UNIVERSIDADE
LUSÓFONA

Centro Universitário de Lisboa

Faculdade de Ciências Sociais, Educação e Administração

Instituto de Educação

Mestrado em Ensino de Informática

O Ensino das Funções em linguagem C através do ambiente de programação Dev C++

Relatório da Prática de Ensino Supervisionada apresentado a provas públicas para obtenção do grau de Mestre em Ensino de Informática, orientado pelos professores Vítor Teodoro e Margarida Batista.

João Alberto Fontinha de Brito

N.º a22207257

2025

UNIVERSIDADE LUSÓFONA

Centro Universitário de Lisboa

Faculdade de Ciências Sociais, Educação e Administração

Instituto de Educação

Mestrado em Ensino de Informática

O Ensino das Funções em linguagem C através do ambiente de programação Dev C++

VERSÃO FINAL

Relatório da Prática de Ensino Supervisionada defendido em provas públicas na Universidade Lusófona, Centro Universitário de Lisboa no dia 9 de julho de 2025, perante o júri, nomeado pelo Despacho de Nomeação nº 372 de 2025, nomeado a 27 de junho de 2025, com a seguinte composição:

Presidente: Professor Doutor João Filipe Matos

Vogal: Professora Doutora Cláudia Barata (Universidade Lusófona de Humanidades e Tecnologias) – Arguente

Orientador: Professor Doutor Vítor Teodoro.

João Alberto Fontinha de Brito – N.º a22207257

2025

Agradecimentos

O relatório da prática de ensino supervisionada representa a conclusão de uma etapa académica, mas também reflete o apoio e o incentivo de diversas pessoas que, direta ou indiretamente, contribuíram para a sua realização. Em primeiro lugar, gostaria de expressar a minha profunda gratidão ao professor orientador da Universidade Lusófona, o professor Vítor Teodoro, por todo o acompanhamento prestado, pela sua orientação precisa e pelo incentivo ao longo deste percurso. O seu conhecimento e disponibilidade foram essenciais e uma mais-valia para a concretização deste trabalho. Agradeço de forma especial à professora cooperante da Escola Secundária de Santo André, a professora Margarida Batista, pelo apoio inestimável, pela partilha de conhecimentos e por toda a orientação prática, fatores que contribuíram significativamente não só para o meu desenvolvimento pessoal, mas também para o meu crescimento profissional e pedagógico. Aos professores do Mestrado em Ensino de Informática da Universidade Lusófona, que na partilha de experiências e conhecimentos durante estes dois anos de Mestrado enriqueceram todo o meu percurso académico, deixo igualmente o meu agradecimento. Um agradecimento especial à Escola Secundária de Santo André, nomeadamente à direção, aos professores do agrupamento e aos alunos da turma 1.ºL, por me acolherem durante a prática supervisionada e por proporcionarem um ambiente de aprendizagem onde pude aplicar e aperfeiçoar as minhas competências pedagógicas. Aos meus amigos e familiares, o meu sincero agradecimento por todo o apoio, paciência e compreensão, especialmente nos momentos mais exigentes. Sem o vosso suporte, este percurso teria sido muito mais difícil.

A todos, um sincero obrigado!

Resumo

O presente relatório, desenvolvido no âmbito do Mestrado em Ensino de Informática, descreve a intervenção pedagógica realizada durante as aulas Práticas Supervisionadas na Escola Secundária de Santo André. A investigação focou-se na motivação dos alunos, no ensino da programação em C e na utilização do Dev C++ aliado à Inteligência Artificial. A prática supervisionada decorreu numa turma do 10.º ano do curso profissional de Técnico de Informática – Sistemas, composta maioritariamente por alunos do sexo masculino, alguns com dificuldades de aprendizagem e pouca experiência em programação. Com base num diagnóstico inicial, foram desenvolvidas estratégias pedagógicas adaptadas ao perfil da turma, combinando métodos expositivos, atividades práticas e ferramentas de Inteligência Artificial como auxiliares da aprendizagem. As aulas foram organizadas progressivamente, desde os conceitos básicos da programação modular até à criação e implementação de funções em C. Durante toda a intervenção, foi promovida a autonomia dos alunos, o trabalho prático, o diálogo e o método interrogativo, recorrendo-se a situações e exemplos reais, mantendo a interação entre professor e alunos. A metodologia incluiu questionários para avaliar a motivação e evolução das aprendizagens dos alunos antes e depois da prática supervisionada. Os resultados indicaram melhoria significativa na compreensão e aplicação dos conceitos, apesar das dificuldades iniciais na transição para um modelo mais autónomo. O uso do Dev C++ e da IA promoveu a interação com os conteúdos e facilitou a compreensão dos erros. Houve um crescimento da motivação dos alunos ao longo das aulas, evidenciado pelo aumento da participação e interesse nas atividades. A utilização de metodologias ativas e tecnologias emergentes, como a IA, mostraram-se eficazes no ensino da programação, promovendo o pensamento computacional e facilitando a aprendizagem. O estudo destaca a importância de estratégias inovadoras para um ensino mais dinâmico e adaptado às necessidades dos alunos.

