o peso total dos exames como provas de ingresso, em percentagem: "))
    print()
    if d>50 or d<35:
        print("Esse valor não é válido!")
        exit()
    else:
        if c==1:
            med=float(input("Quanto teve no seu exame?"))
        if c==2:
            f=int(input("Qual é o peso do seu primeiro exame? "))
            print()
            g=int(input("Qual é o peso do seu segundo exame? "))
            print()
            h=float(input("Quanto teve no primeiro exame?"))
            print()
            i=float(input("Quanto teve no segundo exame?"))
            med=round(((h+i)/2),1)
            if f+g!=d:
                  print("A soma dos seus exames tem que corresponder ao valor acima indicado!")
                  exit()
        if c==3:
            j=int(input("Qual é o peso do seu primeiro exame?"))
            print()
            k=int(input("Qual é o peso do seu segundo exame?"))
            print()
            l=int(input("Qual é o peso do seu terceiro exame?"))
            print()
            m=float(input("Quanto teve no primeiro exame?"))
            print()
            n=float(input("Quanto teve no segundo exame?"))
            print()
            o=float(input("Quanto teve no terceiro exame?"))
            med=round(((m+n+o)/3),1)

            if j+k+l!=d:
                  print("A soma dos seus exames tem que corresponder ao valor acima indicado!")
                  exit()
    print()
    print("Agora que já falámos de exames prepara-te para fazer a média do secundário!")
```

Figura 60 – Linhas de código do programa Médiadosecundário que servem para solicitar ao utilizador os resultados obtidos nos Exames Nacionais do Ensino Secundário

Após a introdução destas informações é necessário introduzir as notas obtidas em todas as disciplinas desde o 10.º até ao 12.º ano. Com todos estes valores declarados o programa calcula a média final do secundário, tendo por base a fórmula seguinte:

$$finalfinal = round((med * 0.01 * d + finalnota * 0.01 * (100 - d)),2)$$

em que '*d*' é o peso dos exames e '*med*', a média dos mesmos, como consta do programa que se encontra em Média do secundário (Anexo V).

Caso esse valor seja inferior ou superior à média do curso pretendido pelo aluno, assim o mesmo irá receber uma mensagem no écran que "talvez precise de estudar um pouquinho mais" ou que "está de parabéns", conforme o caso. Na figura 61 encontra-se a apresentação do menu inicial do programa Médiadosecundário.

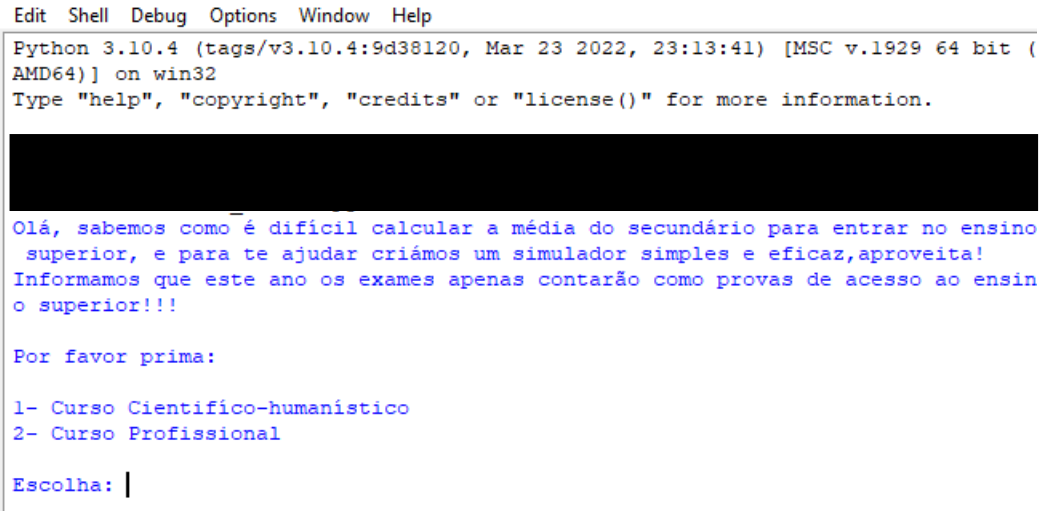


Figura 61 – Menu inicial do programa Médiadosecundário

Na figura 62 está um exemplo da mensagem final dada pelo programa Médiadosecundário, após a introdução de todos os valores solicitados.

```
 A sua nota de secundário de  13.8  com a dos seus exames de 17.5  dá uma média
final de  15.28
Parabéns, tens média para entrar no curso que queres!
```

Figura 62 – Exemplo de mensagem final emitida pelo programa Médiadosecundário

Este programa tem 287 linhas de programação e funciona essencialmente com as instruções *input* e *print* e à base de expressões condicionais *if-elif-else*. É um programa que não se baseia obviamente em conteúdos do programa de Matemática A, mas que permite

potenciar o raciocínio matemático e preenche uma das necessidades dos alunos neste ano de escolaridade para este nível de ensino.

Este programa teria ficado mais simples se as alunas envolvidas na sua produção tivessem criado uma função específica para obter as classificações das disciplinas em cada ano de escolaridade, com a indicação se a disciplina era anual, bienal ou trianual, não havendo assim a necessidade de repetir no programa várias linhas de programação idênticas. Isto permite concluir que o PC nestas alunas não está completamente desenvolvido pois falharam na identificação de padrões combinada com a abstração de forma a simplificar o programa.

No entanto, e atendendo ao que foi lecionado nas aulas de *Python*, está um programa eficiente, apesar de não estar terminado com a opção para o ensino profissional.

6.7.3 Processos utilizados e fontes procuradas

As alunas seguiram "cuidadosamente os passos, contudo tivemos de pesquisar informações no Google no trabalho da fórmula resolvente". Sempre tiveram presente que iriam efetuar um programa sobre médias, atendendo a que é um problema que todos os alunos no secundário se deparam e efetuaram alguma pesquisa para conseguirem fazer a aplicação.

Consideram o programa realizado bastante simples, embora a dificuldade de execução esteja na declaração de todas as variáveis e na atribuição do seu peso.

6.7.4 Dificuldades sentidas e raciocínio matemático envolvido

Acharam os guiões produzidos para esta investigação fáceis de entender e consideram que não sabem ainda o suficiente de *Python* para sugerir alterações aos guiões.

As tarefas mais fáceis de executar nos guiões são as iniciais, enquanto que "a mais difícil é o último", não se conseguindo perceber se se referem ao exercício do 3.º guião, se ao trabalho final. Supõe-se que se referem ao exercício de aplicação do método de *Newton-Raphson*.

Estas alunas consideram que o facto de terem feito este projeto não lhes melhorou em nada o raciocínio, o que permite afirmar que talvez as alunas não saibam devidamente em que consiste o raciocínio matemático pois utilizaram-no ao longo da construção do código.

Entendem que o facto de perceber pouco ou nada de *Python*, a matemática complica. Talvez por essa razão a escolha do tema para o trabalho final pouco tenha em comum com os temas abordados na disciplina de Matemática A.

Para as alunas o trabalho efetuado foi útil pois "agora conhecemos a linguagem um pouco melhor". Ainda segundo estas alunas a realização de tarefas deste tipo, sobre a forma de trabalho de projeto, “ajuda no nosso futuro profissional e torna as aulas mais interativas”.

6.7.5 Conclusões

Com o desenvolvimento do trabalho efetuado,

> Pensamos ter atingindo os objetivos propostos pelas disciplinas quer de Matemática quer de Aplicações Informáticas, ganhando não só conhecimento de programação de uma linguagem que desconhecíamos, mas amplamente usada quer academicamente quer nas empresas, como compreendendo o papel da Matemática na programação quer na lógica que a suporta, quer nas operações e funções a que recorremos. (extraído do relatório apresentado pelas alunas)

Em termos metodológicos, consideram que o facto de terem efetuado um vídeo e um relatório "apenas nos deu trabalho". Também aqui é possível discordar, sendo que as alunas apresentaram um vídeo bastante estimulante e bem conseguido, com uma linguagem bastante simples e acessível, própria para alunos da sua idade.

No relatório efetuaram uma descrição simples do programa e da sintaxe utilizada, enquanto que no vídeo apresentaram exemplos práticos de aplicação do programa, bastante dinâmicos e ‘levando’ qualquer pessoa que assistisse ao vídeo a experimentá-lo.

Verifica-se que o trabalho apresentado por estas duas alunas integra o PC na Matemática, embora com as observações já efetuadas, permitindo trabalhar um tema concreto de forma abstrata, que possibilita o tratamento de casos gerais, sendo que na construção do programa em *Python* as alunas tiveram de conceber um algoritmo lógico e sequencial, potenciando assim o raciocínio matemático.

Como nota final, e apesar do programa estar funcional, este ficaria mais completo se apresentasse igualmente o código para cálculo da média de acesso para alunos do ensino profissional.

6.8 Estudo de caso 8 – Método de Newton-Raphson

Os dois alunos deste grupo, F_8 e H_8, elaboraram um programa que apelidaram de MétodogeométricoNewton (Método de Newton-Raphson, Anexo V), em que aprofundaram sob o ponto de vista gráfico o Método de *Newton-Raphson*, abordado na última sessão de introdução à linguagem *Python*.

6.8.1 Os alunos

Os alunos deste grupo misto têm rendimentos escolares um pouco diferentes, embora ambos pertençam ao quadro de mérito da escola. A aluna *F8* obteve no final do 12.º ano a classificação de 20 valores na disciplina de Matemática A, ficando com uma média final de 20. O aluno *H8* conseguiu um pouco menos na disciplina de Matemática A, tendo conseguido 13 valores, sendo a média final dos três anos de igual valor. No exame nacional de Matemática A, a aluna *F8* obteve 184 pontos e o aluno *H8* obteve 140 pontos. Na disciplina de AI obtiveram ambos 19 valores.

6.8.2 Análise e objetivos do programa

O projeto desenvolvido por estes dois alunos teve como objetivo entender "de que forma a programação, especificamente em linguagem python, pode ser utilizada na aprendizagem da matemática e quais os benefícios do seu uso". Tendo este propósito em mente, decidiram explorar as capacidades gráficas oferecidas por esta linguagem, "pondo-as ao serviço da aprendizagem e da melhor compreensão de um algoritmo previamente estudado", segundo referem no seu relatório.

Assim, pretenderam atingir uma compreensão mais profunda do método de *Newton-Raphson* para o cálculo dos zeros de uma função, através da visualização gráfica das sucessivas iterações deste algoritmo.

Como referem, "em análise numérica, o método de Newton (ou Método de Newton–Raphson), desenvolvido por Isaac Newton e Joseph Raphson, tem como objetivo estimar as raízes de uma função". No programa, começaram por escolher uma aproximação inicial para a raiz e em seguida calcularam a equação reduzida da reta tangente ao gráfico da função para o ponto com a abcissa nesse valor, usando a derivada.

Determinaram a abcissa do ponto de interseção da reta com o eixo das abcissas, tendo por objetivo encontrar uma melhor aproximação para a raiz, e ao repetir sucessivamente o processo, criaram um método iterativo para encontrar o zero da função. Em notação matemática, o método de *Newton* é dado pela expressão expressa na figura 63.

$$x_{n+1} = x_n - \frac{f(x_n)}{f'(x_n)}, n \in \mathbb{N}$$

Figura 63 – Fórmula de aplicação do método de *Newton-Raphson*.

onde o valor inicial, x_0, é uma aproximação inicial dada, n indica a n-ésima iteração do algoritmo e $f'(x_n)$ é a derivada da função f no ponto de abcissa x_n.

Tendo como problema inicial o objetivo de determinar o zero de uma dada função f, conforme a figura 64 ilustra, pretende-se calcular x_1 em função de x_o, sabendo que x_1 é dado pela abcissa do ponto de intercessão da reta tangente ao gráfico em x_0 com o eixo das abcissas.

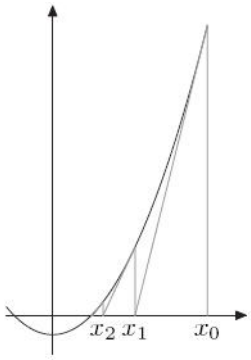


Figura 64 – Duas primeiras iterações por aplicação do método de *Newton-Raphson*

Uma equação da reta tangente ao gráfico da função f no ponto de abcissa x_0 tem declive $m = f'(x_0)$ e é dada por: $y - f(x_0) = f'(x_0).(x - x_0)$. Substituindo na equação da reta o ponto de coordenadas $(x_1, 0)$, obtém-se: $0 - f(x_0) = f'(x_0).(x_1 - x_0)$.

Assim sendo, tem-se que: $x_1 = x_0 - \frac{f(x_0)}{f'(x_0)}$.

A aplicação efetuada pelos alunos tem como objetivo visualizar as sucessivas iterações obtidas por aplicação do método de *Newton-Raphson* numa determinada função. Na figura 65 encontra-se o ciclo de repetição *WHILE* que define a forma como o método de *Newton-Raphson* foi aplicado à função definida no programa.

```
#ciclo do método de newton-raphson
while np.abs(x1-x0)>0.01:
    a0=df.subs(x,x0) #f'(x0)
    y0=f.subs(x,x0)  #f(x0)
    reta=y0+a0*(x-x0) #equação da reta tangente

    lreta = np.array([reta.subs(x,u) for u in lx],dtype='float') #lista com os pontos do gráfico da reta tangente

    plt.plot(lx,lreta, color='blue') #formatação da reta tangente
    x1=x0
    x0=x0-y0/a0 #substituição do valor x0 pela interseção da tangente com o eixo das abcissas
    plt.plot(x0,0, marker="o", markersize=20, markerfacecolor='magenta') #formatação dos pontos de interseção
    label="{:.0f}".format(i)
    plt.text(x0,-0.1, label, fontsize=55, color='magenta')
    i+=1
```

Figura 65 – Linhas de código com o método de *Newton-Raphson*

Note-se que a função é definida diretamente no código, podendo ser alterada para estudar qualquer função (figura 66), sendo que a função derivada é obtida recorrendo à biblioteca *Sympy*.

```
#função e função derivada
f = x**3+1
df=sm.diff(f,x)
```

Figura 66 – Função e função derivada que podem ser inseridas no código pelo utilizador.

O intervalo para a deteção do zero e o valor da aproximação inicial, x_0, são definidos pelo utilizador, diretamente na consola (figura 67).

```
i=1
#intervalo de detção do zero e janela de visualização
a=float(input("Inserir limite mínimo para o intervalo de deteção do zero:"))
b=float(input("Inserir limite máximo para o intervalo de deteção do zero:"))
h1=float(input("Inserir limite inferior do gráfico:"))
h2=float(input("Inserir limite superior do gráfico:"))
#valor da aproximação inicial
x0=float(input("Inserir o valor da aproximação inicial:"))
x1=x0+1
#lista com os pontos do gráfico da função
lx = np.linspace(a,b,500)
ly = np.array([f.subs(x,v) for v in lx],dtype='float')
#formatação do gráfico
plt.figure(figsize=(40,40))
plt.plot(lx,ly, color='red', linewidth=6)
plt.axhline(0, color='black', linewidth=3)
plt.axvline(0, color='black', linewidth=3)
plt.title('Método de Newton-Raphson', fontsize=85)
plt.xlim(a,b)
plt.ylim(h1,h2)
```

Figura 67 – Linhas de código para solicitar ao utilizador o intervalo para a deteção do zero e o valor da aproximação inicial, bem como para definir a janela de visualização do gráfico e os pontos do gráfico da função.

Após o utilizador definir todos os parâmetros necessários (a figura 68 apresenta um exemplo de parâmetros introduzidos), o programa traça o gráfico da função (representado a vermelho), bem como as retas tangentes ao gráfico nos sucessivos pontos de abcissa *x* (representadas a azul) (figura 69).

```
Inserir limite mínimo para o intervalo de deteção do zero:-2
Inserir limite máximo para o intervalo de deteção do zero:1.5
Inserir limite inferior do gráfico:-1.5
Inserir limite superior do gráfico:2
Inserir o valor da aproximação inicial:-0.6
```

Figura 68 – Parâmetros inseridos no programa MétodogeométricoNewton.

No gráfico são também assinalados os pontos de interseção das retas tangentes com o eixo das abcissas, identificados pelo número da iteração correspondente. Quando o algoritmo consegue encontrar uma aproximação ao zero com um intervalo de erro inferior ao máximo

definido (0,01, no caso analisado), o algoritmo é interrompido e o programa indica na consola o valor aproximado da raiz encontrada, bem como o número de iterações necessárias para alcançar este valor (figura 70).

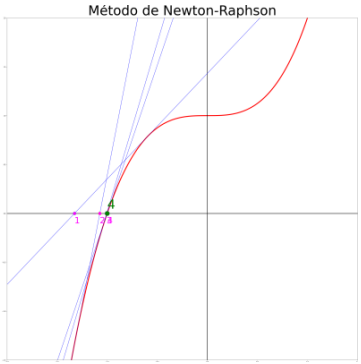


Figura 69 – Gráfico da função e as retas tangentes ao gráfico nos sucessivos pontos de abcissa *x*

```
Inserir o valor da aproximação inicial:-0.6
A função tem um zero aproximadamente em -1.000024088485816.
O método de Newton aproximou-se do zero ao fim de 4 iterações.
```

Figura 70 – Valor aproximado da raiz encontrada e o número de iterações necessárias para alcançar esse valor

O programa desenvolvido por estes dois alunos, como já foi referido, resultou de um apuramento do programa apresentado no guião 3, em que os alunos introduziram uma representação gráfica da função inserida no código e das retas tangentes ao gráfico dessa função.

É um programa com um grau de elaboração algo complexo mas muito bem estruturado, em que os seus autores tiveram uma preocupação acentuada em explicar no próprio código as sucessivas linhas introduzidas no programa, o que constitui um bom indicador e demonstra que os seus autores compreenderam o método utilizado. Por outro lado, além de utilizarem um aplicativo de suporte diferente dos colegas dos outros estudos de caso (*Spyder* (*Python* 3.9)), recorreram a três bibliotecas (*Numpy*, *Sympy* e *Matplotlib*) que não foram referidas ao longo das três aulas de introdução ao *Python*, o que revela por parte dos alunos vontade em pesquisar e experimentar coisas novas, sendo que no vídeo apresentado demonstraram ter perfeito domínio sobre o programa realizado.

6.8.3 Processos utilizados e fontes procuradas

Os alunos tiveram a ideia de efetuar uma representação gráfica do método de *Newton* durante a realização do exercício final proposto no ´3.º guião. De forma a conseguirem efetuar o programa que idealizaram, efetuaram pesquisas sobre o método que utilizaram para construir o código e sobre mais funcionalidades do *Python*.

Tal como os alunos referem no seu relatório, para o funcionamento desta aplicação recorrem a três bibliotecas *open source,* a saber:

Numpy - uma biblioteca que oferece um vasto número de funções matemáticas e ferramentas para trabalho com *arrays*;
Sympy - uma biblioteca direcionada para a matemática simbólica;
Matplotlib - uma biblioteca direcionada para a criação de visualizações gráficas na consola.

Estas três bibliotecas permitiram determinar a derivada da função que se encontrava inserida no código e visualizar, quer a representação gráfica da função, quer as sucessivas retas tangentes ao gráfico da função.

6.8.4 Dificuldades sentidas e raciocínio matemático envolvido

As dificuldades sentidas por estes alunos para levar à prática o seu projeto centraram-se nas "ferramentas de representação gráfica do python", pela sua novidade.

Relativamente aos guiões consideraram que "estavam muito bem estruturados", tendo sido "bastante úteis", nomeadamente por explicarem o código e depois mostrarem em pseudocódigo e em *Python* o que se pretendia, permitindo desta forma uma melhor compreensão da linguagem. Referiram ainda que talvez fosse interessante haver mais exercícios sem a resposta escrita, de forma a que os alunos pudessem praticar mais.

Os conhecimentos mobilizados basearam-se essencialmente em conceitos matemáticos sobre funções, adquiridos nas aulas de Matemática. Por outro lado, necessitaram igualmente de obter novos conhecimentos, nomeadamente sobre programação, no que respeita à necessidade de compreender melhor a sintaxe e o funcionamento de um programa.

Consideraram ainda que a tarefa mais fácil que executaram foi a do guião 1, por requerer conhecimentos básicos, e que a mais complexa foi a do projeto final, que exigiu um programa com mais componentes.

Como referem os alunos no questionário final, "o problema que resolvemos obrigou-nos a compreender aprofundadamente o método de Newton, de forma a podermos analisá-lo e traduzi-lo para uma representação gráfica".

Consideraram que, no geral, o facto de programarem em *Python* é "excelente" para desenvolver o raciocínio matemático, devido a obrigar "à desconstrução e análise crítica dos problemas". Por outro lado, os alunos referiram que a linguagem *Python* tem a vantagem de possuir uma sintaxe simples e de "ter imensas ferramentas matemáticas".

No questionário final estes alunos realçaram a utilidade do trabalho executado, "uma vez que nos permitiu aprender um pouco mais de programação, que é uma excelente forma de desenvolver o raciocínio matemático". Realmente o raciocínio matemático, nas suas diversas formas, encontra-se ao longo de todo o programa que os alunos executaram para o projeto final, em particular no ciclo de repetição implementado em que as sucessivas iterações que se formam estão muito bem delineadas.

6.8.5 Conclusões

Estes alunos, no seu relatório, e como conclusão, consideraram que o seu trabalho permitiu uma compreensão mais aprofundada do método de *Newton-Raphson*, para o cálculo das raízes de uma função, uma vez que, para além de permitir a visualização gráfica deste algoritmo, também os obrigou a analisar o processo de uma forma mais complexa. Nesse sentido, acreditam que o projeto desenvolvido é um bom exemplo de como o uso da programação pode servir de complemento ao estudo da matemática, podendo revelar-se bastante benéfico.

O terem desenvolvido este projeto permitiu igualmente desenvolver abordagens diferentes para a resolução dos problemas matemáticos que se colocam, ajudando assim na sua compreensão.

Por último, consideram igualmente que o facto de terem, em termos metodológicos, executado um relatório e um vídeo permitiu explicar melhor o seu programa e garantiu que o compreendessem verdadeiramente.

Esta tarefa constituiu-se como um bom exemplo de como o PC, através da programação em *Python*, e a Matemática estão intimamente ligados. Os alunos foram bastante autónomos no trabalho executado e partindo de um programa estudado melhoraram-no e além de obter somente a solução conseguiram igualmente que o programa executasse uma visualização gráfica de como o método funciona com uma determinada função.

6.9 Estudo de caso 9 – Juros Compostos

Os dois alunos deste estudo de caso, H_9 e H_{10}, efetuaram um programa (Juros Compostos, Anexo V), a que apelidaram de Juros_compostos, no qual se pretendia saber qual o valor obtido com a aplicação de uma determinada quantia inicial num determinado período de tempo sob o efeito de juros compostos. Uma ideia em tudo semelhante à apresentada pelos estudos de caso 2 e 4.

6.9.1 Os alunos

Os rendimentos escolares dos dois alunos deste estudo de caso são ligeiramente diferentes. O aluno H_9 obteve no final do 12.º ano a classificação de 18 na disciplina de Matemática A, ficando com uma média dos 3 anos de 18 valores. O alunos H_{10} denota algumas dificuldades na disciplina de Matemática, tendo obtido no final do 3.º período do 12.º ano, a classificação de 13 e ficando com uma média dos 3 anos de 14 valores. No exame nacional de Matemática A, o aluno H_9 obteve 176 pontos e o aluno H_{10} não efetuou o exame por este não ser necessário para os cursos que pretendia ingressar. Na disciplina de AI obtiveram ambos 19 valores no final do 12.º ano. Ambos os alunos constam do quadro de mérito da escola.

6.9.2 Análise e objetivos do programa

Com o trabalho apresentado, os alunos pretenderam "demonstrar como o ramo da matemática se consegue relacionar com o da programação", sendo que, segundo eles, "uma matéria por vezes considerada difícil e cansativa" pode ser encarada como "mais simples e interativa".

Para estes alunos, as linguagens de programação exercem um papel fundamental na Internet, não só no "desenvolvimento de sites e softwares como também na interação entre humanos e máquinas por meio de inteligência artificial e na matemática". Consideram o *Python* uma linguagem de programação de alto nível, orientada a objetos, acessível e muito popular em setores emergentes da indústria de tecnologia.

No que concerne ao programa desenvolvido para este projeto, pretenderam saber qual o valor de retorno quando se investe dinheiro num banco, mediante a aplicação de juros sobre o dinheiro emprestado ou depositado, sendo que se for aplicado um juro simples pela aplicação de um capital num depósito durante um determinado período de tempo, significa que o capital obtido com a aplicação do juro após cada período de tempo T é sempre igual ao capital inicial.

Com juros compostos obter-se-á a soma de juros sobre juros que se irá receber sobre o valor depositado. O juro composto corresponde à capitalização dos juros simples que vão sendo adicionados ao montante inicial.

Como os alunos definem no seu relatório, "no juro composto, o juro produzido em cada período de tempo T é adicionado ao capital existente e, consequentemente, este será o capital inicial no período de capitalização seguinte. Este processo repete-se durante n períodos de tempo T", assim, dado um capital inicial, C_0, e aplicando-se juros compostos à taxa $r\%$ durante

um período temporal de T, o capital disponível ao fim de $n \in \mathbb{N}$ períodos de tempo T é igual a $C_n = C_0.\left(1 + \frac{r}{100}\right)^n$.

Para estes alunos, a criação de um programa que permitisse obter o capital acumulado quando se efetua um investimento num determinado banco poderia ser bastante útil. Para tal, utilizaram a linguagem de programação *Python*, na aplicação *Visual Studio Code*.

Importaram a biblioteca *Tkinter* que consiste num módulo do *Python* que usa muitas funções e métodos que podem ser usados na criação de um programa, nomeadamente permite a criação de caixas de diálogo. Para tal, começaram por criar um menu onde solicitam a escolha do tipo de taxa de juro a aplicar sobre o capital, podendo estas ser aplicadas anualmente, semestralmente ou mensalmente (figura 71).

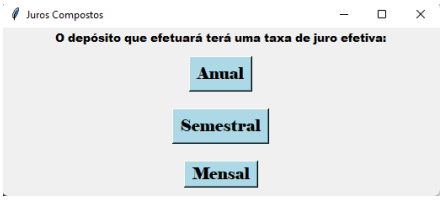


Figura 71 – Janela que permite a escolha da taxa de juro no menu inicial do programa Juros_Compostos

Atendendo à escolha do utilizador, assim o capital acumulado ao fim de x anos será diferente. Na figura 72, estão as linhas de código que permitem ao utilizador efetuar a escolha.