Palavras-Chave: Motivação, Programação em C, Metodologias de Ensino, Tecnologias Emergentes, Estratégias de ensino.

Índice

Índice de Figuras

Índice de Tabelas

Introdução

O estudo realizado insere-se no âmbito do Mestrado em Ensino da Informática da Universidade Lusófona de Lisboa e faz parte da intervenção pedagógica da prática supervisionada na unidade curricular da disciplina de Iniciação à Prática Profissional IV. O objetivo principal deste relatório é descrever todas as atividades, aulas, observações e participações em reuniões e responsabilidades atribuídas na Escola Secundária de Santo André durante o período de estágio, entre setembro de 2024 e abril de 2025. A prática pedagógica supervisionada estará enquadrada na turma de 1.º ano do curso Profissional de Técnico de Informática - Sistemas, com uma carga total letiva de 3125 horas, distribuídas pela componente sociocultural, científica, tecnológica e formação em contexto de trabalho. A disciplina de Programação é uma disciplina da componente tecnológica que, no primeiro ano, conta com quatro módulos, 200 horas, distribuídas em 50 horas por cada módulo, somando um total de 240 aulas letivas. Os alunos irão aprender conhecimentos em Linguagem de Programação C/C++, onde abordarão os tópicos sobre a estrutura básica e os conceitos fundamentais, ciclos e decisões, funções e estruturas e formas complexas. A prática supervisionada incidirá na unidade 0784 – Programação em C/C++ - funções e estruturas, particularmente no que diz respeito a funções.

O curso faz parte do Catálogo Nacional de Qualificações (ANQEP), sendo um curso de nível 4 com o código 481039 da área de ciências informáticas. Este curso é de dupla certificação e prepara os jovens para o mercado de trabalho, permitindo a realização dos estudos ao nível do ensino secundário. Este tipo de cursos é adequado para aqueles alunos que procuram formações mais práticas, dando-lhes a oportunidade de se prepararem melhor para o mercado de trabalho. Terminam com uma apresentação oral e defesa perante um júri de um projeto denominado “Prova de Aptidão Profissional” (PAP), onde os alunos demonstram as capacidades, competências e conhecimentos adquiridos ao longo do curso, com particular incidência na componente técnica/tecnológica. Esta prova é composta por um produto, relatório e apresentação, de acordo com a Portaria n.º 235-A, de 23 de agosto de 2018. Ao longo da

intervenção pedagógica supervisionada, será efetuada uma observação direta da participação e evolução das aprendizagens dos alunos.

A investigação decorreu no contexto da prática pedagógica supervisionada, ao longo de oito aulas com a duração de 50 minutos cada, com início a 4 de fevereiro de 2025 e término a 11 de fevereiro de 2025. A investigação envolveu a aplicação de dois questionários à turma do 1.º L na disciplina de Programação. Estes alunos estiveram em contacto com a programação em contexto curricular pela primeira vez. O objetivo é caracterizar o perfil, a motivação, aferir conhecimentos e aprendizagens ao longo das aulas de programação em linguagem C.

Com base na aferição das aprendizagens dos alunos, foram definidas e implementadas estratégias pedagógicas durante a prática supervisionada, tendo em conta o perfil e a motivação da turma. Inicialmente, as aulas tiveram um contexto mais teórico, incidindo fundamentalmente na programação modular, através de uma abordagem do diálogo entre professor e aluno, de forma a manter um feedback contínuo, permitindo percecionar as aprendizagens e as dificuldades apresentadas. Posteriormente, foram introduzidos exercícios que conduzem a uma abordagem mais prática, com o objetivo de desenvolver a autonomia e o trabalho de pares, de modo a potenciar a motivação e a aprendizagem da linguagem em C. Durante as aulas práticas, será realizado um exercício (criação de uma calculadora com as quatro operações aritméticas básicas) com o objetivo de certificar se a Inteligência Artificial é facilitadora durante o processo de aprendizagem da linguagem C.

1 Caracterização do Agrupamento

A caracterização do agrupamento de escolas tem como objetivo fornecer uma visão abrangente do contexto educativo da instituição escolar, permitindo assim compreender a sua estrutura, funcionamento e impacto na comunidade escolar. Este relatório descreve os aspetos essenciais sobre como o agrupamento administra os recursos disponíveis, apresenta a oferta formativa, o perfil dos alunos e dos professores bem como as dinâmicas pedagógicas e os projetos educativos em vigor. A caracterização do agrupamento permitiu identificar os desafios e as oportunidades, contribuindo assim para o desenvolvimento de estratégias que promovam uma educação inclusiva e eficiente. Também é possível analisar indicadores de sucesso e desempenho assim como o abandono escolar, facilitando a implementação de medidas de melhoria. Através da caracterização do agrupamento podemos refletir e tomar decisões que visam a melhoria do processo educativo e promover assim um ambiente de aprendizagem mais enriquecedor e eficaz.

1.1 O Agrupamento de Escolas Santo André

O Agrupamento de Escolas de Santo André (AESA) foi criado no ano de 2013 e é composto por quatro estabelecimentos de ensino:

- Jardim Infantil Bairro 25 de Abril – Ensino do Pré-escolar.
- Escola Básica da Telha Nova – Ensino do Pré-escolar e 1.º Ciclo.
- Escola Básica da Quinta da Lomba – Ensino dos 2.º e 3.º Ciclos.
- Escola Secundária de Santo André - Ensino dos Cursos Científico-Humanísticos e Cursos Profissionais.

O agrupamento encontra-se inserido na União das Freguesias do Alto do Seixalinho em Santo André e Verderena. O agrupamento oferece também uma educação e formação de adultos dispondo ainda de um Centro Qualifica para o ensino noturno. Este agrupamento público caracteriza-se pela sua diversidade e multiculturalidade. O ensino secundário, disponibiliza as quatro áreas dos cursos científico-humanísticos e duas turmas de ensino profissional, sendo

uma delas na área de Informática. A partir de 2016, o AESA incorporou o projeto "Iniciação à Programação no 1.º ciclo" para os alunos dos 3.º e 4.º anos, e mais recentemente, a nível do 3.º ciclo, introduziu atividades ligadas à expressão artística. No ensino noturno, são oferecidos cursos de educação e formação de adultos, unidades de formação de curta duração e cursos de Português para alunos com dificuldades.

O agrupamento conta com várias equipas profissionais:

- 209 professores.
- 4 psicólogas.
- 13 assistentes técnicos.
- 73 assistentes operacionais.

O agrupamento ainda conta com:

- Um total de 2270 alunos sendo a maioria residente no concelho.
- Alunos de países dos PALOP (Brasil, Ucrânia, China, Paquistão e Venezuela).

1.1.1 Distribuição dos alunos pelos diversos ciclos

Na (**Tabela 1**) é possível analisar o número total de alunos distribuídos pelos diversos ciclos. A tabela não contabiliza o número total de alunos que frequentam o Centro Qualifica do AESA.

Tabela 1
Distribuição do n.º de alunos pelos ciclos de estudo

Grau de ensino	Número de alunos abrangidos por apoio regular de SE
Pré-escolar	73
1.º ciclo do Ensino Básico	378
2.º ciclo do Ensino Básico	259
3.º ciclo do Ensino Básico	423
Secundário	998
Outros	0
Total	**2131**

O agrupamento conta com alunos com necessidades educativas:

- Existem na escola aproximadamente 136 alunos com necessidades educativas que recebem apoio personalizado, com recurso a materiais e equipamentos adaptados, através do grupo de Educação Especial.
- Até ao momento, no agrupamento, há 136 alunos que beneficiam de medidas ao abrigo do Dec.-Lei 54/2018, dos quais 43 têm medidas seletivas e adicionais.
- As respostas que o agrupamento dá a estes alunos são as medidas que se aplicam através do RTP/PEI em cada disciplina e que o respetivo docente define para o aluno.
- Além destas medidas, e para permitir ao aluno ultrapassar algumas barreiras, recorre-se ainda às valências existentes na escola, que podem resolver situações em concreto. Tais como clubes, projetos da escola e todos os espaços da escola que contribuam para a melhoria das suas aprendizagens.