```
def openNewWindow():
    root = Toplevel(menu)
    root.title("Juros Compostos")
    txta = StringVar()
    z= 0
    def adding():
        d=0
        a=float(E1.get())
        b= float(E2.get())
        c= float(E3.get())
        z= abs(1+(b/100))**c
        d= a*z
        txta.set(str(d))
```

Figura 72 – Linhas de código que permitem escolher a taxa de juro no programa Juros_Compostos

De forma a que o programa conseguisse dar o resultado do capital acumulado, criaram *entry boxes*, que consiste numa ferramenta que permite ao utilizador do programa inserir um valor. Para este caso foram criadas três *entry boxes*: para o capital investido; para a taxa de juro aplicada, em %; e, para o período durante o qual essa taxa de juro é aplicada, em anos. A informação sobre o que escrever nas *entry boxes* foi feita através de *labels*. Na figura 73, encontra-se a janela que se obtém logo após a escolha do tipo de taxa de juro a aplicar ao capital inicial.

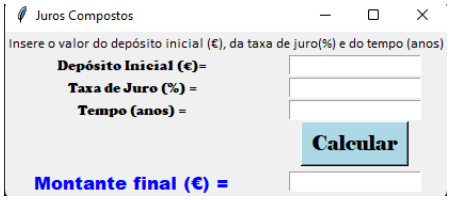


Figura 73 – Janela que permite introduzir o capital a investir, a taxa de juro e o tempo de aplicação no programa Juros_Compostos

No vídeo, que criaram de suporte à apresentação do programa efetuado, utilizaram um exemplo de aplicação retirado do manual adotado na escola (Negra et al., 2017), nomeadamente o exercício 2, da página 9, em que se aplica um depósito inicial de 1500€, sendo que nos primeiros 10 anos, a taxa de juro anual é de 2,5% e nos anos seguintes, a taxa de juro anual passa a 1,65%. Pretende-se saber qual é o capital acumulado ao fim de 18 anos.

Na sua resolução, a qual demonstraram no vídeo, começam por dividir o problema em duas partes. Na primeira, calculam o capital acumulado ao fim dos primeiros 10 anos, e, em seguida, como a taxa de juro anual é alterada, calculam o capital final acumulado ao fim de 18 anos. Após a utilização do programa, obtêm que o capital acumulado ao fim de 10 anos é de 1920,13€ (figura 74), e repetindo o processo, agora utilizando uma taxa de 1,65%, durante 8 anos, obtêm um valor final de 2188,72€ (figura 74).

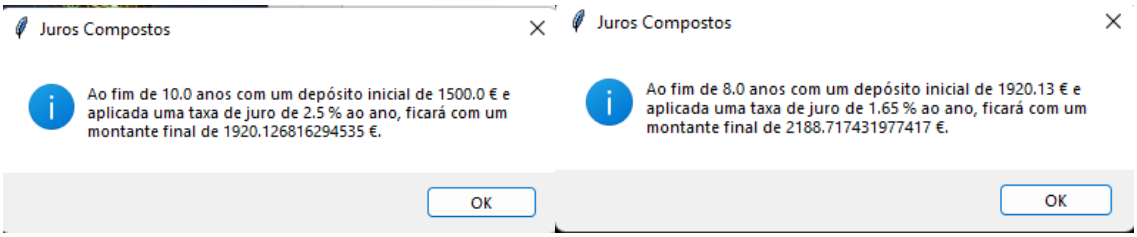


Figura 74 – Resolução com o programa Juros_Compostos do exercício 2, do manual adotado na escola (Negra et al., 2017, p. 9),

O programa desenvolvido por estes dois alunos, H_9 e H_{10}, tem 125 linhas de código e funciona essencialmente com caixas de diálogo que permitem uma boa comunicação com o utilizador e uma maior interatividade. É um programa que teve na sua génese conteúdos do programa de Matemática A, nomeadamente juros simples vs compostos e que permite a utilização de casos práticos dados na aula.

É um programa que utiliza uma biblioteca que não foi dada nas três aulas de introdução ao *Python* nas sessões de Aplicações Informáticas B, e que, como tal, obrigou estes dois a alunos a explorarem novas funcionalidades do *Python* para conseguirem efetuar o programa pretendido.

6.9.3 Processos utilizados e fontes procuradas

Estes alunos consideram que a escolha do tema foi bastante fácil, sendo que os juros compostos são uma "matéria fácil e que pode vir a ser bastante útil no nosso dia-a-dia". Recorreram ao *Google* e ao *Youtube* cada vez que não encontravam nos guiões a informação que necessitavam.

Ainda relativamente aos guiões que foram implementados nas sessões de trabalho, acham que estes “estão bons assim e não mudaríamos nada" pois contêm uma abordagem simples e de fácil compreensão a cada tema matemático, seguido sempre de um exemplo de aplicação. Consideraram úteis as entradas feitas no guião 2 à função condicional (*if-elif-else*) e no guião 3 ao ciclo de repetição *WHILE*.

6.9.4 Dificuldades sentidas e raciocínio matemático envolvido

Tendo por base o questionário final, estes alunos comentaram que as principais dificuldades sentidas se centraram na resolução dos erros que apareceram "ao longo da realização do trabalho e que não permitiam que este arrancasse". Tiveram várias vezes que procurar informação *online* para resolver pequenos obstáculos.

Para eles, a tarefa mais fácil de executar foi o exercício final do guião 1, sendo "necessário apenas definir variáveis e formatá-las". Já a tarefa mais difícil de executar foi o exercício final do guião 3, que “exigiu muita pesquisa”, bem como o trabalho de projeto em que tentaram “tornar o programa criado o mais interativo possível”.

Segundo a opinião dos alunos o trabalho efetuado não permitiu desenvolver raciocínio matemático pois já sabiam a matéria abordada. Acham, no entanto, "que desenvolveu o nosso pensamento enquanto programadores e aprender de certa forma como passar a matemática escrita para outra semelhante mas percetível em termos de programação".

O *Python* torna “a matemática mais simples e interativa”, uma vez que as novas gerações “estão muito voltadas para as novas tecnologias e ter noções básicas de como dominar algumas funções de uma linguagem de programação bastante utilizada pode ser útil”.

Consideram que o trabalho efetuado foi bastante útil para compreenderem a utilização da Matemática na linguagem de programação *Python*, sendo que as pesquisas efetuadas para a realização do projeto final possibilitaram a recolha de muita informação relativa ao *Python*.

6.9.5 Conclusões

Os alunos manifestaram a opinião que a realização deste tipo de tarefas lhes pode ser útil pois "o mundo está a mudar devido às novas tecnologias" e como tal o ensino na escola não pode ficar para trás. Este tipo de atividades, segundo relatam, deverão ter um caráter mais regular, sugerindo inclusive a criação de uma nova disciplina.

Em termos metodológicos não concordaram com a realização simultânea de um vídeo e de um relatório pois não são muito rigorosos a escrever as ideias, efetuando pequenos rascunhos, e para efetuar um relatório não chegou fazer pequenos rascunhos.

Efetivamente, o que se verificou no relatório destes alunos é que as ideias de programação que incorporaram no código estavam lá todas, mas eram algo confusas. No vídeo estas estavam mais bem organizadas e complementadas com pequenos exercícios de aplicação efetuados no programa.

O código efetuado por estes alunos aparentemente promove o raciocínio matemático, em particular o raciocínio indutivo e dedutivo, mais que não seja pelas funções condicionais introduzidas, e a sua execução possibilita o estabelecimento de conjeturas.

Em termos de PC está presente a abstração, a algoritmia e o reconhecimento de padrões, sendo que tal como é afirmado pelos alunos a duração foi fundamental para um bom funcionamento do programa.

6.10 Estudo de caso 10 – Derivação

Os dois alunos deste grupo, H_{11} e H_{12}, efetuaram um programa a que apelidaram de Derivar (Derivação, Anexo V), em que se pretendia obter a derivada de uma função real de variável real.

6.10.1 Os alunos

Os alunos que constituem este estudo de caso são dois dos melhores alunos da turma, ambos do quadro de mérito da escola, sendo que ambos obtiveram 20 na disciplina de Matemática A do 12.º ano. O aluno H_{11} obteve uma média final de 20 valores e o aluno H_{12} obteve 19. No exame nacional de Matemática A, o aluno H_{11} obteve 199 pontos e o aluno H_{12} 200 pontos. Ambos obtiveram 19 valores na disciplina de AI.

6.10.2 Análise e objetivos do programa

Este grupo de dois alunos pretendeu com o seu trabalho de projeto criar um programa em linguagem *Python* que fosse capaz de derivar funções polinomiais de grau inferior a 4, apresentando o gráfico da função e da sua derivada e indicar os zeros da derivada de forma a se conseguir obter os intervalos de monotonia da função.

De forma a tornar o código mais simples, dividiram os vários tipos de equações tendo por base a resposta a uma pergunta inicial, na qual o utilizador responde, escolhendo o grau da função polinomial que quer derivar, como mostra a figura 75.

```
============ RESTART: C:\Users\rikyb\Desktop\Derivar_[redacted]py ===========
Derivar uma função de 1°grau, 2°grau ou de 3°grau? Ans (1, 2 ou 3):
```

Figura 75 – Pergunta que permite ao utilizador escolher o grau da função polinomial que se quer derivar

Assim, os alunos associaram à resposta dada pelo utilizador uma variável numérica e criaram uma função condicional (*if-elif-else*), a qual dividiram em três partes (grau 1, 2 e 3) para poderem programar as regras de derivação especificas para cada grau e apresentar a função derivada da função considerada (figura 76).

```
m = int(input("Derivar uma função de 1°grau, 2°grau ou de 3°grau? Ans (1, 2 ou 3):"))
if m == 1:
elif m == 2:
else:
```

Figura 76 – Escolha do grau da função polinomial

As funções polinomiais do primeiro grau foram, aparentemente, e segundo os alunos, as mais fáceis de programar, pois a sua derivada é uma constante. Tendo por base a condição 'm == 1', o utilizador define a função, e, como o grau já está escolhido, basta ao utilizador introduzir os coeficientes da expressão polinomial do 1.º grau para a definir (o código encontra-se na figura 77).

```
if m == 1:
    a = int(input("a="))
    b = int(input("b="))

    def f(x) :
        return(a*x + b)
    print("f(x) = {}x + {}".format(a,b))
    print("A derivada de f(x) é df(x) = {}".format(a))
```

Figura 77 – Código do programa Derivar para introduzir os parâmetros da função afim

Com a instrução *input* o utilizador atribui os valores dos coeficientes (‘*a’* e ‘*b’*) e não é necessária qualquer outra rotina extra, visto que a derivada deste tipo de funções é apenas o valor do coeficiente ‘*a*’. Falta apenas representar graficamente a derivada desta função, a qual é dada pela instrução descrita na figura 78, uma vez que os alunos estão a utilizar a biblioteca *Matplotlib*.

```
plt.axhline(y=a, color='r', linestyle='-')
plt.show()
```

Figura 78 – Comandos inseridos no código do que permitem visualizar a representação gráfica da função derivada da função afim

Relativamente às funções polinomiais do 2.º grau, é necessário um pouco mais de código para se conseguir caraterizar a derivada e efetuar o estudo da monotonia, uma vez que os gráficos das derivadas das funções polinomiais do segundo grau não são apenas definidas por uma linha horizontal que se observa nas derivadas das funções de primeiro grau.

Tal como no caso das funções polinomiais do 1.º grau, o utilizador escolhe os coeficientes da equação (a, b e c) através da instrução *input*. No entanto, para obterem a derivada, os alunos recorreram à definição de derivada de uma função num ponto (figura 79).

$$f'(x) = \lim_{h \to 0} \frac{f(x+h) - f(x)}{h}$$

Figura 79 – Definição de derivada de uma função num ponto

Como se pode observar a definição de derivada tem por base o conceito de limite, “a ideia de que uma variável tende a aproximar-se infinitamente de um valor”. Contudo, para não ser necessário criar um algoritmo computacional que resolva o limite, os alunos recorreram a uma aproximação do valor real que, apesar de ser bastante simples, se aproxima o suficiente do valor real. Como no limite a variável ‘*h*’ tende para zero, pode-se substituir esta variável na expressão dada na figura 79 por um valor muito próximo de zero, por exemplo 0,00000000001[10], e este ‘artificio’ permite obter um valor aproximado para a derivada num ponto. Obtém-se assim a expressão expressa na figura 80, designada usualmente por derivada numérica.

[10] Nota: para calcular com o computador em aritmética de precisão finita, o h não pode ser demasiado pequeno, se for muito menor do que 10^(-8) pode dar erros bastante grandes. Para os fins deste programa o valor de h escolhido (10^(-10)) funciona bem.

$$f'(a) = \frac{f(a + 0.00000000001) - f(a)}{0.00000000001}$$

Figura 80 – Derivada numérica

Assim, em *Python*, tem-se o código expresso na figura 81.

```
def f(x) :
    return(a*x**2 + b*x + c)

def df(x) :
    h = 0.00000000001
    fun = (f(x+h)-f(x))/(h)
    return float("%.3f" %fun)
```

Figura 81 – Código que permite obter a derivada numérica em *Python*

Os alunos utilizaram a função *float* que permite arredondar um número real para o valor inteiro, tornando a função aproximada ainda mais próxima da derivada.

Para as funções polinomiais do 2.º grau e superior, importa escolher o 'domínio' da função que se quer visualizar. Os alunos deram essa liberdade de escolha ao utilizador (figura 82).

```
#Dominio
d = int(input("Limite inferior do domínio da função é "))
D = int(input("Limite superior do domínio da função é "))
```

Figura 82 – Escolha do domínio da função polinomial com a qual se pretende trabalhar

Depois de definido o 'domínio' tem que se associar a função derivada a uma representação gráfica (figura 83). Para que isso aconteça é necessário utilizar a biblioteca *numpy* que permite dar vários valores a x dentro do domínio considerado que depois serão transformados em imagens pela função inserida, criando assim os vários pontos do gráfico.

```
#Funções
x = np.linspace(d,D,100)
y = real_df(x)
j = f(x)
```

Figura 83 – Linhas de código que permitem a associação entre uma função derivada e a sua representação gráfica

Como se pode observar na figura 83, a função *numpy*.linspace atribui a 'x' 100 valores equidistantes dentro do domínio [d, D], definido pelo utilizador. Depois os valores de 'y' são

obtidos em função de ‘x’ e através dos comandos da biblioteca matplotlib pode-se criar um gráfico com esses 100 pontos com a instrução plot (figura 84).

```
plt.plot(x,j, label="f(x)")
plt.plot(x,y, label="df(x)")
plt.legend()
plt.show()
```

Figura 84 – Linhas de código que permitem obter o gráfico da função e da função derivada

Para calcular os zeros da função, os alunos sentiram a necessidade de recorrer a uma nova biblioteca chamada *Scipy*. Neste módulo existe uma função *fsolve*, que permite obter os zeros de uma função, em particular da derivada. Para que tal aconteça é necessário escolher um ponto do domínio da função inicial para que o comando *fsolve* encontre o valor aproximado do zero mais próximo através de um método iterativo de resolução aproximada de equações (figura 85), do género do método de *Newton-Raphson*, estudado nas aulas de introdução ao *Python*.

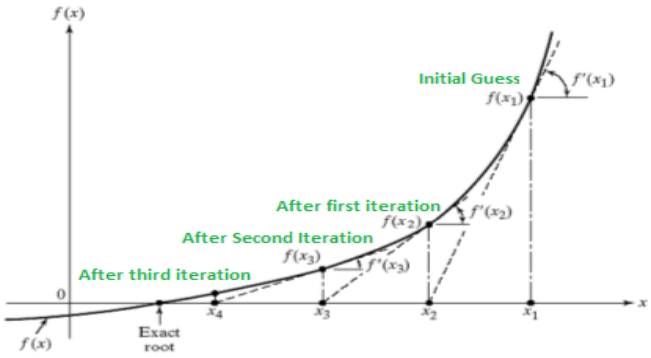


Figura 85 – Método iterativo de resolução de equações

Em linguagem *Python*, o comando introduzido no código do programa Derivar que permite calcular os zeros da derivada de uma dada função polinomial está na figura 86.

```
#Zeros
dzeros = fsolve(real_df, D)
print("Os zeros da derivada são: ", dzeros)
```

Figura 86 – Código do programa com o comando *fsolve* do módulo *Scipy*

Como argumento da função *fsolve* tem-se, em primeiro lugar, a função na qual se pretende encontrar o zero (real_df) e, em segundo lugar, o valor inicial do qual parte o processo iterativo.

Por último, e após o cálculo dos zeros da derivada, pode-se estudar a monotonia das funções pretendidas, partindo do pressuposto que se sabe que a derivada de uma função

quadrática é uma função afim. Existem dois casos possíveis expressos nas linhas de código da figura 87.

```
#Monotonia
if z>0 :
    print("A Função f é crescente em ]{},{}] e decrescente em [{}, {}[".format(dzeros,D,d,dzeros))
else:
    print("A Função f é crescente em [{},{}[ e decrescente em ]{}, {}]".format(d,dzeros,dzeros,D))
```

Figura 87 – Linhas de código do programa Derivar que permitem estudar a monotonia de uma função

No primeiro caso expresso na figura 87, a derivada tem declive positivo, '$z > 0$' (relembra-se que $z = a * 2$, sendo 'a' o coeficiente do termo de maior grau, neste caso 2, e se 'z' for maior que zero, 'a' também o será), logo, a função é decrescente desde o valor inferior definido para o domínio (d) até ao zero (dzeros), e crescente do zero (dzeros) ao valor superior do domínio (D). Caso contrário, ou seja, com $z < 0$, a derivada tem declive negativo, logo a função é decrescente desde o zero (dzeros) ao limite superior do domínio (D), e crescente do limite inferior do domínio (d) ao zero (dzeros).

Relativamente às funções polinomiais do 3.º grau, o utilizador necessita atribuir os valores dos coeficientes e escolher o domínio da função, o que é feito no código através da função *input*. Como produtos obtêm-se os gráficos da função e da sua derivada, sendo que o processo de obtenção dos gráficos é semelhante ao anterior. Os zeros e o estudo da monotonia têm, no entanto, um processo um pouco diferente, que se passa a explicar em seguida.

Como a instrução *fsolve* apenas encontra um zero de cada vez, é necessário começar com dois pontos diferentes para obter dois zeros diferentes que a derivada de grau 2 apresenta, no máximo. Começa-se com dois valores diferentes, neste caso com o limite inferior do domínio e com o limite superior (d, D). Assim obtém-se o menor e o maior zero (mzero, Mzero). Na figura 88 estão as linhas de código que permitem obter os zeros da derivada de uma função polinomial de grau 3.

```
#Zeros
mzero = fsolve(real_df,d)
Mzero = fsolve(real_df,D)
print("Os zeros da derivada são: {} e {}".format(mzero,Mzero))
```

Figura 88 – Linhas de código do programa Derivar que permitem obter os zeros da derivada de uma função polinomial de grau 3

Como a derivada de uma função polinomial de terceiro grau é uma função quadrática, obtêm-se no máximo 2 zeros (mzero e Mzero). De forma semelhante para as funções de grau dois existem dois casos diferentes (figura 89).

```
#Monotonia
if v>0 :
    print("A Função f é crescente em ]{},{}] e [{}, {}[ e decrescente em [{}, {}]".format(d,mzero,Mzero,D,mzero,Mzero))
else:
    print("A Função f é crescente em [{},{}] e decrescente em ]{},{}] e [{}, {}[".format(mzero,Mzero,d,mzero,Mzero,D))
```

Figura 89 – Linhas de código que permitem obter os intervalos de monotonia de uma função polinomial de grau 3

No primeiro caso, a derivada tem o sentido da concavidade voltada para cima, '$v > 0$' (relembrando, $v = a * 3$, sendo 'a' o coeficiente do termo de maior grau, neste caso 3. Se 'v' for maior que zero, 'a' também o será), logo, a função f será crescente onde a derivada é positiva e decrescente onde a derivada apresenta valor negativo, neste caso com o sentido da concavidade voltada para cima tem-se que a derivada é positiva do limite inferior do domínio (d) até o menor zero (mzero) e do zero superior (Mzero) até o limite superior do domínio (D); e tem valor negativo entre os dois zeros (mzero, Mzero).

As figuras 90, 91 e 92 apresentam exemplos de execução do programa para funções afim, quadrática e cúbica, respetivamente.

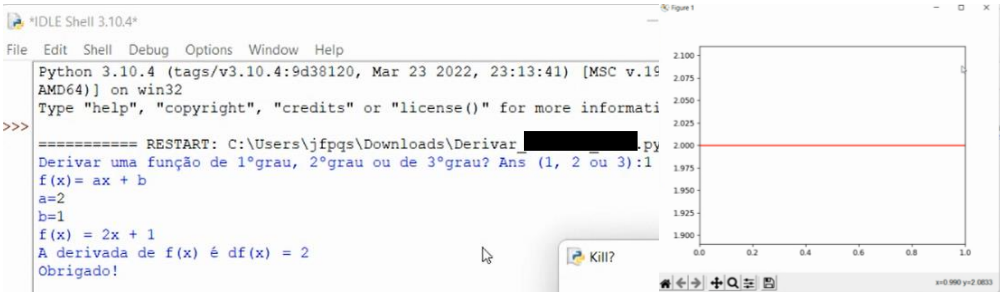


Figura 90 – Exemplo de execução do programa Derivar para uma função afim

Figura 91 – Exemplo de execução do programa Derivar para uma função quadrática

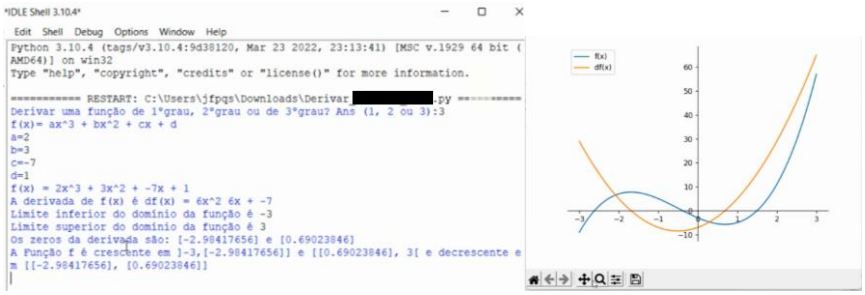


Figura 92 – Exemplo de execução do programa Derivar para uma função cúbica

O programa desenvolvido por estes dois alunos tem 140 linhas de código e baseia-se num processo de decisão dado por uma função condicional, ‘trabalhando’ essencialmente com comandos *input* e *print*.

É um programa que tem por base as aplicações da derivada de uma função real de variável real e que utiliza várias bibliotecas que não foram dadas nas três aulas de introdução ao *Python*.

6.10.3 Processos utilizados e fontes procuradas

Além das bibliotecas *NumPy*, *SciPy*, *Matplotlib* e *Math*, os alunos referem no questionário final que recorreram à internet, no sentido de obter mais funções, e à semelhança dos colegas utilizaram igualmente os guiões fornecidos.

Relativamente aos guiões consideram que “estavam explícitos, mas feitos de forma a desafiar-nos”, não propondo qualquer melhoria.

6.10.4 Dificuldades sentidas e raciocínio matemático envolvido

Estes dois alunos sentiram dificuldades em "conseguir passar as nossas ideias em pseudocódigo para linguagem formal".

Relativamente à tarefa mais fácil, elegeram a primeira, pois "eram fornecidos quase todos os dados". Por outro lado, consideraram o projeto final a tarefa mais "desafiante" em que tiveram "não só de consolidar tudo o que aprenderam até ai, mas adicionalmente de procurar e aprender conceitos novos".

Ambos os alunos consideram que o facto de terem efetuado este tipo de tarefas "permitiu pensar em conceitos matemáticos já bastante conhecidos, (…) formas diferentes, formas computáveis, maneiras de pensar novas." Consideram que este trabalho constituiu "uma boa introdução à matemática na forma de programação e contribuiu de uma maneira diferente para pensar nos problemas que normalmente resolvemos".

6.10.5 Conclusões

Os alunos referem nas suas respostas ao questionário final que "a programação é uma ferramenta cada vez mais usada nos dias de hoje, não só em matemática, mas em muitas outras áreas do saber."

Referem que o desenvolvimento destas tarefas na disciplina de Matemática e o conhecimento que foi adquirido pode ser transportado para outras disciplinas. Terminam salientando

que teria siso suficiente ter efetuado um dos dois elementos, vídeo ou relatório, para demonstrar a plena compreensão do seu programa.

Da análise do programa, do relatório e do vídeo apresentado pelos alunos, verifica-se que estes compreenderam os conceitos estudados na aula de Matemática A e os utilizaram de forma bastante 'engenhosa' na construção do programa em *Python,* desenvolvendo diversos tipos de raciocínio Matemático no processo. Como se constata, o PC está perfeitamente presente nas suas diversas vertentes.

6.11 Estudo de caso 11 – Triângulo de Pascal

As duas alunas deste grupo, F_9 e F_{10}, elaboraram um programa que intitularam de Triângulo de Pascal (Triângulo de Pascal, Anexo V), que devolve todos os valores de uma determinada linha do triângulo de Pascal após a introdução do número da linha pelo operador.

6.11.1 As alunas

As alunas que constituem este estudo de caso são duas alunas que apresentam um rendimento escolar médio, sendo que a aluna F_9 obteve 13 na disciplina de Matemática A do 12.º ano, ficando com uma média de 12, e a aluna F_{10} obteve 14, ficando com 15 de média final dos três anos. No exame nacional de Matemática A, a aluna F_9 obteve 136 pontos e a aluna F_{10} obteve 133 pontos. As alunas F_9 e F_{10} obtiveram 18 e 19 valores, respetivamente, na disciplina de AI. É de realçar que apesar das classificações das duas alunas serem muito idênticas a aluna F_{10} integra o quadro de mérito da escola.

6.11.2 Análise e objetivos do programa

Para estas alunas, a linguagem *Python* é de alto nível, multiparadigma, orientada a objetos, imperativa e funcional. Em virtude das caraterísticas apresentadas, esta linguagem é utilizada, principalmente, para processamento de textos, dados científicos e criação de páginas dinâmicas para a web.

O seu objetivo foi desenvolver um programa que permitisse obter o Triângulo de Pascal, o qual como é referido pelas alunas "consiste num triângulo aritmético infinito onde são dispostos os coeficientes das expansões binominais", sendo que os números que compõem este triângulo apresentam diversas propriedades e relações.

Pretendiam assim dar resposta às seguintes questões:

1. Listar os elementos do triângulo de Pascal;

2. Identificar um elemento do triângulo de Pascal;
3. Somar os elementos de uma linha do triângulo de Pascal;
4. Somar todos os elementos anteriores a uma linha no triângulo.

As alunas F_9 e F_{10} construíram um código que é constituído por uma rotina inicial, onde o operador pode escolher uma das opções de entre as listadas acima.

Para que o programa execute a opção 1, as alunas criaram no código do programa a função *desenhar_triangulo_pascal*(input_num), a qual recebe como parâmetro o número de linhas que se pretende ver listadas do triângulo de Pascal e apresenta esses elementos na consola.

Identifica-se na figura 93 o código que constitui a primeira parte da função indicada, formada por dois ciclos *for*, um dentro do outro, em que se pretende obter várias listas de números com os elementos do triângulo de Pascal até à linha pretendida.