A (**Tabela 2**) seguinte apresenta a distribuição de alunos com necessidades educativas pelos diversos ciclos de aprendizagem:

Tabela 2
N.º de alunos com necessidades educativas

Alunos com Necessidades educativas 2024/2025			
	N.º de alunos	**Medidas Seletivas**	**Medidas adicionais**
Pré-Escolar	8	8	0
1.º ciclo	37	37	16
1.º ano	9	9	5
2.º ano	10	10	4
3.º ano	11	11	4
4.º ano	7	7	3
2.º ciclo	26	26	11
5.º ano	13	13	7
6.º ano	13	13	4
3.º ciclo	40	40	9
7.º ano	16	16	5
8.º ano	14	14	2
9.º ano	10	10	2
Secundário	35	35	7
10.º ano CCH	8	8	1
10.º ano CP	8	8	4
11.º ano CCH	5	5	1
11.º ano CP	9	9	0
12.º ano CCH	3	3	0
12.º ano CP	2	2	1
TOTAL	**136**	**136**	**43**

O AESA promove uma formação ampla para os alunos através de diversos Clubes e Projetos, no entanto, os professores também têm acesso a iniciativas como o projeto "Observisão", que incentiva a supervisão pedagógica entre pares e promove a colaboração e reflexão conjunta. O agrupamento já foi distinguido várias vezes, tendo recebido o reconhecimento como Escola *eTwinning* em 2018/2019 e 2021/2022, com Selos Europeus de Qualidade para vários dos seus projetos. O AESA caracteriza-se como um ambiente seguro e acolhedor, onde alunos, professores, funcionários e encarregados de educação se sentem integrados e respeitados. A direção do agrupamento trabalha continuamente na melhoria das condições das escolas, mantendo uma relação positiva e colaborativa com a comunidade.

Também foi premiado com o Selo de "Boas Práticas" Erasmus+ pelo projeto KA1 "*Learning 4 Future*", além de ter sido recentemente distinguido com o Selo de Ouro de Segurança Digital (SeguraNet), o Selo da Escola Saudável e o Selo da Escola Parceira "EUSOUDIGITAL". Na (**Figura 1**) é apresentada uma imagem do Agrupamento de Escolas de Santo André, local onde foi realizado o estágio da disciplina Introdução à Prática Profissional IV.

Figura 1
Entrada da Escola

1.1.2 Internacionalização do Agrupamento

Há mais de vinte anos que o Agrupamento de Escolas de Santo André tem estado envolvido em projetos internacionais, com a direção do agrupamento a incentivar este tipo de iniciativas. Estes projetos têm sido reconhecidos tanto a nível nacional como internacional, recebendo prémios tais como o Prémio de Boas Práticas da Agência Nacional Erasmus e o Prémio Ibero-americano de Ensino em Direitos Humanos, Oscar Arnulfo Romero, da OEI (Organização dos Estados Ibero-Americanos).

Atualmente, estão em andamento nove projetos Erasmus+, que envolve crianças da educação pré-escolar e jovens do ensino básico e secundário. Recentemente, o Agrupamento também obteve a acreditação Erasmus+ no setor do ensino profissional.

A participação nestes projetos tem sido muito benéfica para promover a inclusão e a equidade, proporcionando a alunos, professores e funcionários a oportunidade de conhecer

outras culturas, assim como melhorar as competências linguísticas e fortalecer a cidadania democrática na escola a nível europeu.

1.1.3 Projetos do Agrupamento

- Parlamento dos Jovens.
- Horta para a cozinha.
- Plano Nacional de Leitura.
- Projeto *WORD* - Um Universo de Aprendizagem.
- Aprender a ser feliz com a floresta.
- SPO (Serviço de Psicologia e Orientação).
- Desporto Escolar.
- PES (Projeto Educação para a Saúde).
- Plano Nacional de Cinema.
- Iniciação à Programação e Robótica no 1º Ciclo do Ensino Básico.
- Laboratórios Educativos Digitais (LED).

1.1.4 Clubes do Agrupamento

- Clube de Teatro da (EBQL).
- Clube da Matemática -Jogos de Lógica.
- Clube de Proteção Civil.
- Oficina Criativa.
- Oficina de Jardinagem.
- Pop-Rock Clube.
- Escola de Teatro da ESSA.
- Infor-code-club.

- Clube Rádio Escola.
- Ornitho AESA.
- Clube GeoAventura.
- Clube de Música da ESSA.
- Clube Ciência Viva na Escola (*Experimentarium*).

1.1.5 A Oferta formativa do Agrupamento

O agrupamento conta com uma vasta oferta formativa atualizada para os diversos níveis de ensino sendo eles discriminados da seguinte forma:

- Educação Pré-escolar: O agrupamento disponibiliza Atividades de Enriquecimento Curricular (AEC) e Atividades de Apoio à Família (AAF).
- 1.º Ciclo do Ensino Básico: Introdução à Programação no 1.º Ciclo, AEC e AAF.
- 3.º Ciclo do Ensino Básico (7.º ano): Oferta de língua estrangeira (Alemão, Espanhol, Francês e Inglês) e Complemento de Expressão Artística.
- Ensino Secundário:
    - Cursos Científico-Humanísticos.
    - Ciências e Tecnologias.
    - Ciências Socioeconómicas.
    - Línguas e Humanidades.
    - Artes Visuais.
- Cursos Profissionais:
    - Técnico de Audiovisuais.
    - Técnico de Design de Comunicação Gráfica.
    - Técnico de Gestão e Programação de Sistemas Informáticos.
    - Técnico de Informática – Instalação e Gestão de Redes.

- Técnico de Informática – Sistemas.

- Oferta Formativa para Adultos:
  - Educação e Formação de Adultos (EFA), com modalidades Tipo A, Tipo B e Tipo C.
  - Português Língua de Acolhimento (PLA).

1.1.6 O Centro Qualifica do AESA

O Agrupamento conta com este centro de formação desde o ano de 2017/18. Estes centros são especializados na qualificação de adultos com o objetivo de fornecer uma qualificação escolar para adultos com idade igual ou superior aos 18 anos. De referir que este centro se encontra preparado para receber adultos portadores de deficiência e incapacidade, com o intuito de assegurar a sua integração na vida ativa e profissional.

- Atividades do Centro Qualifica da ESSA:
  - RVCC – Reconhecimento, Validação e Certificação de Competências.
  - Cursos de curta duração: Folha de cálculo – funcionalidades avançadas (25h), Língua Inglesa (50h), Processador de Texto – funcionalidades avançadas (25h), entre outros.

2 Caracterização da Turma onde decorreu a prática letiva

Caracterizar a turma a lecionar tem como objetivo principal recolher informações detalhadas sobre os alunos da turma, permitindo a aplicação de estratégias eficazes de perfil, necessidades individuais e níveis de conhecimento dos alunos da turma. Através desta análise, é possível compreender a composição da turma, incluindo o número total de alunos, idades, percurso escolar, dificuldades de aprendizagem, níveis de motivação, dificuldades económicas e interesse pela disciplina. Além disso, caracterizar a turma possibilita a adequação dos métodos de ensino, tendo em consideração a sala de aula e recursos disponíveis.

2.1 Composição da turma

A turma é composta por 28 alunos, maioritariamente do sexo masculino (27 rapazes e 2 raparigas). A média de idades da turma ronda os 15 anos de idade. Seis alunos são estrangeiros e recebem apoio à aprendizagem da língua portuguesa como a língua não materna. A distribuição entre os dois cursos técnico-profissionais demonstra um foco e interesse na área tecnológica. Há um equilíbrio entre os dois percursos técnico-profissionais. Nesta turma metade dos alunos são pertencentes aos cursos Técnico-Profissionais de Informática – Sistemas e a outra metade pertencente ao curso de Gestão e Programação de Sistemas Informáticos. A prática pedagógica supervisionada, incidirá sobre o curso Técnico de Informática – Sistemas frequentado por 14 alunos. É importante referir que existe também na turma um aluno com Relatório Técnico Pedagógico (RTP), necessitando de apoios pedagógicos específicos.