```
list = [] #lista vazia
for n in range(input_num):
  list.append([])
  list[n].append(1)
  for m in range(1, n):
     list[n].append(list[n - 1][m - 1] + list[n - 1][m])
  if(input_num != 0):
     list[n].append(1)
```

Figura 93 – Primeira parte da função *desenhar_triangulo_pascal*(input_num) do programa Triângulo de Pascal

Na figura 94, encontra-se a segunda parte da mesma função, cujo objetivo é apresentar na consola o triângulo de Pascal com o número de linhas pretendido.

```
for n in range(input_num):
   print(" " * (input_num - n), end = " ", sep = " ")
for m in range(0, n + 1):
print('{0:5}'.format(list[n][m]), end = " ", sep = " ")
   print()
```

Figura 94 – Segunda parte da função *desenhar_triangulo_pascal*(input_num) do programa Triângulo de Pascal

Na figura 95, pode observar-se a execução do programa Triângulo de Pascal após o utilizador escolher a opção 1 na consola.

```
1-Listar triângulo de pascal
2-Identificar um elemento do triângulo de pascal
3-somar os elementos de uma linha do triângulo de pascal
4-somar todos os elementos anteriores a uma linha no triângulo
Introduza uma opção [1 a 4]:1

1-Listar triângulo de pascal

Insira o número de linhas: 7

Desenhar triângulo de pascal

            1
          1     1
        1     2     1
      1     3     3     1
    1     4     6     4     1
  1     5    10    10     5     1
1     6    15    20    15     6     1
```

Figura 95 – Execução da opção 1 pelo programa Triângulo de Pascal

Caso o utilizador escolha a opção 2 da lista indicada anteriormente, o programa executa a função *elemento*(linha, coluna), a qual recebe como parâmetros uma linha e uma coluna do triângulo de Pascal, e devolve o elemento do triângulo de Pascal respetivo (figura 96).

```
    list = [] #lista vazia
    for n in range(linha):
      list.append([])
      list[n].append(1)
      for m in range(1, n):
         list[n].append(list[n - 1][m - 1] + list[n - 1][m])
      if(linha != 0):
         list[n].append(1)
         print(list[n])
         print(n)
  return(list[n][coluna])
```

Figura 96 – Parte da função *elemento*(linha, coluna) do programa Triângulo de Pascal

Na figura 97, encontra-se a execução do programa Triângulo de Pascal, após o utilizador escolher a opção 2 e selecionar o elemento que se encontra na linha 8, com início em 0, e na coluna 6.

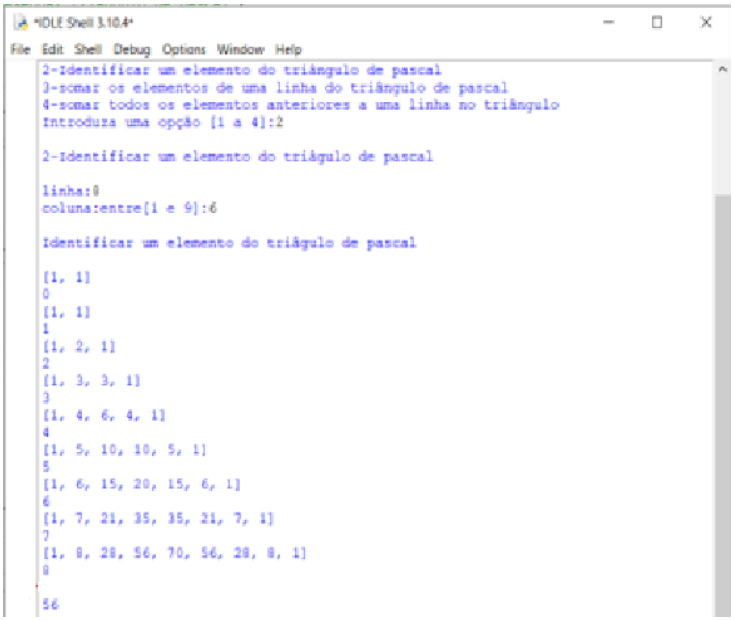


Figura 97 – Apresentação na consola do elemento do triângulo de Pascal que se encontra na linha 8 e coluna 6

Como se pode verificar na figura 97, o programa apresenta na consola todos os elementos até à linha escolhida, neste caso linha 8, e termina indicando qual é o elemento dessa linha que está na coluna 6 (posição 5).

Caso o utilizador escolha a opção 3, da lista listada anteriormente, as alunas F_9 e F_{10}, criaram no código a função *soma_linha*(linha), a qual recebe como parâmetro o número de uma linha do triângulo de Pascal e devolve a soma dos elementos dessa linha. A instrução colocada nessa função é *return*(2**linha). Neste caso as alunas optaram por calcular o valor pretendido recorrendo diretamente à fórmula aplicável (2^n.º da linha).

Na figura 98 pode-se encontrar a execução do programa ao optar pela opção 3 e Indicando a linha 6.

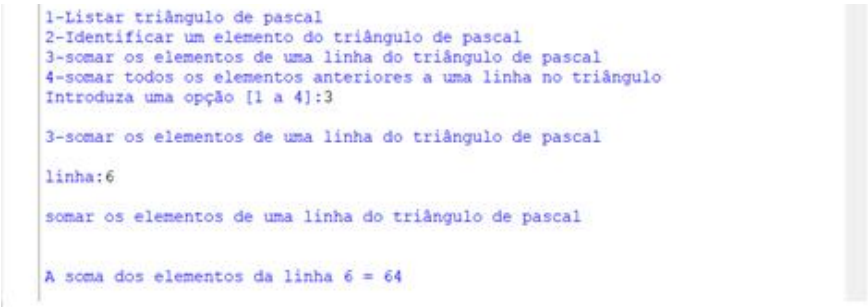


Figura 98 – Apresentação na consola do valor da soma de todos os elementos da linha 6 do triângulo de Pascal

Se o utilizador escolher a opção 4, as alunas introduziram no código a função *soma_anteriores_linha*(linha), que ao receber como parâmetro o número de uma linha do triângulo de

Pascal, permite que o programa determine a soma de todos os elementos desde a 1.ª linha até à linha anterior à escolhida pelo operador.

```
def soma_anteriores_linha(linha):
  print(" ")
  print ("Somar todos os elementos anteriores a uma linha")
  print(" ")
  soma = 0
  for x in range(linha-1):
    soma = soma + 2**x
  return(soma)
```

Figura 99 – Função *soma_anteriores_linha*(linha) do programa Triângulo de Pascal

Na figura 100 pode-se verificar a soma de todos os elementos constantes em todas as linhas anteriores à linha 7, correspondente à execução da função apresentada na figura 99.

```
1-Listar triângulo de pascal
2-Identificar um elemento do triângulo de pascal
3-somar os elementos de uma linha do triângulo de pascal
4-somar todos os elementos anteriores a uma linha no triângulo
Introduza uma opção [1 a 4]:4

4-somar todos os elementos anteriores a uma linha no triângulo

linha:7

Somar todos os elementos anteriores a uma linha


A soma de todos elementos das linhas anteriores à linha 7 = 63
```

Figura 100 – Apresentação na consola do valor da soma de todos os elementos da linhas anteriores à linha 7

Tal como já foi afirmado, de forma a que o programa execute uma das funções anteriormente descritas, é necessário que o utilizador escolha uma opção. As alunas criaram no programa uma lista com as opções para que o utilizador pudesse escolher a função que pretendia ver executada (figura 101).

```
op_lista=["1-Listar triângulo de pascal", "2-Identificar um
elemento do triângulo de pascal", "3-somar os elementos de
uma linha do triângulo de pascal", "4-somar todos os elemen-
tos anteriores a uma linha no triângulo"]
```

Figura 101 – Comando no programa Triângulo de Pascal que permite apresentar as 4 opções possíveis

O programa assegura que o utilizador terá obrigatoriamente de introduzir uma opção correta entre 1 e 4 caso contrário não segue em frente. Em função da opção 1 a 4 que for escolhida, assim será executada a função desenvolvida (figura 102).

```
while (True):
  print("1-Listar triângulo de pascal")
  print("2-Identificar um elemento do triângulo de pascal")
  print("3-somar os elementos de uma linha do triângulo de pascal")
  print("4-somar todos os elementos anteriores a uma linha no triângulo")
  op = 0
  op_int = int (op)
  while (op_int < 1 or op_int > 4):
    op = input('Introduza uma opção [1 a 4]:')
    op_int = int (op)
  print (op_lista[op_int-1])
  if op_int == 1:
     input_num = int(input("Insira o número de linhas: "))
     desenhar_triangulo_pascal(input_num)
  elif op_int == 2:
     linha = int(input('linha:'))
     coluna = 0
     while (coluna < 1 or coluna > linha + 1):
          coluna = int(input('coluna:' + 'entre[1 e ' + str(linha+1) +']:'))
          elem = elemento(linha+1,coluna-1)
          print(elem)
  elif op_int == 3:
     linha = int(input('linha:'))
     soma_linha = soma_linha(linha)
     print('A soma dos elementos da linha ' + str(linha) + ' = ' + str(soma_linha))
  else:
     linha = int(input('linha:'))
     soma = soma_anteriores_linha(linha)
     print('A soma de todos elementos das linhas anteriores à linha ' + str(linha)
+ ' = ' + str(soma))
```

Figura 102 – Função que permite escolher a opção pretendida na lista

Este programa é composto por 115 linhas de código, sendo constituído essencialmente por ciclos de repetição *For* ou *WHILE* e por *print´*s e *input´*s, mas com um nível de trabalho bastante bom e que executa na perfeição as 4 operações que normalmente se pretendem pesquisar num triângulo de Pascal, a saber: listar os elementos de uma determinada linha no triângulo de Pascal; identificar um elemento do triângulo de Pascal; somar os elementos de uma determinada linha do triângulo de Pascal; e, obter o valor da soma de todos os elementos anteriores a uma determinada linha no triângulo.

6.11.3 Processos utilizados e fontes procuradas

Para a prossecução deste trabalho de projeto as alunas recorreram maioritariamente à internet e a alguns conselhos de familiares.

Por outro lado, destacam a utilidade e organização dos guiões apresentados que lhes permitiram compreender melhor “como se trabalha com o Python”, frisando que todos “tinham os passos bem definidos para o nosso melhor entendimento”. Nesse sentido não sugeriram qualquer alteração aos guiões.

6.11.4 Dificuldades sentidas e raciocínio matemático envolvido

Destacaram que a maior dificuldade sentida não foi na produção do código necessário para a execução do trabalho de projeto, mas na elaboração do vídeo pretendido.

Relativamente às tarefas propostas nos guiões consideraram que a mais fácil de executar foi o programa que pretendia determinar o declive e a ordenada na origem de uma reta dadas as coordenadas de dois pontos, sendo que a mais difícil foi a que propunha aplicar a fórmula resolvente a equações canónicas do 2.º grau “porque demorei a entender o processo para a programar”, como é relatado por uma das alunas no questionário final.

Segundo as alunas, o facto de terem programado em *Python* permitiu-lhes desenvolver o raciocínio matemático pois, "houve uma envolvente da matéria de matemática com a programação em Python".

Para estas alunas "programar em Python adequa-se muito bem à resolução de problemas em várias áreas da Matemática e potencia o desenvolvimento do raciocínio lógico".

O trabalho executado "foi bastante útil, pois passamos a conhecer uma nova linguagem de programação". Evidenciaram igualmente que conseguiram usar a matemática que aprenderam de uma forma diferente, variando assim “da monotonia das aulas”.

6.11.5 Conclusões

Com este trabalho as alunas referem que conseguiram compreender melhor a linguagem de programação *Python* e permitiu-lhes aprender algumas das principais bases de programação. Por outro lado, “graças ao desenvolvimento desta aplicação, bem como da possibilidade do uso da programação em aula, concluímos que é possível usar o Python nas mais variadas e complexas aplicações”. Terminam o seu relatório concluindo que “a matemática está bastante presente nos diversos tipos de programação”.

Em termos da metodologia que foi adotada para a execução deste trabalho de projeto as alunas concordam que foi útil terem executado um vídeo e um relatório pois assim sentiram-se obrigadas “a desenvolver uma base mais sólida para o trabalho”.

Relativamente ao raciocínio matemático utilizado no seu projeto estas alunas basearam-se essencialmente no raciocínio dedutivo e indutivo o que é evidenciado por apresentarem ao longo do código do programa construído várias funções condicionais e ciclos de repetição.

6.12 Estudo de caso 12 – Interseções

Os dois alunos deste grupo misto, *F11* e *H13*, efetuaram um programa a que apelidaram de Interseções (Interseções, Anexo V), no qual se pretende obter a resposta para diferentes tipos de interseção: entre uma reta e um plano; entre duas retas; e, entre dois planos, o que segundo eles constitui "um desafio devido à complexidade de algumas deduções de fórmulas".

6.12.1 Os alunos

Têm um rendimento escolar muito semelhante e ambos pertencem ao quadro de mérito da escola. A aluna *F11* obteve no final do 12.º ano a classificação de 19 valores na disciplina de Matemática A, ficando com uma média final de 17. O aluno *H13* conseguiu um pouco mais na disciplina de Matemática A, tendo conseguido 20 valores, sendo a média final dos três anos de 19 valores. No exame nacional de Matemática A, ambos obtiveram 195 pontos. Na disciplina de AI obtiveram 19 valores.

6.12.2 Análise e objetivos do programa

Para o cálculo das interseções referidas anteriormente, estes dois alunos partiram de fórmulas que foram objeto de estudo em anos anteriores e que foram revisitadas no 12.º ano.

O programa que construíram (Interseções, Anexo V), permite, entre outras funções, determinar o resultado da interseção entre duas retas, quer estas estejam definidas no plano, quer no espaço.

No plano, os alunos iniciam o seu programa definindo uma função iniciar(), que pergunta ao utilizador que tipo de interseção este pretende efetuar, sendo que caso o utilizador introduza 'rr' o código desenvolvido solicita ao operador que introduza valores para os coeficientes a e b de duas retas quando as equações das retas se encontram na forma reduzida ($y = ax + b$), como se constata na figura 103.

```
def iniciar():
  print("Que interseção queres determinar?")
  pergunta=input()
  if (pergunta=="rr"):
    print("Escreve a equação das retas na forma:")
    print("y = a(x)+ b")
    a1=float(input("declive da reta 1 - "))
    b1=float(input("ordenada na origem da reta 1 - "))
    a2=float(input("declive da reta 2 - "))
    b2=float(input("ordenada na origem da reta 2 - "))
```

Figura 103 – Parte inicial da função iniciar() do programa Interseções.

Após a introdução de valores, o programa efetua alguns testes recorrendo a funções condicionais (*if-elif-else*), como se verifica na figura 104, nomeadamente para verificar se as retas são coincidentes ou estritamente paralelas, e caso não o sejam, calcula os valores das coordenadas do ponto de interseção entre as duas retas no plano (figura 104).

```
        if a1==a2:
            if b1==b2:
               print("As retas são coincidentes")
               print(" ")
               iniciar()
            else:
               print("As retas são paralelas")
               print(" ")
               iniciar()
               print(" ")
        else:
            x=(b2-b1)/(a1-a2)
            y=(a1*x+b1)
            print("As retas intersetam-se no ponto de coordena-
das:{:.2f};{:.2f}.".format(x,y))
            print(" ")
            iniciar()
        print(" ")
```

Figura 104 – Continuação da função iniciar() do programa Interseções, que possibilita o teste dos coeficientes das equações reduzidas de duas retas no plano introduzidos e o cálculo das coordenadas do ponto de interseção quando as retas se intersetam.

Na figura 105 encontra-se um exemplo de execução desta parte do programa.

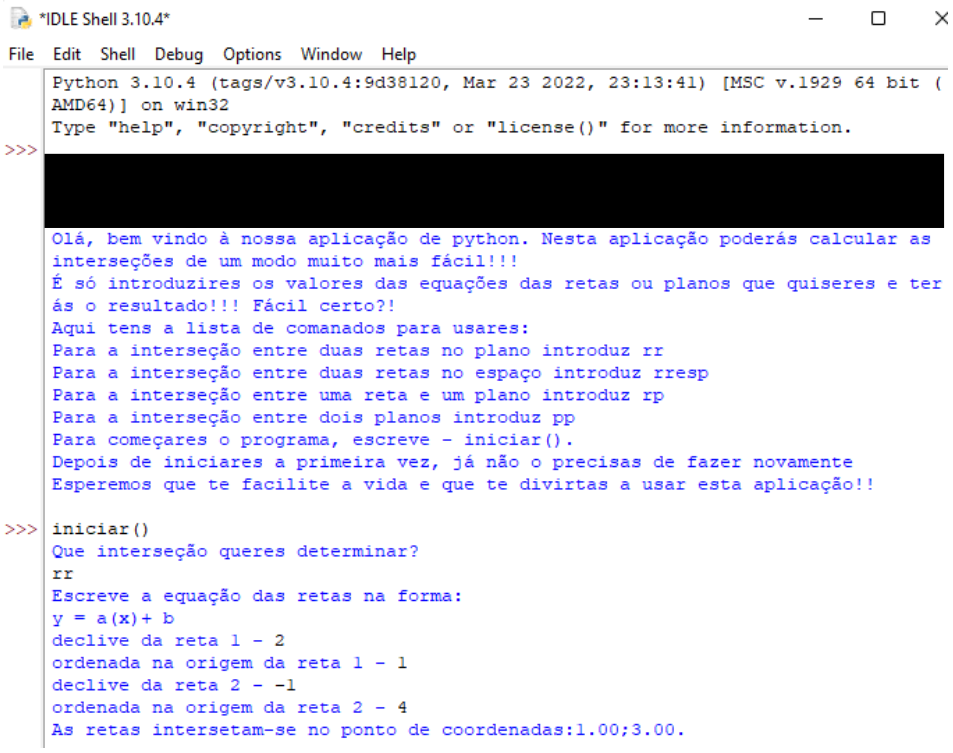


Figura 105 – Execução do programa Interseções para a opção de interseção entre duas retas no plano.

Como se pode verificar, este programa não permite calcular as coordenadas do ponto de interseção entre duas retas se uma delas ou ambas forem verticais.

No espaço, os alunos recorreram à equação vetorial da reta no espaço, a saber:

$$(x, y, z) = (x1, y1, z1) + k(Vx1, Vy1, Vz1), k \in \mathbb{R}$$

De forma a obterem as coordenadas do ponto de interseção entre duas retas, deduziram as equações paramétricas de ambas as retas (a primeira em função de k e a segunda em função de λ). Em seguida, igualaram ambas as 'abcissas' e resolveram a equação assim obtida em ordem a *k*. Depois, substituíram a expressão de *k* nas equações restantes (de y e de z) e resolveram essas equações em ordem a λ. Assim, obtiveram duas expressões para o valor de λ . Se os valores de λ forem iguais, significa que as retas se intersetam, caso contrário não se intersetam, como referem no seu relatório. Por último, substituíram este valor na equação paramétrica da reta correspondente de forma a obter as coordenadas do ponto de interseção.

Em termos de código, precisa-se de perceber se os vetores diretores das retas são ou não colineares. Esta situação verifica-se se o valor dos quocientes entre as respetivas coordenadas dos vetores diretores das retas forem iguais uns aos outros. No entanto, a aplicação direta desta condição obriga a que as coordenadas dos vetores que se encontram no denominador sejam diferentes de 0. Para evitar esta situação, os alunos adicionaram uma condição ao programa – a primeira coordenada do vetor diretor tem que ser diferente de 0. Como referem no seu relatório, "assim, o programa realiza a divisão acima referida, calcula a constante do quociente entre os vetores, e verifica se: Vx2*k==Vx1 and Vy2*k==Vy1 and Vz2*k==Vz1, que, sabendo o k, é equivalente à divisão das coordenadas dos vetores. Se isto se realizar, então as retas são paralelas [ou coincidentes]".

Caso contrário, procede-se ao cálculo de λ. Se os dois valores de λ forem iguais, então calcula-se o ponto de interseção. Caso não o sejam, "então as retas são enviesadas".

Em seguida, para que a primeira coordenada do vetor diretor possa ser 0, repete-se todo o procedimento assinalado, mas com as 2.ª e 3.ª coordenadas do vetor diretor diferentes de 0, sendo o cálculo do k feito pelo quociente destes com as respetivas coordenadas.

O código descrito pode ser analisado em Interseções (Anexo V), sendo que, na figura 106, se encontra um exemplo de execução dessa parte do programa.