2.2 Perfil do aluno à saída da escolaridade obrigatória

O aluno quando completa com sucesso o curso profissional, desenvolveu competências a nível de linguagens e texto, informação e comunicação, raciocínio e resolução de problemas, pensamento critico e criativo, relacionamento interpessoal, desenvolvimento pessoal e autonomia, bem-estar, saúde e ambiente, sensibilidade estética e artística, saber científico, técnico e tecnológico, consciência e domínio do corpo. Estas competências são essenciais para

o aluno poder enfrentar o mercado de trabalho ou até mesmo prosseguir os estudos. O plano de desenvolvimento da Intervenção Pedagógica encontra-se de acordo com o Despacho n. º 6605-A/2021, de 6 de julho.

2.3 Perfil do aluno à saída do curso

De acordo com o Catálogo Nacional de Qualificações o perfil do aluno à saída do curso deve-lhe permitir ser capaz de efetuar instalações, configurações, prestar manutenção de ferramentas, equipamentos e sistemas informáticos, suportados em diferentes plataformas e sistemas operativos, e proceder à gestão e administração de base de dados e ao desenvolvimento de *software*, assegurando a otimização do seu funcionamento, respeitando as normas de segurança, higiene e saúde no trabalho e de proteção do ambiente.

2.4 Estratos sociais dos encarregados de Educação do curso Técnico de Informática – Sistemas

Situação de Emprego:

- A maior parte dos encarregados de educação trabalha por conta de outrem (6 encarregados de educação).
- Um dos encarregados de educação é doméstico, e os restantes têm perfis variados.

2.4.1 Níveis de formação académica

Não foi possível recolher os dados sobre todos os encarregados de educação uma vez que existe omissão de informação na base de dados fornecida pela diretora de turma.

- O Ensino Secundário é o nível mais comum, representado por 6 encarregados de educação.
- Seguem-se os níveis de formação do ensino Básico (2.º ciclo) e básico (3.º ciclo) representado por 2 encarregados de educação cada.

- Há ainda 2 encarregados de educação com Licenciatura, representando uma pequena fração com formação superior.

Os dados mostram uma turma composta por uma diversidade cultural com necessidades e apoios pedagógicos específicos para alguns alunos. Em termos de estratos sociais, a maioria dos encarregados de educação tem formação até ao nível secundário e está inserida no mercado de trabalho, enquanto apenas uma minoria possui formação superior. Estes fatores podem influenciar diretamente o acompanhamento escolar, o apoio e suporte fornecido aos alunos. A presença de alunos estrangeiros com necessidades educativas específicas reflete um desafio para a implementação de estratégias pedagógicas inclusivas e personalizadas.

2.5 Caracterização da sala de aula

A sala está configurada em forma de U, sendo que os computadores estão encostados à parede permitindo que o professor visualize sempre o que os alunos estão a trabalhar. A disposição das mesas permite visualizar o quadro assim como o projetor para que consigam acompanhar melhor as aulas expositivas tal como podemos ver na (**Figura 2**). Para Teixeira e Reis (2012), o espaço da sala de aula tem várias implicações no que diz respeito à aprendizagem e na interação entre professor e aluno. A configuração da sala de aula em U permite que os alunos se vejam uns aos outros estando o professor a ocupar um lugar de destaque permitindo assim uma melhor comunicação entre o professor e os seus alunos. Estando no centro da sala, o professor consegue ter uma liberdade de movimento que permite acesso rápido ao quadro, mas também acesso visual a todos os equipamentos informáticos facilitando uma maior supervisão sobre os seus alunos.

De acordo com o relatório *Global Education Monitoring Report Team* (UNESCO, 2023), a utilização dos telemóveis na sala de aula tem trazido preocupações significativas para a aprendizagem dos alunos. Alguns estudos indicam que a utilização destes aparelhos durante as aulas pode implicar distrações e prejudicar todo o trabalho realizado na sala de aula. A utilização dos telemóveis oferece várias ferramentas de aprendizagem, no entanto o uso do mesmo deve ser gerido de forma cuidadosa para evitar distrações e garantir que os alunos possam aprender.

Figura 2
Sala de Informática da Escola

3 UFCD 0784 – Programação em C/C++ - funções e estruturas

O principal objetivo da UFCD 0784 – Programação em C/C++ - Funções e Estruturas foi proporcionar aos alunos uma compreensão sólida dos conceitos fundamentais da programação modular, permitindo-lhes desenvolver capacidades a nível prático, mas também a nível teórico na criação e utilização de funções e estruturas na linguagem C/C++.