```
Que interseção queres determinar?
rresp
Escreve a equação das retas na forma:
(x;y;z)=(x;y;z)+k(Vx;Vy;Vz)
abcissa do ponto da reta 1 - 1
ordenada do ponto da reta 1 - 2
cota do ponto da reta 1 - 1
coordenada x do vetor da reta 1 - 1
coordenada y do vetor da reta 1 - -1
coordenada z vetor da reta 1 - 1
abcissa do ponto da reta 2 - 0
ordenada do ponto da reta 2 - 4
cota do ponto da reta 2 - 1
coordenada x do vetor da reta 2 - -2
coordenada y do vetor da reta 2 - 3
coordenada z do vetor da reta 2 - -1
As duas retas intersetam-se no ponto de coordenadas:(2.00;1.00;2.00).
```

Figura 106 – Execução do programa Interseções para a opção de interseção entre duas retas no espaço.

Esta parte do programa deveria ainda conter alguns testes que os alunos não conseguiram prever a tempo da entrega, nomeadamente, se introduzirmos os vetores de coordenadas $(1,-1,1)$ e $(-2,1,-1)$ o programa dá uma mensagem de erro. Consegue o leitor perceber o porquê de tal situação, não sendo nenhuma das coordenadas zero?

Em seguida, os alunos analisam a interseção entre uma reta e um plano. Para a equação da reta consideram que o utilizador deverá considerar a reta numa sua forma vetorial, $((x,y,z) = (x1,y1,z1) + k(Vx1,Vy1,Vz1), k \in \mathbb{R})$. sendo-lhe solicitado que introduza as coordenadas de um ponto e as coordenadas de um vetor diretor. Para a equação do plano consideraram a sua forma geral $(x1(x) + y1(y) + z1(z) + d1 = 0)$. sendo que ao utilizador é solicitado que introduza os valores dos parâmetros $x1, y1, z1\ e\ d1$. Para analisar este tipo de interseção, os alunos começaram por escrever a equação paramétrica da reta e posteriormente, substituíram cada parte da equação paramétrica pelas coordenadas correspondentes ao ponto pertencente ao plano. Resolveram a equação e obtiveram uma expressão para o valor de k. Por último e "de forma a determinar o ponto de interseção, temos que substituir o valor de k na equação paramétrica da reta".

Em termos de código, os alunos foram verificar se a reta era coincidente ou estritamente paralela ao plano o que acontece se o vetor diretor da reta é perpendicular a um vetor normal ao plano. Para efetuar tal verificação basta colocar a condição no programa:

$$(Vx1 * x2) + (Vy1 * y2) + (Vz1 * z2) == 0.$$

Caso a condição se verifique, tem que se verificar se o ponto da reta pertence ao plano, e se assim for então a reta está contida no plano. Se não pertencer, a reta e o plano são, estritamente paralelos. Se os vetores não forem perpendiculares, então introduziram comandos no programa que possibilitasse o cálculo das coordenadas do ponto de interseção através das fórmulas deduzidas.

Também neste caso o código descrito pode ser analisado com mais detalhe em Interseções (Anexo V), sendo que, na figura 107, se encontra um exemplo de execução do programa.

```
Que interseção queres determinar?
rp
Escreve a equação da reta na forma:
(x;y;z)=(x;y;z)+k(Vx;Vy;Vz)
Escreve a equação do plano na forma:
a(x)+b(y)+c(z)+d=0
abcissa do ponto da reta - 1
ordenada do ponto da reta - 1
cota do ponto da reta - 1
coordenada x do vetor da reta - -1
coordenada y do vetor da reta - 0
coordenada z vetor da reta - 1
coordenada a vetor do plano - 2
coordenada b vetor do plano - 1
coordenada c vetor do plano - -1
coordenada d do plano - 1
A reta e o plano intersetam-se no ponto de coordenadas:(0.00;1.00;2.00).
```

Figura 107 – Execução do programa Interseções para a opção de interseção entre uma reta e um plano.

Por último, os alunos analisaram a interseção entre dois planos, sendo que para a equação dos planos consideraram a equação geral do plano ($x1(x) + y1(y) + z1(z) + d1 = 0$).

Começaram por idealizar um sistema composto pelas duas equações dos planos. Como se trata de um sistema de duas equações com três incógnitas decidiram atribuir um valor a x e resolver o sistema. Assim, determinaram as coordenadas de um ponto comum aos dois planos. Repetiram o processo e obtiveram um segundo ponto em comum. Através desses dois pontos determinaram um vetor diretor da reta e concluíram com a determinação da equação vetorial da reta resultante da interseção entre os dois planos.

Em termos de código, "se os vetores normais aos planos forem colineares, então [os planos] são paralelos ou coincidentes". Para efetuar esta verificação os alunos efetuaram um raciocínio parecido com o que fizeram na interseção entre duas retas no espaço. Em particular, se d2*k==d1, "então os planos são coincidentes". Se, pelo contrário, apenas a primeira se verificar ($Vx2 * k == Vx1 \; and \; Vy2 * k == Vy1 \; and \; Vz2 * k == Vz1$), então são estritamente paralelos. Caso os vetores não sejam colineares, então o programa procede ao cálculo do ponto de interseção através das fórmulas deduzidas.

Na figura 108 encontra-se um exemplo de execução do programa para esta última opção.