Esta unidade teve como objetivo a identificação das funções e estruturas presentes na linguagem C/C++, sendo que, durante a prática supervisionada, o foco incidirá exclusivamente sobre as funções. Pretende-se que os alunos compreendam a noção de subprogramas, reconhecendo que a sua utilização permite a aplicação dos princípios da programação estruturada. Além disso, espera-se que os estudantes desenvolvam a capacidade de reescrever códigos de forma mais simplificada, recorrendo à divisão de problemas em componentes menores como estratégia para a resolução de problemas mais complexos.

Ao longo das aulas foram explorados também os conceitos de programação modular, com ênfase nas funções e procedimentos, conforme indicado por Damas (2019, p. 157). De acordo com Sentance et al. (2023) também apresentam definições relacionadas às funções em C, que se alinham com os temas abordados neste relatório de estágio. Foram analisadas as principais características de uma função, incluindo a sua nomenclatura, o seu funcionamento, a identificação de argumentos, a estrutura do seu corpo, a utilização da instrução `return`, as funções que retornam valores, a função `void`, as diferenças entre variáveis locais e globais e, por fim, as bibliotecas pré-definidas pela linguagem C. Os alunos são incentivados a aplicar os conhecimentos adquiridos através de exercícios práticos e desenvolvimento de projetos para consolidar aprendizagens e melhorar a sua capacidade na resolução de problemas. Além disto os alunos ainda desenvolvem competências, capacitando-os a estruturar, modularizar e otimizar código, promovendo a reutilização e manutenção eficaz dos programas.

3.1 Plano de estudos da disciplina para o 1.º ano

Através da (**Tabela 3**) podemos analisar a estrutura da disciplina de Programação do curso profissional de Técnico de Informática-Sistemas distribuído em quatro módulos, 240 aulas e 200 horas.

Tabela 3
Distribuição dos módulos da disciplina de Conceitos Básicos de Programação

UFCD	Nome	N.º aulas	N.º de horas
0782	Programação em C/C++ - estrutura básica e conceitos fundamentais	60	50
0783	Programação em C/C++ - ciclos e decisões	60	50
0784	Programação em C/C++ - funções e estruturas	60	50
0785	Programação em C/C++ - formas complexas	60	50

3.2 Caracterização da Disciplina

A disciplina de Programação do 1.º ano, lecionada pela professora Margarida Batista estrutura-se em dois semestres, contabilizando um total de 240 aulas. A disciplina inclui diversas teorias e abordagens que procuram facilitar a aprendizagem destes alunos.

- Conteúdos a abordar:
  - Durante o primeiro semestre a disciplina foca-se na Programação em C/C++, que procura abordar a estrutura básica de um programa.
  - Tipos de dados, a função `printf()`, cadeias de caracteres e a entrada e saída de dados formatados, operadores e expressões em C/C++.
  - Diferentes tipos de estruturas.
  - Durante o segundo semestre os alunos aprenderão tópicos mais avançados, incluindo funções, estruturas, matrizes, apontadores, uniões e cadeias de caracteres (`string`).
- Estratégias para Aprendizagem:

- Aulas teóricas para aprendizagem dos conceitos e fundamentos da programação.
- Projeção de vídeos para exemplificação da utilização do software para facilitar a compreensão e a prática dos conceitos.
- Foco na interação entre o professor e o aluno de forma a promover um ambiente de aprendizagem colaborativo.

- Instrumentos de Avaliação:
  - Fichas de trabalho e trabalhos de projeto para permitir a avaliação do progresso dos alunos de forma contínua.
  - Observação direta em que o professor observa o desempenho dos alunos através da realização de atividades práticas.
  - Esforço consciente por parte do professor para dar apoio adicional aos alunos com dificuldades e que enfrentam desafios na aprendizagem.
- Projetos baseados em trabalho de grupo.

A disciplina também procura incentivar o trabalho em grupo, com o objetivo de promover a cooperação entre os alunos. As atividades realizadas em grupo permitem a discussão do tema do trabalho e estimulam a investigação, assim como o trabalho autónomo desenvolvendo também o pensamento crítico e o raciocínio.

3.3 Conceitos científicos a abordar durante a prática pedagógica

Os conceitos científicos abordados durante as aulas da prática supervisionada serão delineados e desenvolvidos a partir de Damas (2019, p. 157) no seu livro sobre *Linguagem em C* e também a partir de Sentance et al. (2023).