```
Que interseção queres determinar?
pp
Escreve as equações dos planos na forma:
a(x)+b(y)+c(z)+d=0
coordenada a vetor do plano 1 - 2
coordenada b vetor do plano 1 - 1
coordenada c vetor do plano 1 - -1
coordenada d do plano 1 - 1
coordenada a vetor do plano 2 - 1
coordenada b vetor do plano 2 - -1
coordenada c vetor do plano 2 - 3
coordenada d do plano 2 - 0
Os planos intersetam-se na reta de equação: (x;y;z)=(1;-5.00;-2.00)+k(1;-3.50;-1.50).
```

Figura 108 – Execução do programa Interseções para a opção de interseção entre dois planos.

O programa desenvolvido por estes dois alunos, com 249 linhas de código, é ambicioso, e, pelas diversas condições que estão envolvidas, os alunos têm que saber muito bem quais são as posições relativas entre reta/reta, reta/plano e plano/plano, bem como quais são as possíveis relações entre os vetores diretores das retas envolvidas e os vetores normais aos planos.

O problema complica-se em termos de análise quando existem uma ou mais coordenadas dos vetores implicados que é nula. No entanto, é um problema muito interessante e que possibilita o desenvolvimento do pensamento computacional, estando perfeitamente enquadrado nos programas de Matemática A do ensino secundário.

6.12.3 Processos utilizados e fontes procuradas

Para a execução do trabalho de projeto, estes dois alunos efetuaram "pesquisas na internet e no youtube, assim como no caderno de matemática do 11.º ano".

Consideraram que os guiões de trabalho utilizados nas três sessões de introdução ao *Python* "foram úteis e auto-explicativos", destacando os exemplos práticos que aí se encontram. Referem, no entanto, que deveriam ter uma linguagem “mais simplificada”.

6.12.4 Dificuldades sentidas e raciocínio matemático envolvido

Estes alunos reforçaram "que a maior parte dos conhecimentos já [tinha sido] aprendido através dos guiões e da prática". Realçam que necessitaram de procurar uma maior variedade de sinais e mais exemplos de aplicação.

Consideraram que "o nível de dificuldade dos guiões foi aumentando (…), à medida que uma nova tarefa nos era proposta, já tínhamos mais conhecimento", e como tal não destacam nenhuma tarefa como sendo mais fácil ou mais difícil.

Para estes alunos o trabalho efetuado permitiu o desenvolvimento do raciocínio matemático ao “tentar descobrir como escrever todas as condições de modo que não houvesse

erros no programa", nomeadamente referem que na descoberta dessas condições "estaremos a desenvolver o nosso raciocínio lógico".

Na resposta ao questionário final, os alunos consideraram o trabalho útil por “nos ajudar a desenvolver o raciocínio lógico e achamos útil aprender a trabalhar com Python”.

No relatório, e relativamente ao seu trabalho de projeto, referem que a interseção entre elementos no espaço tem diversas aplicações no mundo real, apresentando como exemplo saber quando dois projeteis com rotas aproximadamente retilíneas se irão intersetar, ou saber se meteoritos e asteroides irão embater em algum planeta através do cálculo das suas rotas. Na economia, referem que este tipo de problemas tem a sua utilidade pois permitem “determinar quando é que os lucros vão ultrapassar os prejuízos (…)”. Por fim, consideram que o programa desenvolvido pode ter a sua utilidade “na arquitetura e nas engenharias, nomeadamente na construção de objetos”.

6.12.5 Conclusões

Ao longo do trabalho efetuado, os alunos foram aprimorando o seu conhecimento relativamente à matemática e à programação em *Python*. Referem ter gostado do desafio que passou pela dedução de fórmulas, mas, principalmente, o facto de terem de descobrir a melhor forma de as introduzir no programa constituiu-se como uma tarefa nova.

Em termos metodológicos referiram que o facto de terem de efetuar um relatório e um vídeo os “ajudou a sintetizar a informação do trabalho”. Em suma, gostaram do desafio proposto e atribuíram uma avaliação muito positiva às aulas de *Python* que tiveram.

Aparentemente estes alunos desenvolveram as 5 práticas de PC, nomeadamente a abstração; a decomposição, ao considerar no programa os diversos casos de interseção entre retas, reta e plano e plano/plano; o reconhecimento de padrões, através da análise de detalhes entre os objetos considerados (retas e planos); a algoritmia, uma vez que tiveram que idealizar e estruturar as diversas etapas para a construção do código; e, depuração, pois, como referem os alunos, este programa teve bastantes erros de execução que foram sendo corrigidos com uma depuração fina do código que permitiu resolver falhas ou imprecisões (Wing, 2006; Espadeiro, 2022).

Relativamente ao raciocínio desenvolvido este centra-se no dedutivo e no indutivo, sendo que o abdutivo pode ser igualmente potenciado com o estabelecimento de conjeturas caso os alunos explorem o programa, utilizando-o em aplicações concretas, algumas delas referidas pelos alunos na secção anterior.

7

CONCLUSÕES

Neste capítulo pretende-se estabelecer as conclusões tiradas da análise do trabalho de campo e dos trabalhos efetuados pelos alunos em cada estudo de caso..

7.1 Resultados da investigação

Esta investigação, levada a cabo ao longo de todo o ano letivo de 2021/22, resultou de um trabalho de interdisciplinaridade entre as disciplinas de Matemática A e de Aplicações Informáticas B, tendo o seu maior foco no trabalho de campo que decorreu no 2.º período.

Foi uma experiência de ensino aliciante e gratificante, não só pelos resultados obtidos pelos alunos que demonstraram uma grande predisposição para aprender e saber mais, mas também pela dedicação e resiliência colocados por estes na execução das tarefas que lhes foram propostas, quer nas aulas através dos guiões, quer na execução do projeto final.

Tendo por base as noções de PC, expressão atribuída a Wing (2006), e de Raciocínio Matemático, com raízes na Grécia Antiga com o desenvolvimento do raciocínio dedutivo, nomeadamente da lógica, foram efetuadas três sessões de trabalho com os 28 alunos de uma turma de 12.º ano. No final foi proposto aos alunos desta turma que desenvolvessem um trabalho de projeto em que tiveram de utilizar o computador, nomeadamente a linguagem de programação *Python,* e um tema de Matemática desenvolvido na disciplina de Matemática A ao longo do ensino secundário.

Tal como se assinala no capítulo anterior, das 14 equipas de dois alunos formadas para o desenvolvimento deste trabalho, verificou-se que 12 equipas, constituindo os 12 estudos de caso relatados, desenvolveram um programa nos moldes pretendidos, sendo que o único trabalho que se desviou deste princípio foi o trabalho executado pelas alunas do estudo de caso 7, que mesmo assim efetuaram um projeto de interesse para a generalidade dos alunos neste nível de ensino que pretendam ingressar no ensino superior.

Não esquecendo que este é um estudo de natureza qualitativa, de cariz empírico, em que se pretendeu implementar estratégias de ensino e de aprendizagem, recorrendo a um ambiente de resolução de problemas, sendo que os alunos foram incentivados a aprender e

a consolidar conceitos matemáticos de uma forma prática e dinâmica, o investigador colocou inicialmente as seguintes três questões de investigação:

- Como construir tarefas que incorporem o Pensamento Computacional (PC), potenciadoras do raciocínio matemático no Ensino Secundário?
- Como se processa a aprendizagem da Matemática quando se implementam tarefas que envolvem o PC?
- De que forma se materializam modelos de ensino e aprendizagem, no domínio da Matemática, recorrendo à linguagem de programação *Python*?

Tal como foi relatado no capítulo 5, foram várias as tarefas propostas aos alunos que lhes permitiram ter uma iniciação ao trabalho com a linguagem *Python*, sendo que algumas delas poderiam ser perfeitamente colocadas a alunos no 10.º ano, como as relativas a geometria analítica no plano (distância entre dois pontos no plano e determinação do declive da ordenada na origem de uma equação reduzida da reta no plano) ou a referente à construção de um programa com a fórmula resolvente do 2.º grau, solicitada na 2.ª sessão de trabalho.

Já a tarefa relativa ao problema das dobragens de uma folha de papel, dada no 3.º guião de trabalho, poderia facilmente servir de introdução ao tema de sucessões do 11.º ano, permitindo o desenvolvimento da abstração, da algoritmia e eventualmente da depuração, mas também o reconhecimento de padrões e de regularidades, caraterísticas fundamentais do pensamento computacional (Wing, 2006; Espadeiro, 2022; Albuquerque, 2022).

Tendo sempre por base um processo evolutivo e sequencial, como é destacado por vários alunos no capítulo anterior, nos guiões apresentados nas três sessões de trabalho houve a preocupação de introduzir tarefas com significado para os alunos e de acordo com os conteúdos lecionados em Matemática A.

Nas duas últimas tarefas propostas nos guiões são solicitados dois programas um pouco mais complexos, baseados em processos iterativos e recorrendo a ciclos de repetição. O primeiro programa utiliza o método da bisseção, em que é fornecido o algoritmo e o código, resultando da aplicação do Corolário do Teorema de Bolzano-Cauchy, estudado pelos alunos nas aulas do 12.º ano, e o segundo programa, em que é solicitada a produção de um código que permite a aplicação do método de *Newton-Raphson*, do qual os alunos não tinham qualquer conhecimento, mas em que lhes é solicitado alguma pesquisa recorrendo à *internet*.

Relativamente a esta última tarefa, e embora alguns dos alunos a tenham executado, talvez tivesse sido útil fornecer mais informação e efetuar mais uma sessão para a sua

exploração, uma vez que implicava alguma pesquisa de informação sobre o conceito aplicado na execução do programa.

O facto de se terem apresentadas tarefas com conteúdos matemáticos em que inicialmente, para cada proposta de tarefa, era apresentado o algoritmo e o programa em código permitiu que os alunos fossem interiorizando e consolidando os diversos conceitos, quer matemáticos, quer de utilização da linguagem *Python*.

Em resposta à primeira questão de investigação esta encontra solução no empenho demonstrado pelos alunos na resolução das tarefas propostas nos guiões e no trabalho que estes mesmos desenvolveram na concretização do trabalho de projeto, como está patente na análise efetuada a cada um dos estudos de caso no capítulo anterior.

No que concerne à segunda questão de investigação formulada, houve a preocupação de integrar o currículo de Matemática A nos trabalhos desenvolvidos no decurso das três sessões de trabalho e isso é visível nas escolhas efetuadas pelos alunos dos seus trabalhos de projeto e nas respostas dadas por estes ao questionário final.

Nenhum dos alunos considerou que o trabalho desenvolvido não fosse útil, embora alguns considerassem que este tipo de trabalho não desenvolvia o raciocínio matemático, acabando invariavelmente por caírem numa contradição ao afirmar que tiveram que apresentar um algoritmo que fosse lógico e sequenciado.

Todos os alunos inquiridos consideraram os guiões fornecidos como sendo úteis, quer para a aprendizagem da linguagem *Python*, quer para o entendimento matemático requerido em cada uma das tarefas propostas.

Revelou-se igualmente útil o facto de ter-se solicitado aos alunos que construíssem um relatório e um vídeo do seu projeto pois só com estes dois elementos foi possível os alunos apropriarem-se adequadamente do objeto do seu trabalho e assim desenvolver competências tão valiosas à Matemática como sejam a criação de espírito crítico e de autoconfiança, além de que ao fornecer estes trabalhos aos restantes alunos através da equipa do Microsoft Teams® se tenha permitido que todos explorassem os diversos programas desenvolvidos.

Talvez tivesse sido igualmente benéfico ter solicitado aos alunos que apresentassem o seu projeto aos restantes alunos da turma pois desta forma ter-se-ia fomentado igualmente a comunicação matemática entre pares e refletido sobre os diversos programas efetuados e possíveis implicações, criando assim um espaço de discussão e interajuda, com possíveis melhorias do produto final, no entanto, e embora tal situação estivesse prevista, não foi possível levá-la à prática pois os trabalhos foram entregues num prazo posterior ao limite fixado inicialmente.

De qualquer das formas, em todas as situações analisadas nos 12 estudos de caso. a resolução de problemas em aberto, sob a forma de tarefas, esteve sempre presente e permitiu incentivar os alunos na utilização dos três tipos de raciocínio, em especial o dedutivo e o indutivo (Ponte et al., 2020), permitindo que os alunos em cada um dos grupos formados assumissem um papel relevante na interpretação e análise das questões e na conceção e concretização de estratégias de resolução para os problemas formulados por estes.

Os trabalhos finais realizados pelos 12 grupos revelaram-se igualmente bastante interessantes e apesar de alguns trabalhos denotarem algumas insuficiências em termos de abstração e de algoritmia, por os alunos não terem conseguido dominar na perfeição o PC, foi uma tarefa que possibilitou a criação de autonomia e de autoconfiança nos alunos.

Relativamente à terceira questão, a introdução da linguagem de programação *Python* serviu aqui de pretexto para fomentar o PC e potenciar o raciocínio matemático, sendo que nos diversos projetos desenvolvidos e analisados nos estudos de caso, os alunos tiveram que saber representar e dar significado a cada conteúdo matemático escolhido.

A resolução de cada um dos problemas matemáticos referenciados por cada grupo de alunos, além de começar pela formulação de questões, passou igualmente pelo estabelecimento de conjeturas e pelo desenvolvimento de estratégias de resolução que permitiu ou não a confirmação das conjeturas formuladas, tendo-se revelado nos diversos passos de código efetuado pelos alunos, sendo que em certos projetos esse trabalho foi inclusivamente mais vincado do que noutros, quer no próprio programa desenvolvido, quer nos subprodutos apresentados, como sejam o vídeo e o relatório.

No entanto, e tal como referem Ponte et al. (2020), a abordagem exploratória desenvolvida pela generalidade dos alunos em cada grupo de trabalho permitiu "enfatizar a construção de conceitos, a modelação de situações e também a utilização de definições e propriedades de objetos matemáticos para os alunos realizarem raciocínios (…)." (p. 10)

A metodologia de trabalho utilizada que se materializou num trabalho de projeto, precedida de três sessões de trabalho de apresentação do *Python*, permitiu que os alunos desenvolvessem processos de raciocínio matemático, que estes fossem ligados aos seus conhecimentos e capacidades, possibilitando igualmente uma reflexão sobre as suas práticas e aumentando desta forma os seus níveis de autonomia, de autoestima e de sentido crítico.

A utilização do *Python*, quer através da construção de programas, como se pretendeu fazer ao longo deste trabalho, quer mesmo através da análise de diversos códigos, constituiu-se assim como uma mais-valia para que os alunos conseguissem perceber melhor os

processos envolvidos na construção do raciocínio matemático e ter assim plena consciência da sua capacidade de pensar e raciocinar.

7.2 Reflexão final

Todo este trabalho efetuado com os alunos pode e deve ser concretizado em aula desde o início do secundário, não sendo imprescindível que existam computadores pessoais disponíveis para cada um dos alunos da turma em sala de aula, bastando somente que cada discente tenha na sua posse um *tablet* ou um *smartphone* com ligação à *internet* ou uma calculadora gráfica que possua a linguagem de programação *Python* incorporada.

O desenvolvimento do PC utilizando a linguagem *Python* pode e deve ser persistente e continuado, e ser efetuado ao longo de todo o ano letivo, em cada um dos três anos de escolaridade do ensino secundário, com todos os temas que forem objeto de análise na disciplina de Matemática.

Torna-se aqui fundamental escrutinar bem a metodologia de trabalho que se adota em aula com o conjunto-turma, quer este trabalho seja feito individualmente ou em pequenos grupos. Os conceitos de *Python* podem ser lecionados de forma gradual, à medida que forem sendo necessários no decurso da lecionação de um determinado tema matemático, sempre com recurso a programas já construídos e outros por construir, quer sob a forma de algoritmo, quer fornecendo o código para que o aluno consiga compreender as diferenças/semelhanças entre os dois tipos de representação.

Por outro lado, é importante que seja dada a possibilidade ao aluno de efetuar pequenas alterações no código do programa de forma a atingir determinado objetivo. Sugere-se igualmente que primeiro sejam dados os conceitos referidos no guião 1 (Anexo III) e posteriormente introduzir os comandos relacionados com a função condicional (if-elif-else). Por último, devem ser introduzidos os ciclos de repetição (*FOR* e *WHILE*), estando na base destas funções o reconhecimento de padrões e de regularidades.

De realçar que não se pretende transformar a Matemática numa disciplina de programação. O objetivo é que os alunos reconheçam o PC e saibam resolver problemas de uma forma encadeada e natural, tendo por base as 5 práticas preconizadas por este tipo de pensamento (Wing, 2006; Espadeiro, 2022).

A propósito, e já depois destes alunos terem efetuado o seu trabalho de projeto, convidei dois dos alunos deste grupo de trabalho, nomeadamente os alunos F_8 *e* H_6, para que, em conjunto com outros dois alunos de outra turma participassem no *8th Annual 2022 International Mathematical Modeling Challenge* (*IM2C*). O tema proposto a concurso foi a determinação

de quais os métodos mais eficazes para o embarque e o desembarque de passageiros de um avião, atendendo a diferentes configurações de aeronaves e diferentes lotações das mesmas.

Recorrendo a um simulador desenvolvido em linguagem *Python,* e adaptando o código a cada uma das situações pretendidas, os alunos simularam e compararam diferentes métodos de embarque em cada um dos casos referidos, bem como efetuaram um relatório em que descreveram os resultados obtidos. A classificação obtida por estes alunos foi de 'participação com sucesso'[11] entre várias equipas nacionais e internacionais.

[11] 'Sucessful Participant'

REFERÊNCIAS BIBLIOGRÁFICAS

Ackermann, E. (2001). Piaget´s constructivism, Papert´s constructivism: What´s the difference. *Future of Learning Group Publication*, 5(3), 438-449.

Albuquerque, C. (2022). Pensamento Computacional e Matemática. *Educação e Matemática*, *162*, 31-38.

Bivar, A., Grosso, C., Oliveira, F., Timóteo, C., & Loura, L. (2014). *Programa e Metas curriculares de Matemática A, Ensino Secundário*. Lisboa: Ministério da Educação e Ciência.

Bocconi, S., Chioccariello, A., Dettori, G., Ferrari, A., & Engelhardt, K. (2016). *Developing computational thinking in compulsory education – Implications for policy and practice*. Luxembourg: Publications Office of the European Union.

Bogdan, R. e Biklen, S. (1994). *Investigação qualitativa em Educação: Uma introdução à teoria e aos métodos*. Porto: Porto Editora.

Campos, A. de (2006). Poesias de Álvaro de Campos. In *Obras Completas de Fernando Pessoa*. Lisboa: Editorial Nova Ática.

Canavarro, A., Mestre C., Gomes D., Santos, E., Santos, L., Brunheira, L., Vicente, M., Gouveia, M., Correia, P., Marques, P., & Espadeiro, R. (2021). *Aprendizagens Essenciais de Matemática – 7.º Ano*. Lisboa: DGE. Obtido de https://www.dge .mec.pt/sites/default/files/Curriculo/Aprendizagens_Essenciais/3_ciclo/aemat_7a_2021 -08-19.pdf

Carvalho e Silva, J., Fonseca, G., Martins, A., & Lopes, I. (2001). *Matemática A, 10.º Ano*. Lisboa: Departamento do Ensino Secundário, Ministério de Educação.

Carvalho e Silva, J., Canavarro, A., Albuquerque, C., Mestre, C., Martins, H., Almiro, J., Santos, L., Gabriel, L., Seabra, O., & Correia, P. (2020). *Recomendações para a melhoria das aprendizagens dos alunos em Matemática*. Lisboa: DGE.

Carvalho e Silva, J. (2022). Pensamento Computacional no ensino em França. *Educação e Matemática, 162*, 53-54.

Carvalho e Silva, J. (2022a). *Os Algoritmos e o Pensamento Computacional*. Conferência realizada na Escola Secundária António Damásio, em Lisboa.

Christiansen, B. & Walther, G. (1986). Task and activity. In B. Cristiansen, A. G. Howson & M. Otte (Eds.), *Perspectives on mathematics education* (pp. 73-81). Dordrecht: D. Reidel.

[DGE] Direção Geral de Educação (2018). *Aprendizagens Essenciais de Matemática A – 12.º Ano*. Lisboa: DGE. Obtido de https://www.dge.mec.pt/sites/default/files/Curriculo/Aprendizagens_Essenciais/12_matematica_a.pdf

Domingos, A. (2003). *Compreensão de conceitos matemáticos avançados – A Matemática no início do superior.* Tese de Doutoramento. Lisboa: FCT-UNL.

Domingos, P. (2017). *A revolução do algoritmo mestre*. Queluz de Baixo: Letras & Diálogos.

Espadeiro, R. (2022). O Pensamento Computacional no currículo de Matemática. *Educação e Matemática*, *162*, 5-10.

Fey, J.T. (1991). Tecnologia e Educação Matemática – Uma Revisão de Desenvolvimento Recentes e Problemas Importantes. In J. P. Ponte (Org.), *O Computador na Educação Matemática. Cadernos de Educação e Matemática, n.º 2* (pp. 45-79). Lisboa: APM.

Flick, U. (2004). Introducción a la investigación cualitativa. Madrid: Morata.

Gilchrist, P.O., Alexander, A.B., Green, A.J., Sanders, F.E., Hooker, A.Q., & Reif, D.M. (2021). Development of a Pandemic Awareness STEM Outreach Curriculum: Utilizing a Computational Thinking Taxonomy Framework. *Educ. Sci. 11*, 109. Obtido de https://doi.org/10.3390/educsci11030109https://doi.org/10.3390/educsci11030109

Hamel, J. (1997). *Étude de cas et sciences sociales*. Paris: L'Harmattan.

Ludke, M., & André, M. E. (1986). *A. Pesquisa em educação : abordagens qualitativas*. São Paulo: EPU.

Mailund, T. (2021). *Introduction to Computational Thinking Problem Solving, Algorithms, Data Structures, and More*. New York: Apress Media.

Martins, G., Gomes, C., Brocardo, J., Pedroso, J., Carillo, J., Silva, L., Encarnação, M., Horta, M., Calçada, M., Nery, R., & Rodrigues, S. (2017). *Perfil dos alunos à saída da escolaridade obrigatória.* Lisboa: Direção-Geral da Educação, Ministério da Educação.

Martins, H. (2016). Tarefas para a Aprendizagem de Estatística no Ensino Secundário com Calculadora Gráfica. Tese de Mestrado. Lisboa: FCT-UNL.

Martins, H., & Domingos, A. (2018). Estatística no Secundário com Calculadora Gráfica. Caparica: Nova.fct Editorial.

Merriam, S. (1988). *The case study research in education*. San Francisco: Jossey-Bass.

[ME] Ministério da Educação (2018a). *Aprendizagens essenciais. 10.º ano, Matemática*. Obtido de http://www.dge.mec.pt/sites/default/files/Curriculo/Aprendizagens_Essenciais/10_matematica_a.pdf

[ME] Ministério da Educação (2018b). *Aprendizagens essenciais. 11.º ano, Matemática*. Obtido de http://www.dge.mec.pt/sites/default/files/Curriculo/Aprendizagens_Essenciais/11_matematica_a.pdf

[ME] Ministério da Educação (2018c). *Aprendizagens essenciais. 12.º ano, Matemática*. Obtido de http://www.dge.mec.pt/sites/default/files/Curriculo/Aprendizagens_Essenciais/12_matematica_a.pdf

[NCTM] National Council of Teachers of Mathematics (1991). *Normas para o Currículo e a Avaliação em Matemática Escolar*. Lisboa: APM.

[NCTM] National Council of Teachers of Mathematics (1994). *Normas profissionais para o ensino da Matemática (NCTM).* Lisboa: Associação de professores de matemática e Instituto de inovação Educacional.

[NCTM] National Council of Teachers of Mathematics (2007). *Princípios e Normas para a Matemática Escolar*. Lisboa: APM.

[NCTM] National Council of Teachers of Mathematics (2017). *Princípios para a ação – Assegurar a todos o sucesso em Matemática*. Lisboa: APM.

Negra, C., Martinho, E. & Martins, H. (2017). *Dimensões de Matemática A 12.º Ano*. Queluz de Baixo: Santillana

[NGSS] Next Generation Science Standards (2013). Appendix F-science and engineering practices in the NGSS. In *Next Generation Science Standards: For States, By States*. NGSS Lead States: Washington, DC, USA.

[OECD] Organisation for Economic Cooperation and Development (2019). *PISA 2018 Assessment and Analytical Framework*. Paris: OECD Publishing. Obtido de https://doi.org/10.1787/b25efab8-en

Papert, S. (1980). *Mindstorms – Children, Computers, and Powerful Ideas*. Basic Books.

Papert, S. (1991). O Computador na Educação Matemática – Série Cadernos de Educação e Matemática, número 2, organização de João Pedro da Ponte em *Ensinar crianças a serem matemáticos vs ensinar Matemática* (pp. 29-44). Lisboa: Associação de Professores de Matemática.

Ponte, J. P., Nunes, F., & Veloso, E. (1991). *Computadores no Ensino da Matemática*. Lisboa: APM e Pólo do Projeto Minerva do DEFCUL.

Ponte, J. P., Quaresma, M., & Pereira, J. M. (2020). Como desenvolver o raciocínio matemático na sala de aula? *Educação e Matemática*, *156*, 7-11.

Prensky, M. (2012). *From digital natives to digital wisdom: Hopeful essays for 21st century learning*. Corwin.

Resnick, M. (2008). Cultivando las semillas para una sociedad más creativa. *Actualidades Investigativas en Educación*, *1*(8), E-ISSN: 1409-4703. Obtido de https://www.redalyc.org/pdf/447/44780123.pdf

Stake, R. E. (1994). Case Studies. In N. Denzin Y. Lincoln, *Handbook of qualitative research* (pp. 236-247). Newsbury Park: Sage.

Taylor, S., & Bogdan, R. (1984). *Introduction to Quantitative Research Methods: The Search for Meanings*. Wiley.

Tesch, R. (1990). *Qualitative Research: Analysis Types and Software Tools*. London: RoutledgeFalmer.

Vásquez. R. R., & Angulo, R. F. (2003). *Introducción a los estudios de casos. Los primeros-contactos con la investigación etnográfica*. Málaga: Ediciones Aljibe.

Yin, R. (2005). *Estudo de Caso. Planejamento e Métodos*. Porto Alegre: Bookman.

Yin, R. (2012). *Applications of case study research*. Beverly Hills, CA: Sage Publishing.

Wing, J. (2006). Computational Thinking. *Communications of the ACM*, *49*(3), 33-35. Obtido de https://doi.org/10.1145/1118178.1118215

I

Anexo - Autorizações e Comunicações

Exma. Sra. Diretora,

[redacted]

Eu, Helder Manuel da Conceição Barão Martins, professor do grupo 500, venho por este meio solicitar a possibilidade de concretizar, na turma [redacted] do 12.º ano da [redacted], no ano letivo 2021/2022, o projeto de investigação em educação matemática intitulado "Utilização da linguagem Python para desenvolver o pensamento computacional no Ensino Secundário". Este projeto integra-se no âmbito do Mestrado em Ciências da Educação, na especialidade de Tecnologias no Ensino das Ciências, Tecnologias, Engenharia e Matemática (CTEM), da Faculdade de Ciências e Tecnologia da Universidade Nova de Lisboa.

Pretende-se com este projeto introduzir o pensamento computacional no ensino secundário, nomeadamente a alunos do 12.º ano, nas disciplinas de Matemática A e Aplicações Informáticas B, recorrendo à utilização da linguagem Python inserida num computador e nas calculadoras gráficas, e analisando de que forma este tipo de pensamento pode potenciar a aprendizagem da Matemática.

As principais formas de recolha de dados para a concretização do projeto serão: 1) observação direta das aulas; 2) trabalhos produzidos dentro da sala de aula pelos alunos; e, 3) entrevistas aos alunos que constituirão os estudos de caso. A recolha de dados recorrerá igualmente aos registos criados pelos alunos no computador e nas calculadoras TI-*n*spire *CX* e Casio fx-CG50.

Informo, ainda, que irá ser solicitada autorização aos encarregados de educação, de forma que os seus educandos possam participar neste projeto de investigação, sendo sempre salvaguardado o anonimato, quer dos alunos, quer da escola.

Com os melhores cumprimentos,

Lisboa, 7 de fevereiro de 2022

O investigador,

_____

(Helder Martins)

Exmo. Sr. Coordenador,

Departamento de Matemática e Ciências Experimentais do [redacted]

Eu, Helder Manuel da Conceição Barão Martins, professor do grupo 500, venho por este meio comunicar a minha intenção de concretizar, na turma [redacted] do 12.º ano desta escola, no ano letivo 2021/2022, o projeto de investigação em educação matemática intitulado "Utilização da linguagem Python para desenvolver o pensamento computacional no Ensino Secundário". Este projeto integra-se no âmbito do Mestrado em Ciências da Educação, na especialidade de Tecnologias no Ensino das Ciências, Tecnologias, Engenharia e Matemática (CTEM), da Faculdade de Ciências e Tecnologia da Universidade Nova de Lisboa.

Pretende-se com este projeto introduzir o pensamento computacional no ensino secundário, nomeadamente a alunos do 12.º ano, nas disciplinas de Matemática A e Aplicações Informáticas B, recorrendo à utilização da linguagem Python inserida num computador e nas calculadoras gráficas, e analisando de que forma este tipo de pensamento pode potenciar a aprendizagem da Matemática.

As principais formas de recolha de dados para a concretização do projeto serão: 1) observação direta das aulas; 2) trabalhos produzidos dentro da sala de aula pelos alunos; e, 3) entrevistas aos alunos que constituirão os estudos de caso. A recolha de dados recorrerá igualmente aos registos criados pelos alunos no computador e nas calculadoras TI-*n*spire *CX* e Casio fx-CG50.

Informo, ainda, que foi solicitada autorização à direção da escola, a qual foi concedida, e que brevemente irei solicitar autorização aos encarregados de educação, de forma a que os seus educandos possam participar neste projeto de investigação, sendo sempre salvaguardado o anonimato, quer dos alunos, quer da escola.

Com os melhores cumprimentos,

Lisboa, 9 de fevereiro de 2022

O investigador,

_____

(Helder Martins)

Exma. Sra. Representante,

Grupo de Recrutamento de Matemática - 500 da [redacted]

Eu, Helder Manuel da Conceição Barão Martins, professor do grupo 500, venho por este meio comunicar a minha intenção de concretizar, na turma [redacted] do 12.º ano desta escola, no ano letivo 2021/2022, o projeto de investigação em educação matemática intitulado "Utilização da linguagem Python para desenvolver o pensamento computacional no Ensino Secundário". Este projeto integra-se no âmbito do Mestrado em Ciências da Educação, na especialidade de Tecnologias no Ensino das Ciências, Tecnologias, Engenharia e Matemática (CTEM), da Faculdade de Ciências e Tecnologia da Universidade Nova de Lisboa.

Pretende-se com este projeto introduzir o pensamento computacional no ensino secundário, nomeadamente a alunos do 12.º ano, nas disciplinas de Matemática A e Aplicações Informáticas B, recorrendo à utilização da linguagem Python inserida num computador e nas calculadoras gráficas, e analisando de que forma este tipo de pensamento pode potenciar a aprendizagem da Matemática.

As principais formas de recolha de dados para a concretização do projeto serão: 1) observação direta das aulas; 2) trabalhos produzidos dentro da sala de aula pelos alunos; e, 3) entrevistas aos alunos que constituirão os estudos de caso. A recolha de dados recorrerá igualmente aos registos criados pelos alunos no computador e nas calculadoras TI-*n*spire *CX* e Casio fx-CG50.

Informo, ainda, que foi solicitada autorização à direção da escola, a qual foi concedida, e que brevemente irei solicitar autorização aos encarregados de educação, de forma que os seus educandos possam participar neste projeto de investigação, sendo sempre salvaguardado o anonimato, quer dos alunos, quer da escola.

Com os melhores cumprimentos,

Lisboa, 9 de fevereiro de 2022

O investigador,

_____

(Helder Martins)

Exmo. Sr. Diretor,

[redacted]

Eu, Helder Manuel da Conceição Barão Martins, professor do grupo 500, venho por este meio comunicar a minha intenção de concretizar, na turma [redacted] do 12.º ano desta escola, no ano letivo 2021/2022, o projeto de investigação em educação matemática intitulado "Utilização da linguagem Python para desenvolver o pensamento computacional no Ensino Secundário". Este projeto integra-se no âmbito do Mestrado em Ciências da Educação, na especialidade de Tecnologias no Ensino das Ciências, Tecnologias, Engenharia e Matemática (CTEM), da Faculdade de Ciências e Tecnologia da Universidade Nova de Lisboa.

Pretende-se com este projeto introduzir o pensamento computacional no ensino secundário, nomeadamente a alunos do 12.º ano, nas disciplinas de Matemática A e Aplicações Informáticas B, recorrendo à utilização da linguagem Python inserida num computador e nas calculadoras gráficas, e analisando de que forma este tipo de pensamento pode potenciar a aprendizagem da Matemática.

As principais formas de recolha de dados para a concretização do projeto serão: 1) observação direta das aulas; 2) trabalhos produzidos dentro da sala de aula pelos alunos; e, 3) entrevistas aos alunos que constituirão os estudos de caso. A recolha de dados recorrerá igualmente aos registos criados pelos alunos no computador e nas calculadoras TI-*n*spire *CX* e Casio fx-CG50.

Informo, ainda, que foi solicitada autorização à direção da escola, a qual foi concedida, e que brevemente irei solicitar autorização aos encarregados de educação, de forma que os seus educandos possam participar neste projeto de investigação, sendo sempre salvaguardado o anonimato, quer dos alunos, quer da escola.

Com os melhores cumprimentos,

Lisboa, 9 de fevereiro de 2022

O investigador,

_____

(Helder Martins)

Exmo.(a) Sr.(a) Encarregado(a) de Educação

Eu, Helder Manuel da Conceição Barão Martins, professor de Matemática do Quadro de Nomeação Definitiva da [redacted], venho por este meio solicitar autorização para a participação do seu educando no projeto de investigação em educação matemática intitulado "Utilização da linguagem Python para desenvolver o pensamento computacional no Ensino Secundário".

Este projeto integra-se no âmbito do Mestrado em Ciências da Educação, na especialidade de Tecnologias no Ensino das Ciências, Tecnologias, Engenharia e Matemática (CTEM), da Faculdade de Ciências e Tecnologia da Universidade Nova de Lisboa, que estou a desenvolver.

Pretende-se com este projeto introduzir o pensamento computacional no ensino secundário, nomeadamente a alunos do 12.° ano, nas disciplinas de Matemática A e Aplicações Informáticas B, recorrendo à utilização da linguagem Python inserida num computador e nas calculadoras gráficas, e analisando de que forma este tipo de pensamento pode potenciar a aprendizagem da Matemática.

A concretização do projeto implicará a recolha de dados ao longo do ano letivo 2021/2022, acompanhando os alunos de 12.° ano, ocorrendo em momentos em que estejam a ser lecionados os domínios de Funções Reais de Variável Real, Trigonometria/Funções Trigonométricas e Funções Exponenciais/Funções logarítmicas (2.° e 3.° períodos). De modo a obter informação mais detalhada está ainda prevista a recolha de alguns trabalhos escritos dos alunos, assim como a realização de entrevistas a alguns alunos, sendo os Encarregados de Educação dos mesmos posteriormente contactados.

Mais declaro que será sempre preservado o anonimato, quer dos alunos, quer da escola, sendo que todos e quaisquer dados recolhidos serão mantidos em arquivos por um período indefinido, e usados única e exclusivamente para propósitos de investigação.

Convém referir que, deste trabalho, não resultará qualquer prejuízo para os alunos, não havendo lugar a qualquer alteração relativamente aos conteúdos programáticos nem às indicações metodológicas preconizadas nos programas de Matemática A e de Aplicações Informáticas B em vigor. A participação do seu educando não é obrigatória e este poderá sair do estudo em qualquer momento, sem qualquer consequência.

Para indicar se dá o seu consentimento à participação do seu educando na investigação, por favor preencha o formulário de resposta, em anexo, e devolva-o ao professor de Matemática do seu educando.

Note-se que os alunos que não desejam participar na investigação não serão objeto de recolha de quaisquer dados. No entanto, apreciaria a sua resposta positiva, uma vez que considero que este estudo pode contribuir para compreender como os professores podem ensinar Matemática de uma forma adequada a todos os alunos e que promoverá uma aprendizagem de qualidade. Consequentemente, considero que a participação do seu educando no estudo poderá revelar-se uma mais-valia para a sua aprendizagem, pois terá igualmente a oportunidade para refletir acerca da sua atividade matemática, desenvolvendo espírito crítico e autonomia.

Colocando-me ao dispor para quaisquer esclarecimentos adicionais e com os melhores cumprimentos,

Lisboa, 10 de fevereiro de 2022

O investigador,

_____

(Helder Martins)

Autorização

Eu, _____, Encarregado(a) de Educação do(a) aluno(a) _____, n.º ____, da turma ___, do 12.º ano da [redacted], declaro que tomei conhecimento dos objetivos do projeto de investigação em educação matemática intitulado "Utilização da linguagem Python para desenvolver o pensamento computacional no Ensino Secundário" e da necessidade de existirem alguns momentos de recolha de dados; e

autorizo ..... [ ]

não autorizo ..... [ ]

(colocar uma cruz no local pretendido)

a participação do meu educando no estudo, salvaguardando o respetivo anonimato.

____/____/_____

O(a) Encarregado(a) de Educação

_____

II

Anexo - Questionário inicial efetuado aos alunos

NOME: _____ TURMA: ____ N.º ____

1. Género: Masculino: ____ Feminino: ____ (coloca uma cruz)
2. Idade: _____
3. Naturalidade: _____
4. Local de residência: _____
5. Classificações obtidas na disciplina de Matemática: 10.º ano: ____ 11.º ano: ____
6. Classificações obtidas na disciplina de Português: 10.º ano: ____ 11.º ano: ____
7. O que significa para ti estudar Matemática?

8. O que gostas mais em Matemática? Porquê? E o que gostas menos? Porquê?

9. Qual é o tipo de atividades matemáticas que gostas mais de realizar na aula de Matemática? Porquê? E quais gostas menos? Porquê?

10. Gostas de efetuar programação? Achas que o fazes de forma eficiente?

11. Achas que o fato de efetuares programas te ajuda a compreender melhor a disciplina de Matemática?

FIM

III

Anexo - Guiões de trabalho utilizados na sala de aula

Guião 1 – Linguagem de programação Python
Matemática A

REPÚBLICA PORTUGUESA | EDUCAÇÃO

12.º Ano de escolaridade | Plataformas/Dados/Operadores/Funções | Turma: [redacted]

A linguagem de programação Python foi concebida pelo matemático Guido van Rossum, na Holanda, no final de 1989, e deve o seu nome aos humoristas ingleses *Monty Python*, criadores da série *Monty Python´s Flying Circus*, transmitida pela *British Bradcasting Corporation* (BBC) entre 1969 e 1974.
É uma das linguagens mais difundidas e usadas em todo o mundo, devido essencialmente a permitir uma fácil leitura do seu código e necessitar de poucas linhas de programação quando a comparamos com outras linguagens. É utilizada quer por programadores menos experientes, quer por especialistas, sendo usada essencialmente para processamento de textos, análise de dados e criação de páginas para a internet.

1. Plataformas

A linguagem Python pode ser utilizada em diferentes aplicativos/plataformas, nomeadamente em computadores/tablets (podes efetuar o download em https://www.python.org/downloads/), e nas calculadoras gráficas CASIO fx-CG50 (ver em https://www.casio-calculadoras.com) ou TI-84 Plus CE-T Python Edition/TI-Nspire™ CX II-T (ver em https://education.ti.com/pt).

2. Início

Ao criar e utilizar um programa em Python é possível distinguir dois tipos de ambientes: O modo **Editor (E)**, onde são escritas a sequência de instruções/comandos que se pretende que o programa execute, e o modo **Interpretador (I)** ou ***Shell*** que se encontra ligado ao Editor e permite a execução da sequência de comandos criados. Nas figuras 1 e 2, encontra-se o Editor (E) e o Interpretador (I), respetivamente, do programa Exemplo1.py, constituído somente pela instrução print().

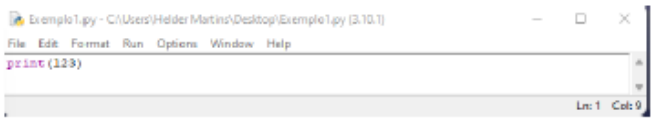


Fig. 1: Editor (E) do programa Exemplo1.py

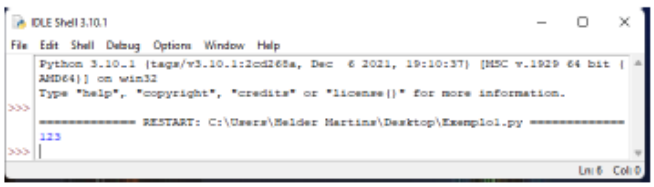


Fig. 2: Resultado da execução da função print() no Interpretador (I).

3. Dados

Existem vários tipos de dados. Os mais usados são números, inteiros (*int*), sem parte decimal, ou reais (*float*), com parte decimal[1]; valores lógicos, verdadeiro (*True*) ou falso (*False*); e, cadeias de texto (*strings*), sequências de carateres entre aspas. Para armazenar estes dados em memória utilizam-se variáveis (Figura 3).

4. Operadores

Os operadores de comparação como, por exemplo, ==, < e >=, são utilizados com todo o tipo de dados. Efetuar comparações como, por exemplo, a < b tem o mesmo significado que na

[1] O símbolo da vírgula (“,”) nos números com parte decimal, é substituída por ponto (“.”).

matemática: os termos são comparados respeitando a ordem. Ao efetuar uma comparação é criado um valor lógico (verdadeiro ou falso). Os operadores booleanos: and (e) e or (ou), seguem igualmente este princípio.
Na figura 3 encontram-se exemplos de números, valores lógicos e cadeias de texto, bem como de operadores. Para obter o valor das variáveis no Interpretador (I) utiliza-se a função print().

E:

```
a=2
x,y=3,4.02
nome1="Olá"
nome2=" Rui"
print("a =",a)
print(nome1+nome2)
print(y==4 or x*y<=12)
```

I:

```
a = 2
Olá Rui
False
```

Fig. 3: E e I com a introdução de números e cadeias de texto em Python[2].

A linguagem Python contem igualmente algumas funções de raiz que permitem realizar diversos tipos de operações, quer estas sejam aritméticas, quer doutro tipo matemático. Assim, é possível obter o valor absoluto de x (abs(x)); obter o simétrico de x (-x); converter uma variável x num número inteiro, retirando a parte decimal (int(x)); converter x num número real (float(x)); arredondar x com n casas decimais (round(x,n)); obter o menor ou o maior valor entre x e y (min(x,y) e max(x,y), respetivamente).
É igualmente possível obter o resultado da operacionalização de x com y, nomeadamente para a adição, subtração, multiplicação, divisão e potenciação (x+y, x-y, x*y, x/y, x**y, respetivamente). Consegue-se ainda obter o quociente da divisão inteira de x por y e o resto dessa divisão (x//y e x%y, respetivamente)[3]. Na figura 4, encontram-se alguns exemplos de aplicação.

E:

```
print(abs(-1.23))
print(int(-12.35))
print(float(7))
print(round(2.587,1))
print(min(1.4,2.5))
print(5*3)
print(11//3)
print(17%4)
```

I:

```
1.23
-12
7.0
2.6
1.4
15
3
1
```

Fig. 4: A apresentação em E e I de operações unárias ou entre vários valores[4].

O Python permite efetuar igualmente a formatação de números quando utilizamos a instrução print(), evitando assim que os números apresentados no Interpretador (I) tenham um número excessivo de casas decimais, permitindo igualmente que as variáveis definidas apareçam numa determinada posição de uma frase que se pretende colocar no Interpretador. A figura 5 apresenta alguns exemplos.

[2] Sugere-se que executes na plataforma escolhida as instruções referidas no Editor de forma a confirmares o que está no Interpretador.
[3] A linguagem Python segue, em termos de prioridades das operações, o que está estabelecido matematicamente.
[4] Ver nota ([2]).

E:

```
a,b=2.357,1.2
print("O valor de a é {}.".format(a))
print("Os valores de a e b são {} e
{}.".format(a,b))
print("O valor de a arredondado com uma casa
decimal é {:.1f}".format(a))
```

I:

```
O valor de a é 2.357.
Os valores de a e b são 2.357 e 1.2.
O valor de a arredondado com uma casa
decimal é 2.4
```

Fig. 5: Exemplos de utilização da instrução format() no print()[5].

Por vezes, quando se trabalha com cadeias de texto, é útil saber o número de carateres da sequência x (len(x)); ou o carater da sequência x que está na ordem k (x[k-1]) (Figura 6).

E:

```
a="Python"
print(len(a))
print(a[3])
```

I:

```
6
h
```

Fig. 6: Exemplos de utilização de cadeias de texto[6].

É igualmente necessário importar bibliotecas (ou módulos) para o programa para se conseguir utilizar determinado tipo de funções. Nestes casos utiliza-se o comando *import* *
Um dos módulos mais utilizados é a biblioteca *math*, que contem algumas funções importantes, como sejam a raíz quadrada (sqrt()); as funções trigonométricas (sen(), cos() ou tan(), por exemplo); ou constantes matemáticas como, por exemplo, o pi (π). Na figura 7 encontra-se um exemplo de utilização desta biblioteca[7].

E:

```
from math import *
print(sqrt(3))
print(cos(pi/6))
```

I:

```
1.7320508075688772
0.8660254037844387
```

Fig. 7: Exemplo de utilização da biblioteca *math*[8].

5. Introdução de dados interactivamente

A leitura de dados de forma interativa no Interpretador implica a introdução no Editor da função input(). Esta função permite a criação de uma variável pela leitura de um símbolo ou sequência de símbolos, introduzido(s) pelo utilizador, que poderá ser convertida em número, inteiro ou real, ou permanecer como cadeia de texto, conforme a instrução do Editor (Figura 8).

E:

```
nome = input("nome? ")
n = int(input("n= "))
pi = float(input("pi= "))
```

I:

```
nome? Ana
n= 6
pi= 3.14
```

Fig. 8: Leitura interativa de dados e respetivo comportamento nos ambientes E e I[9].

[5] Ver nota (2).
[6] Ver nota (2).
[7] Caso a biblioteca *math* não seja importada, aparece a mensagem de erro em I: name 'sqrt' is not defined
[8] Ver nota (2).
[9] Ver nota (2).

6. Funções

Uma das grandes vantagens de utilizar a linguagem Python reside na utilização de funções, que não são mais do que estruturas que agrupam uma sequência de instruções.
Para caraterizar uma função em Python utiliza-se a instrução *def* seguida do nome da função, separada da palavra *def* por um espaço, e o(s) argumento(s) entre parenteses, separados por vírgulas. Esta linha termina com dois pontos (":"), separada do segundo parenteses por um espaço (a introdução dos dois pontos "força" a indentação[10] (um avanço à esquerda) nas linhas seguintes) Por último, e nas linhas seguintes, tem que se incluir a sequência de instruções necessárias para atingir o objetivo definido[11] (Figura 9).

```
def nome(argumento1,argumento2,...) :
--->|sequência de instruções
```

Fig. 9: Esquema de utilização do comando *def*.

Apresenta-se, em seguida, o exemplo de um programa definido por uma função que permite obter a distância entre dois pontos no plano, dadas as suas coordenadas (Figura 10).

E:

```
from math import *
def dist(xA,yA,xB,yB) :
    d=sqrt((xA-xB)**2+(yA-yB)**2)
    return(d)
```

I:

```
dist(2,3,-1,7)
5.0
```

Fig. 10: Editor e Interpretador do programa distância.py[12].

Uma função possibilita a decomposição do problema inicialmente proposto em subcategorias e assim evitar a repetição de instruções. Uma dada função pode ser "chamada" durante o programa tantas vezes quanto for necessário, podendo não ter argumentos e ser "chamada" noutro programa, bastando para tal inserir uma instrução da função com valores nos argumentos no 2º programa.

7. Exercício

Escreva um programa em Python que permita devolver o declive e a ordenada na origem da equação reduzida de uma reta, dadas as coordenadas de dois pontos no plano.
A resposta deverá conter duas linhas, uma com o declive e outra com a ordenada na origem.
Nome do ficheiro: reta.py

[10] A indentação em Python tem que ser respeitada sob prejuízo do programa não funcionar.
[11] Na sequência de instruções pode ser utilizada a instrução *return*(), como forma de obter um valor como resposta.
[12] Ver nota ([2]).

No Guião 1 verificaste onde podes encontrar a linguagem Python, bem como se podem introduzir dados no programa, diretamente no Editor ou interactivamente no Interpretador. Conheceste igualmente o que são operadores e ficaste a saber o que são funções em Python.
Neste 2º Guião ficarás a conhecer o que é uma função condicional.

1. Algoritmo

Entende-se por algoritmo uma sequência finita de instruções com os quais se pretende obter uma solução para um determinado problema. Na Figura 1 encontra-se o algoritmo que serviu de base ao ficheiro reta.py, apresentado no 1º Guião.

```
função ret
    ler abcA,ordA,abcB,ordB
    declive=(ordA-ordB)/(abcA-abcB)
    ordenada na origem=ordA-declive*abcA
    escrever declive
    escrever ordenada na origem
```

Fig. 1: Algoritmo do ficheiro reta.py.

2. Função condicional (IF-ELIF-ELSE)

Numa função condicional tem que se testar uma condição e conforme esta se torne numa proposição verdadeira ou numa proposição falsa, mediante o teste de um determinado valor, assim o programa executa uma instrução A, uma instrução B, ou uma instrução C. A sintaxe encontra-se na Figura 2.

```
If condição1:
    Instrução A
elif condição2:
    Instrução B
else:
    Instrução C
```

Fig. 2: Estrutura de uma função condicional.

Tome-se, como exemplo, a criação de um programa (Figura 3) que permita calcular a imagem de um determinado objeto da função real de variável real f definida por:

$$f(x) = \begin{cases} 2x-1 & se\ x < 0 \\ \sqrt{2+x} & se\ 0 \leq x < 2 \\ x^2+3 & se\ 2 \leq x < 4 \\ \frac{1}{x} & se\ x \geq 4 \end{cases}$$

Algoritmo:

função f
  ler x
  Se $x < 0$ então
    $y = 2x - 1$
  Se $(x \geq 0$ e $x < 2)$ então
    $y = \sqrt{2 + x}$
  Se $(x \geq 2$ e $x < 4)$ então
    $y = x^2 + 3$
  Se $x \geq 4$ então
    $y = 1/x$
  devolver y

Programa funcao:

```
def f(x):
    if x<0:
        y=2*x-1
    elif (x>=0 and x<2):
        y=(2+x)**(1/2)
    elif (x>=2 and x<4):
        y=x**2+3
    else
        y=1/x
    return(y)
```

Fig. 3: Algoritmo e programa função para a função f, usando uma função condicional[1].

3. Exercício

Escreve um programa em Python com uma função que leia os coeficientes de uma equação do 2º grau e devolva as soluções dessa equação.
A resposta deverá ter uma frase que refira os valores da(s) solução(ões), se existir(em), arredondada(s) às centésimas (usando a função format), ou que não tem solução.
Nome do ficheiro: resolvente.py

[1] Executa o ficheiro funcao.py e testa diversos valores para objetos da função f.

4. Programa resolvente (proposta de solução)

Em 3. foi lançado um desafio. Apresenta-se em seguida uma proposta de programa que satisfaz os requisitos considerados para o ficheiro resolvente.py (Figura 4).

Editor (E):

```
from math import *
def resolve(a, b, c) :
  delta=b**2-4*a*c
  if delta<0:
     print("A equação não tem soluções")
  elif delta==0:
     x1=-b/(2*a)
     print("A equação tem solução única:
{:.2f}".format(x1))
  else:
     x1=(-b-sqrt(delta))/(2*a)
     x2=(-b+sqrt(delta))/(2*a)
     print("A equação tem duas soluções: {:.2f}
e [:.2f}".format(x1,x2))
```

Interpretador (I):

```
resolve(1,2,1)
A equação tem solução única: -1.00
resolve(1,-3,2)
A equação tem duas soluções: 1.00 e 2.00
resolve(1,2,4)
A equação não tem soluções
```

Fig. 4: Proposta de solução para o programa resolvente[2].

[2] Executa as instruções referidas no Editor e confirma o seu resultado no Interpretador correndo o programa.

Guião 3 – Linguagem de programação Python
Matemática A

12.º Ano de escolaridade | Estruturas de repetição - Ciclos FOR e WHILE | Turma: [redacted]

No Guião 2 verificaste como deves proceder para efetuar uma função condicional. Neste Guião 3 encontras uma solução possível para o desafio lançado no Guião 2, bem como ficarás a conhecer como criar ciclos em Python.

1. Incrementar uma variável

Para incrementar uma variável numérica existem duas formas possíveis, apresentadas na Figura 1. No primeiro caso o contador começa em 1 e incrementa-se 1, obtendo-se 2 (pode-se visualizar o valor efetuando print(cont) na linha seguinte à segunda instrução). No segundo caso a variável contador inicia-se em 0 e acrescenta-se 2.

A

```
cont=1
cont=cont+1
```

B

```
cont=0
cont+=2
```

Fig. 1: Duas formas, A e B, de incrementar uma variável no Editor[1].

2. Ciclo FOR

Para a execução de certos programas é útil repetir uma ou várias instruções um determinado número finito de vezes. Sabendo antecipadamente o número de repetições que é necessário efetuar pode-se utilizar um ciclo FOR. A sintaxe encontra-se na Figura 2.

```
for variável in range():
    Instrução A
    Instrução B
    ...
```

Fig. 2: Estrutura de um ciclo FOR.

Ressalve-se que o contador do ciclo FOR se inicia, por defeito, em 0, e termina em $b-1$, com passo igual a 1, caso a instrução seja range(b).
Caso a instrução dada seja range(a,b), o contador inicia-se em a e percorre todos os números inteiros até b-1, sendo que no caso de a instrução dada ter a sintaxe range(a,b,c), altera-se o passo e em vez de ser de 1 em 1, é de c em c.
Na figura 3 encontra-se um problema que pode ser resolvido em Python usando um ciclo FOR (Figura 4).

> Pretende-se determinar a espessura do aglomerado de folhas de papel que se obtêm ao dobrar sucessivamente um determinado número de vezes uma folha de papel com uma espessura inicial de 0,1 mm. Em particular, qual é a espessura das folhas de papel que se obtêm quando se efetuam 10 dobragens? Qual é o número mínimo de dobragens que serão necessárias para que se obtenha uma espessura superior à distância da Terra à Lua (384 400 km ou 384 400 000 000 mm)?

Fig. 3: Problema das dobragens de uma folha de papel.

[1] ([1]).

Algoritmo:

```
função dob
  ler n
  e = 0.1
  Para i de 1 até n+1 :
     e = 2e
  devolver e
```

Ficheiro dobragens.py:

```
def dob(n) :
   e = 0.1
   for i in range(1,n+1):
      e = 2 * e
   return(e)
```

Fig. 4: Algoritmo e programa para o problema das dobragens de uma folha de papel[2].

Tendo por base o programa, verifica-se que ao fim de 10 dobragens se obtém uma espessura de 1,024 dm e que são necessárias 42 dobragens para obter uma espessura superior à distância da Terra à Lua, atingindo uma espessura de 439804651110,4 mm.

3. Ciclo WHILE

Quando se pretende repetir uma ou várias instruções um número indeterminado de vezes, é aconselhável utilizar um ciclo não limitado WHILE ("enquanto"). A sintaxe encontra-se na Figura 5.

```
while condição:
    Instrução A
    Instrução B
    ...
```

Fig. 5: Estrutura de um ciclo WHILE.

Não existe nenhuma instrução para indicar o fim de ciclo. O fim obtém-se através da indentação de instruções.
Na figura 6 encontra-se um problema que pode ser resolvido em Python usando um ciclo WHILE (Figura 7).

Método da bissecção:
Considere-se a função f definida em $[-1,2]$ definida por $f(x) = -x^3 + x + 1$.
Utilize a calculadora gráfica para traçar uma representação gráfica de f .
Determine a solução da equação $f(x) = 0$ utilizando o algoritmo conhecido por "método da bissecção".

Fig. 6: Resolução de uma equação usando o método da bissecção.

A utilização do algoritmo da bissecção, para determinar uma solução da equação $f(x) = 0$, baseia-se no Teorema de Bolzano-Cauchy e, sendo assim, pressupõe-se que a função dada, f, é contínua em $[a, b]$ e que $f(a) * f(b) < 0$.
Como tal admite-se que a solução, c, se encontra em $]a, b[$ e determina-se um valor para c com um erro máximo definido por $erro$.
O processo passa por calcular $f(a) \times f\left(\frac{a+b}{2}\right)$ e verificar o sinal. Se for negativo tomamos $b = \frac{a+b}{2}$, caso contrário $a = \frac{a+b}{2}$ e assim sucessivamente até $b - a < erro$.
O programa permite obter um intervalo de valores de amplitude $erro$ onde se encontra a solução pretendida.

[2] Executa o programa e testa diversos valores para um determinado número dobragens. Qual é o número mínimo de dobragens para obter uma distância superior à distância da Terra à Lua?

Algoritmo:

```
função f
  devolver −x³ + x + 1

função biss
  ler a, b, erro
  Enquanto (b − a) > erro
    c = (a + b)/2
    Se f(a) * f(c) ≤ 0:
        b = c
    Senão:
        a = c
  devolver a, b
```

Ficheiro bissecção.py:

```
def f(x) :
  return(−x ** 3 + x + 1)

def biss(a, b, erro)
  while (b − a) > erro:
    c = (a + b)/2
    if f(a) * f(c) <= 0:
      b = c
    else:
      a = c
  return(a, b)
```

Fig. 7: Algoritmo e programa para o método da bissecção[3].

Tendo por base o programa bissecção e escrevendo biss(-1,2,0.001) (erro máximo de 0,001) no modo interpretativo, tem-se que o intervalo onde se encontra a solução com amplitude 0,001 é [1,32470703125; 1,325439453125] (a solução obtida na calculadora é 1,324717957).

4. Exercício

Investiga o método de Newton sobre a determinação de zeros de uma função num intervalo dado e escreve um programa em Python que permita devolver uma solução aproximada para a equação $f(x) = 0$ com $f(x) = -x^3 + x + 1$ em $[-1,2]$.
Nome do ficheiro: newton.py

[3] Introduz as instruções referidas no ficheiro bissecção.py e confirma os resultados apresentados correndo o programa.

5. Programa newton (proposta de solução)

No ponto 4 deste guião foi lançado um desafio. Apresenta-se em seguida uma proposta de programa que satisfaz os requisitos considerados para o ficheiro newton.py (Figura 8).

Editor (E):

```
from math import *
def f(x) :
  return(-x ** 3 + x + 1)
def df(x) :
  return(-3 * x ** 2 + 1)
erro=10**-10
n=100
x0=0.5
x1=x0-f(x0)/df(x0)
i=1
while (abs(x1-x0)>=erro and i<n):
  x0=x1
  x1=x1-f(x1)/df(x1)
  i=i+1
print(x1)
```

Interpretador (I):

```
1.324717957244746
```

Fig. 8: Proposta de solução para o programa newton[4].

Como se pode verificar no interpretador o zero obtido satisfaz perfeitamente o valor obtido na calculadora e está no intervalo de valores obtido com o programa bissecção.

6. Exercício final

Trabalhando com o colega de trabalho com o qual executaste estes guiões, cria um programa em linguagem Python, tendo por base um tema matemático aprendido no secundário, aplicando conteúdos tratados nos guiões. No final devem elaborar um vídeo, com uma duração máxima de 5 minutos, explicando o programa desenvolvido e mostrando exemplos de aplicação. Devem ainda efetuar um relatório com um máximo de 3 páginas, sem contar com a capa e eventuais anexos, onde descrevam o que foi feito. Este relatório deve ser entregue até dia 31 de março.

[4] Introduz as instruções referidas no ficheiro newton.py e confirma os resultados apresentados correndo o programa.

IV

Anexo - Questionário final efetuado aos alunos

Questionário final

Nomes: _____ **N.º** _____

1. Aplicação dos guiões:

1.1. O que acharam dos guiões que utilizaram para aprender a trabalhar com a linguagem *Python*? Foram úteis? O que destacam da sua utilização?

1.2 Que conhecimentos mobilizaram na sua utilização? Que conhecimentos necessitaram de procurar?

1.3. Atendendo às tarefas propostas nos guiões, qual consideraram mais fácil e porquê? E mais complexa e porquê?

1.4. Se tivessem que propor estes guiões a colegas vossos que alterações introduziam?

No guião 1:

No guião 2:

No guião 3:

2. Trabalho final

2.1. Descrevam o processo que vos permitiu chegar ao programa que apresentaram no trabalho final. Quais foram as fontes que procuraram?

2.2. Quais foram as principais dificuldades sentidas na elaboração da tarefa que efetuaram?

2.3. Em que medida consideram que o problema que resolveram vos permitiu desenvolver o raciocínio matemático?

2.4. Como acham que programar em *Python* vos pode ajudar a desenvolver raciocínio matemático?

2.5. Acham que o trabalho desenvolvido foi útil? De que forma a resolução desta tarefa contribuiu para a vossa aprendizagem da Matemática?

2.5. Consideram importante a realização deste tipo de tarefas na disciplina de Matemática? Porquê?

2.6. Em termos metodológicos, acham útil terem efetuado um vídeo e um relatório para suportar o programa em *Python* que conceberam?

FIM

Utilizem este espaço para complementar alguma resposta a qualquer um dos itens:

V

ANEXO - PROGRAMAS EM *PYTHON* EFETUADOS PELOS ALUNOS

Estudo de caso 1 - Derivada

```
from math import*
numx4=0
numx3=0
numx2=0
numx1=0
numx0=0

derx3=0
derx2=0
derx1=0
derx0=0

der2x2=0
der2x1=0
der2x0=0

print ("Insira os coeficientes de uma função polinomial até ao grau 4 de forma
decrescente de maior grau:")
print ("Exemplo: fx(-5,4,-6,2,-3)")
def fx(numx4,numx3,numx2,numx1,numx0):
    if numx4!=0:
        derx3=numx4*4
    else:
        derx3=0
    if numx3!=0:
        derx2=numx3*3
    else:
        derx2=0
    if numx2!=0:
        derx1=numx2*2
    else:
        derx1=0
        if numx1!=0:
             derx0=numx1
    else:
        derx0=0
    if derx3!=0:
        print("f'(x)=",derx3,"x^3 ", end = "")
    else:
        print("f'(x)=")
    if derx2>0:
        print("+ ", end ="")
    if derx2!=0:
        print(derx2,"x^2 ", end = "")
    if derx1>0:
        print("+ ", end ="")
```

```
if derx1!=0:
     print(derx1,"x ", end = "")
if derx0>0:
     print("+ ", end ="")
if derx0!=0:
     print(derx0)

print("Para calcular a segunda derivada clique na cruz da janela com o gráfico")

import numpy as np
import matplotlib.pyplot as plt
def f(x): return derx3*x**3 + derx2*x**2 + derx1*x + derx0
x = np.linspace(-10,10)
plt.plot(x, f(x))
plt.title("1°derivada (f'(x))")
plt.grid()
plt.show()

import sympy
import sympy.solvers
from sympy.solvers.inequalities import reduce_rational_inequalities
from sympy import Symbol, N

x = sympy.Symbol("x")

ExampleInequalities1 = [[f(x)<=0]]
ResultDomain1 = reduce_rational_inequalities(ExampleInequalities1, x)
print ("A função é descrescente em",N(ResultDomain1.as_set()))

ExampleInequalities1 = [[f(x)>=0]]
ResultDomain1 = reduce_rational_inequalities(ExampleInequalities1, x)
print ("A função é crescente em",N(ResultDomain1.as_set()))

gx = (derx3,derx2,derx1,derx0)

if derx3!=0:
     der2x2=derx3*3
else:
     der2x2=0
if derx2!=0:
     der2x1=derx2*2
else:
     der2x1=0
if derx1!=0:
     der2x0=derx1
else:
     der2x0=0
```

```
   if der2x2!=0:
        print("f''(x)=",der2x2,"x^2 ", end = "")
   else:
        print("f''(x)=")
   if der2x1>0:
        print("+ ", end ="")
   if der2x1!=0:
        print(der2x1,"x ", end = "")
   if der2x0>0:
        print("+ ", end ="")
   if der2x0!=0:
        print(der2x0)

   print("Para calcular as concavidades do gráfico da função clique na cruz da janela
com o gráfico")

   import numpy as np
   import matplotlib.pyplot as plt

   def g(x): return der2x2*x**2 + der2x1*x + der2x0
   x = np.linspace(-10,10)
   plt.plot(x, g(x))
   plt.title("2°derivada (f''(x))")
   plt.grid()
   plt.show()

   import sympy
   import sympy.solvers
   from sympy.solvers.inequalities import reduce_rational_inequalities
   from sympy import Symbol, N

   x = sympy.Symbol("x")

   ExampleInequalities1 = [[g(x)>=0]]
   ResultDomain1 = reduce_rational_inequalities(ExampleInequalities1, x)
   print ("O gráfico da função tem a concavidade voltada para cima em",N(ResultDo-
main1.as_set()))

   ExampleInequalities1 = [[g(x)<=0]]
   ResultDomain1 = reduce_rational_inequalities(ExampleInequalities1, x)
   print ("O gráfico da função tem a concavidade voltada para baixo em",N(ResultDo-
main1.as_set()))
```

Estudo de caso 2 – Bancos

```
print("Aplicação que indica o dinheiro acumulado ao fim de n anos após depositá-lo
num determinado banco.")
print()
print()
print("A moeda utilizada é o euro(€).")
print()
print("Banco Português")
print()
print("Taxa de juro: 5%")
print("Capitalização semestral")
print("Se tiver o dinheiro depositado por 3 anos, no ano seguinte, a taxa sobe 1% por
ano.")
print("Se tiver o dinheiro depositado por 5 anos, no ano seguinte, a capitalização
passa a ser trimestral.")
print()
print()
print("Banco Europeu")
print()
print("Taxa de juro: 10%")
print("Capitalização anual")
print("Se tiver o dinheiro depositado por 4 anos, no ano seguinte, haverá mais 1
período de capitalização a cada 5 anos.")
print()
print("Escolha um banco para depositar o seu dinheiro.")
print()
print("Opção 1: Banco Português")
print("Opção 2: Banco Europeu")
print()
b=input("Opção escolhida:")
print()
while b != "1" and b != "2":
  print("Escolha entre as opções 1 e 2.")
  print()
  b=input("Opção escolhida:")
  print()

if b == "1":
  c0=int(input("Quanto dinheiro tenciona depositar?"))
  print()
  t=int(input("Durante quantos anos?"))
  print()
  if t<0 or c0<0:
    print()
    print("Não coloque números negativos.")
  else:
    if t>=0 and t<=3:
```

```
        k=c0*(1.025**(2*t))
    elif t>3 and t<=5:
        k=c0*((1+((2+t)/200))**(2*t))
    else:
        k=c0*((1+((2+t)/400))**(4*t))
    print()
    print("O valor acumulado será {:.2f}€".format(k))
if b== "2":
  c0=int(input("Quanto dinheiro tenciona depositar?"))
  print()
  t=int(input("Durante quantos anos?"))
  x=(1+(t//5))
  if t<0 or c0<0:
    print()
    print("Não coloque números negativos.")
  else:
    if t>=0 and t<=4:
        k=c0*(1.1**t)
    else:
        k=c0*(1+(0.1/x))**(x*t)
    print()
    print("O valor acumulado será {:.2f}€".format(k))
```

Estudo de caso 3 – Binómio de Newton

```
def binomialCoeff(n, k):
    result = 1
    for i in range(1, k+1):
        result = result * (n-i+1) / i
    return int(result)

def msg(msg):
    print("=" * 30)
    print ('~ %s ' % msg)
    print("=" * 30)

def bnewton():
    msg('Binómio de Newton (a+b)^n')
    n = int(input("Escreva um valor para n: "))
    if n < 0:
        print("Não, deve ser um número inteiro não negativo.")
    else:
        for i in range(0,n+1):
            print ('%i*a^(%i)*b^(%i)' %(binomialCoeff(n,i), n-i, i), end=" ")
            if i != n:
                print('+', end=" ")
```

Estudo de caso 4 – Juros

```
print("Olá, este é um site de um banco que te irá ajudar a realizar câmbio entre moedas, comparar diferentes moedas e adquirir os teus produtos de sonho a taxas de juro imbatíveis!")
print()
print("ATENÇÃO- Dada a situação volátil que se tem verificado a nível global com a guerra na Ucrânia, algumas operações de câmbio estão inoperacionais!")
print("Por favor, prima:")
print()
print("1 - Câmbio entre moedas")
print("2 - Comparação de moedas")
print("3 - Chega para comprar algo em euros?")
print("4 - Pagamento a prestações")
print("5 - Dinheiro acumulado ao fim de x meses")
print()

def funcao(t):
    if t == 1:
        return func1()
    elif t == 2:
        return func2()
    elif t == 3:
        return func3()
    elif t == 4:
        return func4()
    elif t==5:
        return func5()
    else:
        print("Que inteligente, essa opção não existe!!!")

def func1():
  print("Os valores de câmbio apresentados são relativos ao dia de hoje, com base na atual cotação das moedas apresentadas!")
  print("As opções de conversão de moeda são:")
  print()
  print("1 - EUR")
  print("2 - USD")
  print("3 - GBP")
  print("4 - JPY")
  print()
  print()
  print()
  a = int(input("Moeda a converter: "))
    print()
  b = round(float(input("Quantia a converter: ")), 2)
    print()
```

```
c = int(input("Moeda para converter: "))
print()
print()
if a == 1:
  if c == 1:
    print("Essa conversão é um pouco idiota, escolha 2 moedas diferentes!")
  elif c== 2:
    print(b, "€ =", round(b * 1.10795, 2), "$")
  elif c == 3:
    print(b, "€ =", round(b * 0.84355, 2), "£")
  elif c == 4:
    print(b, "€ =", round(b * 134.72435, 2), "¥")
  else:
    print("Que inteligente, essa opção não existe!!!")
elif a == 2:
  if c == 1:
    print(b, "$ =", round(b * 0.90239, 2), "€")
  elif c == 2:
    print("Essa conversão é um pouco idiota, escolha 2 moedas diferentes!")
  elif c == 3:
    print(b, "$ =", round(b * 0.76113, 2), "£")
  elif c == 4:
    print(b, "$ =", round(b * 121.59003, 2), "¥")
  else:
    print("Que inteligente, essa opção não existe!!!")
  elif a == 3:
  if c == 1:
    print(b, "£ =", round(b * 1.18551, 2), "€")
  elif c == 2:
    print(b, "£ =", round(b * 1.31427, 2), "$")
  elif c == 3:
    print("Essa conversão é um pouco idiota, escolha 2 moedas diferentes!")
  elif c == 4:
    print(b, "£ =", round(b * 159.80032, 2), "¥")
  else:
    print("Que inteligente, essa opção não existe!!!")
elif a == 4:
  if c == 1:
    print(b, "¥ =", round(b * 0.00742, 2), "€")
  elif c == 2:
    print(b, "¥ =", round(b * 0.00822, 2), "$")
  elif c == 3:
    print(b, "¥ =", round(b * 0.00668, 2), "£")
  elif c == 4:
    print("Essa conversão é um pouco idiota, escolha 2 moedas diferentes!")
  else:
    print("Que inteligente, essa opção não existe!!!")
```

```
  else:
    print("Program error. Por favor insira um número válido!")
  print()
  print("Deseja:")
  print()
  print("1 - Voltar à página inicial")
  print("2 - Sair do site")
  print()
  print()
  v=int(input("Escolha: "))
  print()
  print()
  if v==1:
      print()
      print("1 - Câmbio entre moedas")
      print("2 - Comparação de moedas")
      print("3 - Chega para comprar algo em euros?")
      print("4 - Pagamento a prestações")
      print("5 - Dinheiro acumulado ao fim de x meses")
      print()
      funcao(int(input("Escolha: ")))
  if v==2:
      exit()
  else:
      print("Program error. Por favor insira um número válido!")

def func2():
  print("As opções de comparação são:")
  print()
  print("1 - EUR")
  print("2 - USD")
  print("3 - GBP")
  print("4 - JPY")
  print()
  print()
  d = int(input("Moeda 1: "))
  print()
  e = int(input("Moeda 2: "))
  print()
  print()
  if d == 1:
    if e == 1:
      print("1.00 € = 1.00 €")
    elif e == 2:
      print("1.00 € > 1.00 $")
    elif e == 3:
      print("1.00 € < 1.00 £")
    elif e == 4:
```

```
    print("1.00 € > 1.00 ¥")
  else:
    print("Program error. Por favor insira um número válido!")
elif d == 2:
  if e == 1:
    print("1.00 $ < 1.00 €")
  elif e == 2:
    print("1.00 $ = 1.00 $")
  elif e == 3:
    print("1.00 $ < 1.00 £")
  elif e == 4:
    print("1.00 $ > 1.00 ¥")
  else:
    print("Program error. Por favor insira um número válido!")
  elif d == 3:
  if e == 1:
    print("1.00 £ > 1.00 €")
  elif e == 2:
    print("1.00 £ > 1.00 $")
  elif e == 3:
    print("1.00 £ = 1.00 £")
  elif e == 4:
    print("1.00 £ > 1.00 ¥")
  else:
    print("Program error. Por favor insira um número válido!")
elif d == 4:
  if e == 1:
    print("1.00 ¥ < 1.00 €")
  elif e == 2:
    print("1.00 ¥ < 1.00 $")
  elif e == 3:
    print("1.00 ¥ < 1.00 £")
  elif e == 4:
    print("1.00 ¥ = 1.00 ¥")
  else:
    print("Program error. Por favor insira um número válido!")
else:
  print("Program error. Por favor insira um número válido!")
print()
print("Deseja:")
print()
print("1 - Voltar à página inicial")
print("2 - Sair do site")
print()
print()
v=int(input("Escolha: "))
print()
print()
```

```
  if v==1:
      print()
      print("1 - Câmbio entre moedas")
      print("2 - Comparação de moedas")
      print("3 - Chega para comprar algo em euros?")
      print("4 - Pagamento a prestações")
      print("5 - Dinheiro acumulado ao fim de x meses")
      print()
      funcao(int(input("Escolha: ")))
  if v==2:
      exit()
  else:
      print("Program error. Por favor insira um número válido!")

def func3():
  x = int(input("Preço artigo desejado em euros: "))
    print()
  print()
    print("Diga a moeda que quer usar:")
  print()
  print("1 - EUR")
  print("2 - USD")
  print("3 - GBP")
  print("4 - JPY")
  print()
  print()
  f = int(input("Escolha: "))
  print()
  g = round(float(input("Quantia: ")), 2)
  print()
  print()
  if f == 1:
    h = g
  elif f == 2:
    h = round(g * 0.90190, 2)
  elif f == 3:
    h = round(g * 1.18554, 2)
  elif f == 4:
    h = round(g * 0.00742, 2)
  else:
    return print("Program error. Por favor insira um número válido!")
  if int(h//x) == 0:
    print("Infelizmente não tem dinheiro suficiente para comprar o seu artigo.")
  elif int(h//x) == 1:
    print("Com esse dinheiro pode comprar, pelo menos, 1 artigo")
  elif int(h//x) > 1:
    print("Com esse dinheiro pode comprar, pelo menos,", int(h//x), "artigos.")
  else:
```

```
    print("Program error. Por favor insira um número válido!")
  print("Deseja:")
  print()
  print("1 - Voltar à página inicial")
  print("2 - Sair do site")
  print()
  print()
  v=int(input("Escolha: "))
  print()
  print()
  if v==1:
      print()
      print("1 - Câmbio entre moedas")
      print("2 - Comparação de moedas")
      print("3 - Chega para comprar algo em euros?")
      print("4 - Pagamento a prestações")
      print("5 - Dinheiro acumulado ao fim de x meses")
      print()
      funcao(int(input("Escolha: ")))
  if v==2:
      exit()
  else:
      print("Program error. Por favor insira um número válido!")

def func4():
  print("Deseja:")
  print()
  print("1 - Converter o total em prestações")
  print("2 - Converter as prestações num total")
  print()
  print()
    i = int(input("Escolha: "))
  print()
  print()
  if i == 1:
    j = round(float(input("Total: ")), 2)
    print()
    k = int(input("Número de meses: "))
    print()
    print()
    if j > 0 and k > 0:
        print("A prestação seria de", round(j/k+j*0.1/k,2), "por mês. O que resulta
do pagamento simplificado de", round(j/k,2), "por mês e da taxa de juro fixa de 5%,
que equivale a ", j*0.05)
    else:
        print("Program error. Por favor insira um número válido!")
    elif i == 2:
        j = round(float(input("Custo por mês: ")), 2)
```

```
        print()
        k = int(input("Número de meses: "))
        print()
        print()
    if j > 0 and k > 0:
        print("No total estaria a pagar", round(j*k, 2))
    else:
        print("Program error. Por favor insira um número válido!")
    else:
        print("Program error. Por favor insira um número válido!")
  print()
  print("Deseja:")
  print()
  print("1 - Voltar à página inicial")
  print("2 - Sair do site")
  print()
  print()
  v=int(input("Escolha: "))
  print()
  print()
  if v==1:
      print()
      print("1 - Câmbio entre moedas")
      print("2 - Comparação de moedas")
      print("3 - Chega para comprar algo em euros?")
      print("4 - Pagamento a prestações")
      print("5 - Dinheiro acumulado ao fim de x meses")
      print()
      funcao(int(input("Escolha: ")))
  if v==2:
      exit()
  else:
      print("Program error. Por favor insira um número válido!")

def func5():
    print()
    q=round(float(input("Qual é o seu saldo atual? " )),2)
    print()
    r=int(input("Quer saber quanto dinheiro terá daqui a quantos meses? "))
    print()
    x=round(float(input("Qual é o seu juro composto mensal? ")),2)
    if x <0:
        print("Program error. Por favor insira um número válido!")
    elif x>5:
        print("Nenhum banco é assim tão simpático com os seus clientes!")
    else:
        y= (q*(1+x/100)**r)
```

```
        print("Daqui a", r, " meses terá um saldo de", (round(float(y),2)))
    print()
    print("Deseja:")
    print()
    print("1 - Voltar à página inicial")
    print("2 - Sair do site")
    print()
    print()
    v=int(input("Escolha: "))
    print()
    print()
    if v==1:
        print()
        print("1 - Câmbio entre moedas")
        print("2 - Comparação de moedas")
        print("3 - Chega para comprar algo em euros?")
        print("4 - Pagamento a prestações")
        print("5- Dinheiro acumulado ao fim de x meses")
        print()
        funcao(int(input("Escolher: ")))
    if v==2:
        exit()
    else:
        print("Program error. Por favor insira um número válido!")

funcao(int(input("Escolha: ")))
```

Estudo de caso 5 – Triângulo de Pascal

```
from math import *
def pascal():
    n = int(input("Escreva o numero da linha de pascal: "))
    print("\nAmostra do triângulo ate á linha " + str(n))
    triângulo(n + 1)
    soma = 2**(n)
    print("\nA soma da linha " + str(n) + " é " + str(soma))
    somatório = ((1 - 2**n)//(1 - 2)) * 1
    print("\nA soma de todas linhas anteriores até á linha " + str(n) + " é " +
str(somatório))

def triângulo(c):
    for n in range(c):
        for k in range(n + 1):
            print(factorial(n)//(factorial(k)*factorial(n-k)), end = " ")
        print()

pascal()
```

Estudo de caso 6 – Funções_Cálculo

```
from math import *
print("Esta é uma ferramenta de análise de funções. Para começar é necessário definir
a função.")
print()
a=float(input("Indique um valor para a: "))
b=float(input("Indique um valor para b: "))
c=float(input("Indique um valor para c: "))
print()
if a==0:
   print("A sua função é uma reta e está definida por f(x) =",b,"x +",c,".")
   print()
   print("O declive da sua função é ",b,".")
   print()
   if b<0:
       zero=-c/b
       print("O zero da sua função é {:.2f}.".format(zero))
       print()
       print("A função é decrescente.")
       print()
   if b>0:
       zero=-c/b
       print("O zero da sua função é {:.2f}.".format(zero))
       print()
       print("A função é crescente.")
       print()
   else:
       print("A função é constante, logo não tem zeros.")
       print()
else:
   if b<0 and c>0:
       print("A sua função é uma parábola e está definida por f(x) =",a,"x^2",b,"x
+",c,".")
       print()
   if c<0 and b>0:
       print("A sua função é uma parábola e está definida por f(x) =",a,"x^2
+",b,"x",c,".")
       print()
   if b<0 and c<0:
       print("A sua função é uma parábola e está definida por f(x)
=",a,"x^2",b,"x",c,".")
       print()
   if b>0 and c>0:
       print("A sua função é uma parábola e está definida por f(x) =",a,"x^2 +",b,"x
+",c,".")
       print()
```

```
   print("A função está definida em |R.")
   print()
   x1=(-b)/(2*a)
   y1=((-b**2+4*a*c)/(4*a))
   print("O vértice da sua parábola tem as coordenadas ({:.2f},{:.2f}).".for-
mat(x1,y1))
   print()
   if a < 0 :
       print("A concavidade da parábola está voltada para baixo.")
       print()
       print("A função é crescente em ]-∞,{:.2f}] e é decrescente em
[{:.2f},+∞[.".format(x1,x1))
       print()
       print("O contradomínio da função é ]-∞,{:.2f}].".format(y1))
       print()
    else:
       print("A concavidade da parábola está voltada para cima.")
       print()
       print("A função é decrescente em ]-∞,{:.2f}] e é crescente em
[{:.2f},+∞[.".format(x1,x1))
       print()
       print("O contradomínio da função é [{:.2f},+∞[.".format(y1))
       print()
       bi=b**2-4*a*c
   if bi<0:
       print("A função não tem zeros :(.")
       print()
       if a<0:
          print("A função tem sinal negativo em |R.")
          print()
       else:
          print("A função tem sinal positivo em |R.")
          print()
   elif bi==0:
       x2=-b/(2*a)
       print("A função tem um zero: {:.2f}.".format(x2))
       print()
       if a<0:
          print("A função tem sinal negativo em |R\{{:.2f}}.".format(x2))
          print()
       else:
          print("A função tem sinal positivo em |R\{{:.2f}}.".format(x2))
          print()
   else:
       x3=(-b-sqrt(bi))/(2*a)
       x4=(-b+sqrt(bi))/(2*a)
       print("A função tem dois zeros: {:.2f} e {:.2f}.".format(x3,x4))
       print()
```

```
       if a<0:
          print("A função tem sinal negativo em ]-∞,{:.2f}[ e em ]{:.2f},+∞[.".for-
mat(x3,x4))
          print()
          print("A função tem sinal positivo em ]{:.2f},{:.2f}[.".format(x3,x4))
          print()
       else:
          print("A função tem sinal positivo em ]-∞,{:.2f}[ e em ]{:.2f},+∞[.".for-
mat(x3,x4))
          print()
          print("A função tem sinal negativo em ]{:.2f},{:.2f}[.".format(x3,x4))
          print()
       if b<0:
          deriv=2*a
          print("A expressão da derivada da função é {:.1f}x{:.1f}.".format(deriv,b))
          print()
          ponto=float(input("Calcular a derivada no ponto de abcissa: "))
          print()
          dponto=2*a*ponto+b
          print("A derivada no ponto de abcissa",ponto,"é",dponto,".")
          print()
       else:
          deriv2=2*a
          print("A expressão da derivada da função é {:.1f}x + {:.1f}.".format(de-
riv2,b))
          print()
          ponto2=float(input("Calcular a derivada no ponto de abcissa: "))
          print()
          dponto2=2*a*ponto2+b
          print("A derivada no ponto de abcissa",ponto2,"é",dponto2,".")
          print()
aval=float(input("De  0 a 10, por favor, classifique a sua experiência durante a
análise desta função: "))
print()
print("Obrigado pelo feedback!")
```

Estudo de caso 7 – Média do secundário

```
print("Olá, sabemos como é difícil calcular a média do secundário para entrar no ensino superior, e para te ajudar criámos um simulador simples e eficaz,aproveita!")
print("Informamos que este ano os exames apenas contarão como provas de acesso ao ensino superior!!!")
print()
print("Por favor prima:")
print()
print("1- Curso Científico-humanístico")
print("2- Curso Profissional")
print()

def funcao(a):
   print()
   if a==1:
      return funcao1()
   elif a==2:
      print("Infelizmente ainda não temos esse simulador disponível!")
      exit()
   else:
      print("Erro, tente um número válido!")
      exit()

def funcao1():
   b=round(float(input("Qual é a média do curso que pretendes seguir? ")),2)
   print()
   if b>20 or b<0:
      print("Esse valor não é válido")
      exit()
   elif b>0 and b<10:
      print("Duvido que existam cursos com médias tão fracas!")
   print()
   c=int(input("Quantos exames vais usar como provas de ingresso? "))
   print()
   if c>3 or c <0:
      print("Só podes usar até 3 exames!")
   elif c==0:
      print("Tens de usar pelo menos um exame para qualquer curso!")
   else:
      d=int(input("Qual é o peso total dos exames como provas de ingresso, em percentagem: "))
      print()
      if d>50 or d<35:
         print("Esse valor não é válido!")
         exit()
      else:
         if c==1:
```

```
            med=float(input("Quanto teve no seu exame?"))
        if c==2:
            f=int(input("Qual é o peso do seu primeiro exame? "))
            print()
            g=int(input("Qual é o peso do seu segundo exame? "))
            print()
            h=float(input("Quanto teve no primeiro exame?"))
            print()
            i=float(input("Quanto teve no segundo exame?"))
            med=round(((h+i)/2),1)
            if f+g!=d:
               print("A soma dos seus exames tem que corresponder ao valor acima
indicado!")
               exit()
        if c==3:
            j=int(input("Qual é o peso do seu primeiro exame?"))
            print()
            k=int(input("Qual é o peso do seu segundo exame?"))
            print()
            l=int(input("Qual é o peso do seu terceiro exame?"))
            print()
            m=float(input("Quanto teve no primeiro exame?"))
            print()
            n=float(input("Quanto teve no segundo exame?"))
            print()
            o=float(input("Quanto teve no terceiro exame?"))
            med=round(((m+n+o)/3),1)
            if j+k+l!=d:
               print("A soma dos seus exames tem que corresponder ao valor acima
indicado!")
               exit()
      print()
      print("Agora que já falámos de exames prepara-te para fazer a média do secun-
dário!")
  p=int(input("Quanto teve a português no 10 ano?"))
  print()
  if p>20:
     print("Esse número não é válido!")
  if p<10:
     print("Não aceitamos notas negativas no nosso site, estudaaaaa!")
  print()
  q=int(input("Quanto teve a português no 11 ano?"))
  print()
  if q>20:
     print("Esse número não é válido!")
  if q<10:
     print("Não aceitamos notas negativas no nosso site, estudaaaaa!")
  print()
```

```
r=int(input("Quanto teve a português no 12 ano?"))
print()
if r>20:
   print("Esse número não é válido!")
if r<10:
   print("Não aceitamos notas negativas no nosso site, estudaaaaa!")

print()
s=int(input("Quanto teve a educação física no 10 ano?"))
print()
if s>20:
   print("Esse número não é válido!")
   exit()
if s<10:
   print("Não aceitamos notas negativas no nosso site, estudaaaaa!")
   exit()
print()

u=int(input("Quanto teve a educação física no 11 ano?"))
print()
if u>20:
   print("Esse número não é válido!")
   exit()
if u<10:
   print("Não aceitamos notas negativas no nosso site, estudaaaaa!")
   exit()
print()
v=int(input("Quanto teve a educação física no 12 ano?"))
print()
if v>20:
   print("Esse número não é válido!")
   exit()
if v<10:
   print("Não aceitamos notas negativas no nosso site, estudaaaaa!")
   exit()

print()
w=int(input("Quanto teve à disciplina de opção trienal no 10 ano?"))
print()
if w>20:
   print("Esse número não é válido!")
   exit()
if w<10:
   print("Não aceitamos notas negativas no nosso site, estudaaaaa!")
   exit()
print()
y=int(input("Quanto teve à disciplina de opção trienal no 11 ano?"))
```

```
print()
if y>20:
   print("Esse número não é válido!")
   exit()
if y<10:
   print("Não aceitamos notas negativas no nosso site, estudaaaaa!")
   exit()
print()

x=int(input("Quanto teve à disciplina de opção trienal no 12 ano?"))
print()
if x>20:
   print("Esse número não é válido!")
   exit()
if x<10:
   print("Não aceitamos notas negativas no nosso site, estudaaaaa!")
   exit()

fil=int(input("Quanto teve a filosofia no 10 ano?"))
print()
if fil>20:
   print("Esse número não é válido!")
   exit()
if fil<10:
   print("Não aceitamos notas negativas no nosso site, estudaaaaa!")
   print()
   exit()

fil2=int(input("Quanto teve a filosofia no 11 ano?"))
print()
if fil2>20:
   print("Esse número não é válido!")
   exit()
if fil2<10:
   print("Não aceitamos notas negativas no nosso site, estudaaaaa!")
   print()
   exit()

ing=int(input("Quanto teve a inglês no 10 ano?"))
print()
if ing>20:
   print("Esse número não é válido!")
   exit()
if ing<10:
   print("Não aceitamos notas negativas no nosso site, estudaaaaa!")
   print()
   exit()
ing2=int(input("Quanto teve a inglês no 11 ano?"))
```

```
print()
if fil2>20:
   print("Esse número não é válido!")
   exit()
if fil2<10:
   print("Não aceitamos notas negativas no nosso site, estudaaaaa!")
   print()
   exit()

opc1=int(input("Quanto teve à disciplina de opção bianual 1 no 10 ano?"))
print()
if opc1>20:
   print("Esse número não é válido!")
   exit()
if opc1<10:
   print("Não aceitamos notas negativas no nosso site, estudaaaaa!")
   print()
   exit()

opc11=int(input("Quanto teve à disciplina de opção bianual 1 no 11 ano?"))
print()
if opc11>20:
   print("Esse número não é válido!")
   exit()
if opc11<10:
   print("Não aceitamos notas negativas no nosso site, estudaaaaa!")
   print()
   exit()

opc2=int(input("Quanto teve à disciplina de opção bianual 2 no 10 ano?"))
print()
if opc2>20:
   print("Esse número não é válido!")
   exit()
if opc2<10:
   print("Não aceitamos notas negativas no nosso site, estudaaaaa!")
   print()
   exit()

opc21=int(input("Quanto teve à disciplina de opção bianual 2 no 11 ano?"))
print()
if opc21>20:
   print("Esse número não é válido!")
   exit()
if opc21<10:
   print("Não aceitamos notas negativas no nosso site, estudaaaaa!")
   print()
    exit()
```

```
  opcao1=int(input("Quanto teve à disciplina de opção anual 1 no 12 ano?"))
  print()
  if opcao1>20:
     print("Esse número não é válido!")
     exit()
  if opcao1<10:
     print("Não aceitamos notas negativas no nosso site, estudaaaaa!")
     print()
     exit()

  opcao2=int(input("Quanto teve à disciplina de opção anual 2 no 12 ano?"))
  print()
  if opcao2>20:
     print("Esse número não é válido!")
     exit()
  if opcao2<10:
     print("Não aceitamos notas negativas no nosso site, estudaaaaa!")
     print()
     exit()
  print()

  ptfin=int((p+q+r)//3)
  edfin=int((s+u+v)//3)
  opctri=int((w+x+y)//3)
  filfin=int((fil+fil2)//2)
  ingfin=int((ing+ing2)//2)
  opcbi=int((opc1+opc11)//2)
  opcbi2=int((opc2+opc21)//2)
  finalnota=round(((ptfin+edfin+opctri+filfin+ingfin+opcbi+opcbi2+opcao1+op-
cao2)/9),1)

  finalfinal=round((med*0.01*d+finalnota*0.01*(100-d)),2)
  print(" A sua nota de secundário de ", finalnota, " com a dos seus exames de", med,
" dá uma média final de ", finalfinal)
  if finalfinal>=b:
     print("Parabéns, tens média para entrar no curso que queres!")
  else:
     print("Infelizmente não tens média para o curso que queres, tenta melhorar as
notas dos exames ou então procura outro curso. Não desistas!!!!!")

funcao(int(input("Escolha: ")))
```

Estudo de caso 8 – Método de Newton-Raphson

```
import numpy as np
import matplotlib.pyplot as plt
import sympy as sm

x=sm.Symbol('x')

#função e função derivada
f = x**3+1
df=sm.diff(f,x)

i=1
#intervalo de detção do zero e janela de visualização
a=float(input("Inserir limite mínimo para o intervalo de deteção do zero:"))
b=float(input("Inserir limite máximo para o intervalo de deteção do zero:"))
h1=float(input("Inserir limite inferior do gráfico:"))
h2=float(input("Inserir limite superior do gráfico:"))
#valor da aproximação inicial
x0=float(input("Inserir o valor da aproximação inicial:"))
x1=x0+1
#lista com os pontos do gráfico da função
lx = np.linspace(a,b,500)
ly = np.array([f.subs(x,v) for v in lx],dtype='float')
#formatação do gráfico
plt.figure(figsize=(40,40))
plt.plot(lx,ly, color='red', linewidth=6)
plt.axhline(0, color='black', linewidth=3)
plt.axvline(0, color='black', linewidth=3)
plt.title('Método de Newton-Raphson', fontsize=85)
plt.xlim(a,b)
plt.ylim(h1,h2)

#ciclo do método de newton-raphson
while np.abs(x1-x0)>0.01:
    a0=df.subs(x,x0) #f'(x0)
    y0=f.subs(x,x0)  #f(x0)
    reta=y0+a0*(x-x0) #equação da reta tangente

    lreta = np.array([reta.subs(x,u) for u in lx],dtype='float') #lista com os pontos
do gráfico da reta tangente

    plt.plot(lx,lreta, color='blue') #formatação da reta tangente
    x1=x0
    x0=x0-y0/a0 #substituição do valor x0 pela interseção da tangente com o eixo das
abcissas
    plt.plot(x0,0, marker="o", markersize=20, markerfacecolor='magenta') #formatação
dos pontos de interseção
```

```
    label="{:.0f}".format(i)
    plt.text(x0,-0.1, label, fontsize=55, color='magenta')
    i+=1
#fim do ciclo
print("A função tem um zero aproximadamente em {}.".format(float(x0)))
print("O método de Newton aproximou-se do zero ao fim de {:.0f} iterações.".format(i-
1))
plt.plot(x0,0, marker="o", markersize=30, markerfacecolor='green')
plt.text(x0,0.05, label, fontsize=80, color='green')
```

Estudo de caso 9 – Juros compostos

```
import tkinter as tk
from tkinter import BOTTOM, END, RIGHT, StringVar, Toplevel, filedialog, Text
menu = tk.Tk()
menu.geometry("520x200")
menu.title("Juros Compostos")
lbl = tk.Label(menu, text ="O depósito que efetuará terá uma taxa de juro efetiva:",
font=("Arial Black", 10))
lbl.pack()

def openNewWindow():
    root = Toplevel(menu)
    root.title("Juros Compostos")
    txta = StringVar()
    z= 0
    def adding():
        d=0
        a= float(E1.get())
        b= float(E2.get())
        c= float(E3.get())
        z= abs(1+(b/100))**c
        d= a*z
        txta.set(str(d))
        tk.messagebox.showinfo("Juros Compostos", "Ao fim de {} anos com um depósito
inicial de {} € e aplicada uma taxa de juro de {} % ao ano, ficará com um montante
final de {} €.".format(c, a, b, d))
    button1= tk.Button(root, text = "Calcular", font="Elephant", bg="Light Blue",
activebackground="yellow", command= lambda :adding())
    button1.grid(row=4, column=1)
    L1 = tk.Label(root,text = "Insere o valor do depósito inicial (€), da taxa de
juro(%) e do tempo (anos)", font=("Calibri", 10))
    L1.grid(row = 0, column = 0, columnspan =2)
    LA = tk.Label(root,text= "Depósito Inicial (€)= ", font=("Cooper Black", 10))
    LA.grid(row = 1, column= 0)
    LB = tk.Label(root,text= "Taxa de Juro (%) = ", font=("Cooper Black", 10))
    LB.grid(row = 2, column= 0)
    LC = tk.Label(root,text= "Tempo (anos) = ", font=("Cooper Black", 10))
    LC.grid(row = 3, column= 0)
    E1= tk.Entry(root)
    E1.grid(row= 1, column= 1)
    E2= tk.Entry(root)
    E2.grid(row= 2, column= 1)
    E3= tk.Entry(root)
    E3.grid(row= 3, column= 1)
    L4= tk.Label(root,text = "Montante final (€) = ", font=("Arial Black", 13),
fg="blue")
    L4.grid(row=5, column= 0)
```

```
    E4 = tk.Entry(root, textvariable = txta)
    E4.grid (row=5, column=1)

def create_window():
    semestre = Toplevel(menu)
    semestre.title("Juros Compostos")
    txta = StringVar()
    z= 0
    def adding():
        d=0
        a=float(E1.get())
        b= float(E2.get())
        c= float(E3.get())
        z= abs(1+(b/(100*2)))**(c*2)
        d= a*z
        txta.set(str(d))
        tk.messagebox.showinfo("Juros Compostos", "Ao fim de {} anos com um depósito
inicial de {} € e aplicada uma taxa de juro de {} % por semestre, ficará com um
montante final de {} €.".format(c, a, b, d))
    button1= tk.Button(semestre, text = "Calcular", font="Elephant", bg="Light Blue",
activebackground="yellow", command= lambda :adding())
    button1.grid(row=4, column=1)
    L1 = tk.Label(semestre,text = "Insere o valor do depósito inicial (€), da taxa
de juro(%) e do tempo (anos)", font=("Calibri", 10))
    L1.grid(row = 0, column = 0, columnspan =2)
    LA = tk.Label(semestre,text= "Depósito Inicial (€)= ", font=("Cooper Black", 10))
    LA.grid(row = 1, column= 0)
    LB = tk.Label(semestre,text= "Taxa de Juro (%) = ", font=("Cooper Black", 10))
    LB.grid(row = 2, column= 0)
    LC = tk.Label(semestre,text= "Tempo (anos) = ", font=("Cooper Black", 10))
    LC.grid(row = 3, column= 0)
    E1= tk.Entry(semestre)
    E1.grid(row= 1, column= 1)
    E2= tk.Entry(semestre)
    E2.grid(row= 2, column= 1)
    E3= tk.Entry(semestre)
    E3.grid(row= 3, column= 1)
    L4= tk.Label(semestre,text = "Montante final (€) = ", font=("Arial Black", 13),
fg="blue")
    L4.grid(row=5, column= 0)
    E4 = tk.Entry(semestre, textvariable = txta)
    E4.grid ( row=5, column=1)

def create():
    meses = Toplevel(menu)
    meses.title("Juros Compostos")
    txta = StringVar()
    z= 0
```

```
 def adding():
     d=0
     a=float(E1.get())
     b= float(E2.get())
     c= float(E3.get())
     z= abs(1+(b/(100*12)))**(c*12)
     d= a*z
     txta.set(str(d))
     tk.messagebox.showinfo("Juros Compostos", "Ao fim de {} anos com um depósito
inicial de {} € e aplicada uma taxa de juro mensal de {} %, ficará com um montante
final de {} €.".format(c, a, b, d))
    button1= tk.Button(meses, text = "Calcular", font="Elephant", bg="Light Blue",
activebackground="yellow", command= lambda :adding())
    button1.grid(row=4, column=1)
    L1 = tk.Label(meses,text = "Insere o valor do depósito inicial (€), da taxa de
juro(%) e do tempo (anos)", font=("Calibri", 10))
    L1.grid(row = 0, column = 0, columnspan =2)
    LA = tk.Label(meses,text= "Depósito Inicial (€)= ", font=("Cooper Black", 10))
    LA.grid(row = 1, column= 0)
    LB = tk.Label(meses,text= "Taxa de Juro (%) = ", font=("Cooper Black", 10))
    LB.grid(row = 2, column= 0)
    LC = tk.Label(meses,text= "Tempo (anos) = ", font=("Cooper Black", 10))
    LC.grid(row = 3, column= 0)
    E1= tk.Entry(meses)
    E1.grid(row= 1, column= 1)
    E2= tk.Entry(meses)
    E2.grid(row= 2, column= 1)
    E3= tk.Entry(meses)
    E3.grid(row= 3, column= 1)
    L4= tk.Label(meses,text = "Montante final (€) = ", font=("Arial Black", 13),
fg="blue")
    L4.grid(row=5, column= 0)
    E4 = tk.Entry(meses, textvariable = txta)
    E4.grid ( row=5, column=1)

btn1 = tk.Button(menu,text ="Anual",font="Elephant", bg="Light Blue", activeback-
ground="yellow", command = openNewWindow)
btn1.pack(pady = 10)

btn2 = tk.Button(menu,text ="Semestral",font="Elephant", bg="Light Blue", activeback-
ground="yellow", command = create_window)
btn2.pack(pady = 10)

btn3 = tk.Button(menu,text ="Mensal",font="Elephant", bg="Light Blue", activeback-
ground="yellow", command = create)
btn3.pack(pady = 10)

menu.mainloop()
```

Estudo de caso 10 – Derivação

```
import numpy as np
import matplotlib.pyplot as plt
from math import *
from scipy.optimize import fsolve
import scipy as sp

m = int(input("Derivar uma função de 1°grau, 2°grau ou de 3°grau? Ans (1, 2 ou 3):"))
if m == 1:
    print("f(x)= ax + b")
    a = int(input("a="))
    b = int(input("b="))

    def f(x) :
        return(a*x + b)
    print("f(x) = {}x + {}".format(a,b))
    print("A derivada de f(x) é df(x) = {}".format(a))

    plt.axhline(y=a, color='r', linestyle='-')
    plt.show()

elif m == 2:
    print("f(x)= ax^2 + bx + c")
    a = int(input("a="))
    b = int(input("b="))
    c = int(input("c="))
    print("f(x) = {}x^2 + {}x + {}".format(a,b,c))
    z=2*a
    print("A derivada de f(x) é df(x) = {}x + {}".format(z,b))

    def f(x) :
        return(a*x**2 + b*x + c)

    def df(x) :
        h = 0.00000000001
        fun = (f(x+h)-f(x))/(h)
        return float("%.3f" %fun)

    def real_df(x) :
        h = 0.00000000001
        fun = (f(x+h)-f(x))/(h)
        return(fun)

#Grafico
#Dominio
d = int(input("Limite inferior do domínio da função é "))
D = int(input("Limite superior do domínio da função é "))
```

```
#Funções
x = np.linspace(d,D,100)
y = real_df(x)
j = f(x)

#Eixos
ax = plt.gca()
ax.spines['top'].set_color('none')
ax.spines['left'].set_position('zero')
ax.spines['right'].set_color('none')
ax.spines['bottom'].set_position('zero')

#Zeros
dzeros = fsolve(real_df, D)
print("Os zeros da derivada são: ", dzeros)

#Monotonia
if z>0 :
   print("A Função f é crescente em ]{},{}] e decrescente em [{}, {}[".format(dze-
ros,D,d,dzeros))
else:
   print("A Função f é crescente em [{},{}[ e decrescente em ]{}, {}]".format(d,dze-
ros,dzeros,D))

plt.plot(x,j, label="f(x)")
plt.plot(x,y, label="df(x)")
plt.legend()
plt.show()
else:
    print("f(x)= ax^3 + bx^2 + cx + d")
    a = int(input("a="))
    b = int(input("b="))
    c = int(input("c="))
    d = int(input("d="))
    print("f(x) = {}x^3 + {}x^2 + {}x + {}".format(a,b,c,d))
    v=3*a
    z=2*b
    print("A derivada de f(x) é df(x) = {}x^2 {}x + {}".format(v,z,c))

def f(x) :
    return(a*x**3 + b*x**2 + c*x + d)

def df(x) :
    h = 0.00000000001
    fun = (f(x+h)-f(x))/(h)
    return float("%.3f" %fun)
```

```
def real_df(x) :
    h = 0.00000000001
    fun = (f(x+h)-f(x))/(h)
    return(fun)

#Grafico

#Dominio
d = int(input("Limite inferior do domínio da função é "))
D = int(input("Limite superior do domínio da função é "))

#Funções
x = np.linspace(d,D,100)
y = real_df(x)
j = f(x)
#Eixos
ax = plt.gca()
ax.spines['top'].set_color('none')
ax.spines['left'].set_position('zero')
ax.spines['right'].set_color('none')
ax.spines['bottom'].set_position('zero')

#Zeros
mzero = fsolve(real_df,d)
Mzero = fsolve(real_df,D)
print("Os zeros da derivada são: {} e {}".format(mzero,Mzero))

#Monotonia
if v>0 :
   print("A Função f é crescente em ]{},{}] e [{}, {}[ e decrescente em [{}, {}]".for-
mat(d,mzero,Mzero,D,mzero,Mzero))
else:
   print("A Função f é crescente em [{},{}] e decrescente em ]{},{}] e [{}, {}[".for-
mat(mzero,Mzero,d,mzero,Mzero,D))

plt.plot(x,j, label="f(x)")
plt.plot(x,y, label="df(x)")
plt.legend()
plt.show()

print("Obrigado!")
exit()
```

Estudo de caso 11 – Triângulo de Pascal

```
def desenhar_triangulo_pascal(input_num):
    print(" ")
    print ("Desenhar triângulo de pascal")
    print(" ")
    list = [] #lista vazia
    for n in range(input_num):
        list.append([])
        list[n].append(1)
        for m in range(1, n):
            list[n].append(list[n - 1][m - 1] + list[n - 1][m])
        if(input_num != 0):
            list[n].append(1)

#imprimir lista
for n in range(input_num):
   print(" " * (input_num - n), end = " ", sep = " ")
   for m in range(0, n + 1):
      print('{0:5}'.format(list[n][m]), end = " ", sep = " ")
   print()

def elemento(linha,coluna):
    print(" ")
    print ("Identificar um elemento do triângulo de pascal")
    print(" ")
    list = [] #lista vazia
    for n in range(linha):
        list.append([])
        list[n].append(1)
        for m in range(1, n):
            list[n].append(list[n - 1][m - 1] + list[n - 1][m])
        if(linha != 0):
            list[n].append(1)
            print(list[n])
            print(n)

#print (list[linha][coluna+1])
return(list[n][coluna])

def soma_linha(linha):
    print(" ")
    print ("somar os elementos de uma linha do triângulo de pascal")
```

```
    print(" ")
    return(2**linha)

def soma_anteriores_linha(linha):
    print(" ")
    print ("Somar todos os elementos anteriores a uma linha")
    print(" ")
    soma = 0
    for x in range(linha-1):
      soma = soma + 2**x
    return(soma)

op_lista =["1-Listar triângulo de pascal", "2-Identificar um elemento do
triâgulo de pascal", "3-somar os elementos de uma linha do triângulo de
pascal", "4-somar todos os elementos anteriores a uma linha no triângulo"]

while (True):
    print(" ")
    print(" ")
    print("1-Listar triângulo de pascal")
    print("2-Identificar um elemento do triângulo de pascal")
    print("3-somar os elementos de uma linha do triângulo de pascal")
    print("4-somar todos os elementos anteriores a uma linha no triângulo")
    op = 0
    op_int = int (op)
    while (op_int < 1 or op_int > 4):
       op = input('Introduza uma opção [1 a 4]:')
       op_int = int (op)
    print(" ")
    print (op_lista[op_int-1])
    print(" ")

    if op_int == 1:
        input_num = int(input("Insira o número de linhas: "))
        desenhar_triangulo_pascal(input_num)
    elif op_int == 2:
        linha = int(input('linha:'))
        coluna = 0
        while (coluna < 1 or coluna > linha + 1):
               coluna = int(input('coluna:' + 'entre[1 e ' + str(linha+1)
+']:'))
        elem = elemento(linha+1,coluna-1)
        print(" ")
```

```
        print(elem)
        print(" ")
    elif op_int == 3:
        linha = int(input('linha:'))
        soma_linha = soma_linha(linha)
        print(" ")
        print('A soma dos elementos da linha ' + str(linha) + ' = ' +
str(soma_linha))
        print(" ")
    else:
        linha = int(input('linha:'))
        soma = soma_anteriores_linha(linha)
        print(" ")
        print('A soma de todos elementos das linhas anteriores à linha ' +
str(linha) + ' = ' + str(soma))
        print(" ")
```

Estudo de caso 12 – Interseções

```
print("Olá, bem vindo à nossa aplicação de python. Nesta aplicação poderás calcular
as interseções de um modo muito mais fácil!!!")
print("É só introduzires os valores das equações das retas ou planos que quiseres e
terás o resultado!!! Fácil certo?!")
print("Aqui tens a lista de comanados para usares:")
print("Para a interseção entre duas retas no plano introduz rr")
print("Para a interseção entre duas retas no espaço introduz rresp")
print("Para a interseção entre uma reta e um plano introduz rp")
print("Para a interseção entre dois planos introduz pp")
print("Para começares o programa, escreve - iniciar().")
print("Depois de iniciares a primeira vez, já não o precisas de fazer novamente")
print("Esperemos que te facilite a vida e que te divirtas a usar esta aplicação!!")
print(" ")
def iniciar():
   print("Que interseção queres determinar?")
   pergunta=input()
   if (pergunta=="rr"):
      print("Escreve a equação das retas na forma:")
      print("y = a(x)+ b")
      a1=float(input("declive da reta 1 - "))
      b1=float(input("ordenada na origem da reta 1 - "))
      a2=float(input("declive da reta 2 - "))
      b2=float(input("ordenada na origem da reta 2 - "))
      if a1==a2:
          if b1==b2:
             print("As retas são coincidentes")
             print(" ")
             iniciar()
          else:
             print("As retas são paralelas")
             print(" ")
             iniciar()
             print(" ")
      else:
          x=(b2-b1)/(a1-a2)
          y=(a1*x+b1)
          print("As retas intersetam-se no ponto de coordenadas:{:.2f};{:.2f}.".for-
mat(x,y))
          print(" ")
          iniciar()
          print(" ")
   elif (pergunta=="rresp"):
       print("Escreve a equação das retas na forma:")
       print("(x;y;z)=(x;y;z)+k(Vx;Vy;Vz)")
       x1=float(input("abcissa do ponto da reta 1 - "))
       y1=float(input("ordenada do ponto da reta 1 - "))
```

```
        z1=float(input("cota do ponto da reta 1 - "))
        Vx1=float(input("coordenada x do vetor da reta 1 - "))
        Vy1=float(input("coordenada y do vetor da reta 1 - "))
        Vz1=float(input("coordenada z vetor da reta 1 - "))
        x2=float(input("abcissa do ponto da reta 2 - "))
        y2=float(input("ordenada do ponto da reta 2 - "))
        z2=float(input("cota do ponto da reta 2 - "))
        Vx2=float(input("coordenada x do vetor da reta 2 - "))
        Vy2=float(input("coordenada y do vetor da reta 2 - "))
        Vz2=float(input("coordenada z do vetor da reta 2 - "))
        if Vx2!=0:
           k=Vx1/Vx2
           if Vx2*k==Vx1 and Vy2*k==Vy1 and Vz2*k==Vz1:
              print("As retas são paralelas")
              print(" ")
              iniciar()
              print(" ")
           else:
               k1=((Vx1*y1)-(Vx1*y2)+(Vy1*x2)-(Vy1*x1))/((Vy2*Vx1)-(Vy1*Vx2))
               k2=((Vx1*z1)-(Vx1*z2)+(Vz1*x2)-(Vz1*x1))/((Vz2*Vx1)-(Vz1*Vx2))
               k3=((Vy1*z1)-(Vy1*z2)+(Vz1*y2)-(Vz1*y1))/((Vz2*Vy1)-(Vz1*Vy2))
               P1=x2+(k1*Vx2)
               P2=y2+(k1*Vy2)
               P3=z2+(k1*Vz2)
               if k1==k2==k3:
                  print("As duas retas intersetam-se no ponto de coordena-
das:({:.2f};{:.2f};{:.2f}).".format(P1,P2,P3))
                  print(" ")
                  iniciar()
                  print(" ")
               else:
                  print("As retas são enviesadas")
                  print(" ")
                  iniciar()
                  print(" ")
        elif Vx2==0 and Vy2!=0:
            k=Vy1/Vy2
            if Vx2*k==Vx1 and Vy2*k==Vy1 and Vz2*k==Vz1:
               print("As retas são paralelas")
               print(" ")
               iniciar()
               print(" ")
            else:
               k1=((Vx1*y1)-(Vx1*y2)+(Vy1*x2)-(Vy1*x1))/((Vy2*Vx1)-(Vy1*Vx2))
               k2=((Vx1*z1)-(Vx1*z2)+(Vz1*x2)-(Vz1*x1))/((Vz2*Vx1)-(Vz1*Vx2))
               k3=((Vy1*z1)-(Vy1*z2)+(Vz1*y2)-(Vz1*y1))/((Vz2*Vy1)-(Vz1*Vy2))
               P1=x2+(k1*Vx2)
               P2=y2+(k1*Vy2)
```

```
            P3=z2+(k1*Vz2)
            if k1==k2==k3:
               print("As duas retas intersetam-se no ponto de coordena-
das:({:.2f};{:.2f};{:.2f}).".format(P1,P2,P3))
               print(" ")
               iniciar()
               print(" ")
            else:
                print("As retas são enviesadas")
                print(" ")
                iniciar()
                print(" ")
      elif Vx2==0 and Vy2==0 and Vz2!=0:
          k=Vz1/Vz2
          if Vx2*k==Vx1 and Vy2*k==Vy1 and Vz2*k==Vz1:
             print("As retas são paralelas")
             print(" ")
             iniciar()
             print(" ")
          else:
             k1=((Vx1*y1)-(Vx1*y2)+(Vy1*x2)-(Vy1*x1))/((Vy2*Vx1)-(Vy1*Vx2))
             k2=((Vx1*z1)-(Vx1*z2)+(Vz1*x2)-(Vz1*x1))/((Vz2*Vx1)-(Vz1*Vx2))
             k3=((Vy1*z1)-(Vy1*z2)+(Vz1*y2)-(Vz1*y1))/((Vz2*Vy1)-(Vz1*Vy2))
             P1=x2+(k1*Vx2)
             P2=y2+(k1*Vy2)
             P3=z2+(k1*Vz2)
             if k1==k2==k3:
                print("As duas retas intersetam-se no ponto de coordena-
das:({:.2f};{:.2f};{:.2f}).".format(P1,P2,P3))
                print(" ")
                iniciar()
                print(" ")
             else:
                 print("As retas são enviesadas")
                 print(" ")
                 iniciar()
                 print(" ")

   elif (pergunta=="rp"):
        print("Escreve a equação da reta na forma:")
        print("(x;y;z)=(x;y;z)+k(Vx;Vy;Vz)")
        print("Escreve a equação do plano na forma:")
        print("a(x)+b(y)+c(z)+d=0")
        x1=float(input("abcissa do ponto da reta - "))
        y1=float(input("ordenada do ponto da reta - "))
        z1=float(input("cota do ponto da reta - "))
        Vx1=float(input("coordenada x do vetor da reta - "))
        Vy1=float(input("coordenada y do vetor da reta - "))
```

```
        Vz1=float(input("coordenada z vetor da reta - "))
        x2=float(input("coordenada a vetor do plano - "))
        y2=float(input("coordenada b vetor do plano - "))
        z2=float(input("coordenada c vetor do plano - "))
        d=float(input("coordenada d do plano - "))
        if (Vx1*x2)+(Vy1*y2)+(Vz1*z2)==0:
           if (x1*x2)+(y1*y2)+(z1*z2)+d==0:
              print("A reta pertence ao plano.")
              print(" ")
              iniciar()
              print(" ")
           else:
              print("A reta é paralela ao plano.")
              print(" ")
              iniciar()
              print(" ")
        else:
            k=-((x2*x1)+(y2*y1)+(z2*z1)+d)/((x2*Vx1)+(y2*Vy1)+(z2*Vz1))
            P1=x1+(Vx1*k)
            P2=y1+(Vy1*k)
            P3=z1+(Vz1*k)
            print("A reta e o plano intersetam-se no ponto de coordena-
das:({:.2f};{:.2f};{:.2f}).".format(P1,P2,P3))
            print(" ")
            iniciar()
            print(" ")

   elif(pergunta=="pp"):
        print("Escreve as equações dos planos na forma:")
        print("a(x)+b(y)+c(z)+d=0")
        x1=float(input("coordenada a vetor do plano 1 - "))
        y1=float(input("coordenada b vetor do plano 1 - "))
        z1=float(input("coordenada c vetor do plano 1 - "))
        d1=float(input("coordenada d do plano 1 - "))
        x2=float(input("coordenada a vetor do plano 2 - "))
        y2=float(input("coordenada b vetor do plano 2 - "))
        z2=float(input("coordenada c vetor do plano 2 - "))
        d2=float(input("coordenada d do plano 2 - "))
        if x2!=0:
           k=x1/x2
           if x2*k==x1 and y2*k==y1 and z2*k==z1:
              if d2*k==d1:
                 print("Os planos são coincidestes (são o mesmo plano).")
                 print(" ")
                 iniciar()
                 print(" ")
              else:
                 print("Os planos são paralelos.")
```

```
                print(" ")
                iniciar()
                print(" ")
            else:
                Z11=((y2*d1)-(d2*y1))/((z2*y1)-(z1*y2))
                Y11=(-(z1*Z11)-d1)/y1
                Z22=((x1*y2)+(d1*y2)-(x2*y1)-(d2*y1))/((z2*y1)-(z1*y2))
                Y22=(-x1-(z1*Z22)-d1)/y1
                P2=Y22-Y11
                P3=Z22-Z11
                print("Os planos intersetam-se na reta de equação:
(x;y;z)=(1;{:.2f};{:.2f})+k(1;{:.2f};{:.2f}).".format(Y22,Z22,P2,P3))
                print(" ")
                iniciar()
                print(" ")
        elif y2!=0 and x2==0:
             k=y1/y2
             if x2*k==x1 and y2*k==y1 and z2*k==z1:
                if d2*k==d1:
                   print("Os planos são coincidestes (são o mesmo plano).")
                   print(" ")
                   iniciar()
                   print(" ")
                else:
                   print("Os planos são paralelos.")
                   print(" ")
                   iniciar()
                   print(" ")
             else:
                 Z11=((y2*d1)-(d2*y1))/((z2*y1)-(z1*y2))
                 Y11=(-(z1*Z11)-d1)/y1
                 Z22=((x1*y2)+(d1*y2)-(x2*y1)-(d2*y1))/((z2*y1)-(z1*y2))
                 Y22=(-x1-(z1*Z22)-d1)/y1
                 P2=Y22-Y11
                 P3=Z22-Z11
                 print("Os planos intersetam-se na reta de equação:
(x;y;z)=(1;{:.2f};{:.2f})+k(1;{:.2f};{:.2f}).".format(Y22,Z22,P2,P3))
                 print(" ")
                 iniciar()
                 print(" ")
        elif z2!=0 and x2==0 and y2==0:
             k=z1/z2
             if x2*k==x1 and y2*k==y1 and z2*k==z1:
                if d2*k==d1:
                   print("Os planos são coincidestes (são o mesmo plano).")
                   print(" ")
                   iniciar()
                   print(" ")
```

```
                else:
                    print("Os planos são paralelos.")
                    print(" ")
                    iniciar()
                    print(" ")
            else:
                Z11=((y2*d1)-(d2*y1))/((z2*y1)-(z1*y2))
                Y11=(-(z1*Z11)-d1)/y1
                Z22=((x1*y2)+(d1*y2)-(x2*y1)-(d2*y1))/((z2*y1)-(z1*y2))
                Y22=(-x1-(z1*Z22)-d1)/y1
                P2=Y22-Y11
                P3=Z22-Z11
                print("Os planos intersetam-se na reta de equação:
(x;y;z)=(1;{:.2f};{:.2f})+k(1;{:.2f};{:.2f}).".format(Y22,Z22,P2,P3))
                print(" ")
                iniciar()
                print(" ")
    else:
        print("Comando não reconhecido.")
        print(" ")
        iniciar()
```