3.3.1 Funções e Procedimentos em C

De acordo com Damas (2019, p. 157), as funções e os procedimentos dividem-se nos seguintes tópicos essenciais:

- Conceito de Função.
- Características de uma Função.
- Nome de uma Função.
- Funcionamento de uma Função.
- Parâmetros.
- Corpo de uma Função.
- O valor `Return`.
- Funções que retornam valor.
- A Função `Void`.
- Variáveis.
    - Locais.
    - Globais.
- Bibliotecas.

3.3.2 Fundamentos da programação em C

De acordo com Damas (2019, p. 157), torna-se necessário aprender os vários tipos de funções existentes que fazem parte da biblioteca standard do C. É fundamental que qualquer programador em C seja capaz de escrever programas de forma modular através de funções e procedimentos. A decomposição é uma técnica de resolução de problemas que envolve dividir um problema complexo em partes menores, chamados subprogramas ou funções. Na linguagem C, a programação modular facilita esta decomposição, pois permite que o código seja organizado em módulos independentes. Cada módulo pode ser desenvolvido, testado e mantido separadamente, o que torna o processo de programação mais eficiente e organizado. No livro refere a importância entre a diferença entre função e procedimento. Afinal qual é a diferença entre função e procedimento? Uma função tem sempre um tipo e um valor de retorno, no entanto o procedimento não devolve qualquer valor. A forma como invocamos funções e

procedimentos é diferente uma vez que um procedimento não tem um valor de retorno. Relativamente às funções estas podem ser invocadas de diferentes formas.

- Exemplo de uma *função*: `int max(int n1, int n2)` – Esta função verifica o maior dos inteiros e devolve um.
- Exemplo de um *procedimento*: `linha ()` que coloca no ecrã uma linha e termina de seguida sem devolver qualquer valor.

De acordo com Sentance et al. (2023), existem conceitos relacionados às funções em C e que podem ser definidos da seguinte forma:

- **Características de uma função**: As funções são elementos fundamentais na programação, permitindo a reutilização de código e modularidade. Estas facilitam a organização e manutenção do programa.
- **Nome de uma função**: O nome de uma função deve ser significativo e descritivo para indicar o seu propósito. Nomes bem escolhidos aumentam a legibilidade do código e ajudam a identificar a função ao longo do programa.
- **Funcionamento de uma função**: A função executa operações especificas no seu corpo e, opcionalmente, pode retornar um valor. A execução ocorre quando a função é chamada noutro ponto do programa.
- **Parâmetros**: Os parâmetros permitem passar dados para dentro da função. Estes são utilizados como variáveis locais dentro da função, e o seu valor pode ser modificado apenas dentro dessa mesma função.
- **Corpo de uma função**: O corpo da função contém as instruções que definem o que a função faz. Normalmente, é delimitado por chavetas `{}` que representam o bloco de código que será executado quando a função é chamada.
- **O valor** `Return`*:* Muitas funções terminam com uma instrução `return`, que permite que a função envie de volta um valor ao código que a chamou. Esta instrução é especialmente útil para quando a função devolve um resultado.

- **Funções que retornam valor**: Algumas funções são criadas para executar a tarefa e devolver um valor, como por exemplo as funções matemáticas que calculam e retornam resultados específicos.
- **A função** `Void`*:* A função `void` não retorna qualquer valor. Este tipo de função é útil para funções que apenas executam ações (como exibir uma mensagem ou modificar variáveis globais) sem necessidade de devolver valores.
- **Variáveis locais e globais**: Variáveis locais são declaradas dentro de uma função e só podem ser utilizadas dentro desta, enquanto variáveis globais são mais fáceis de aceder a partir de qualquer ponto do programa.
- **Recursividade**: A recursividade ocorre quando uma função chama a si mesma e torna-se útil para resolver problemas que podem ser decompostos em subproblemas semelhantes.
- **Bibliotecas**: As bibliotecas oferecem funções pré-escritas que podem ser importadas para evitar a reescrita de código comum e aumentar a funcionalidade do programa.

3.4 Objetivos da Aprendizagem

No final deste módulo os alunos devem ter adquirido conhecimentos, procedimentos e atitudes que lhes permitam:

- Adquirir a noção de subprograma.
- Conhecer as regras de declaração de subprogramas.
- Conhecer as regras de execução de subprogramas.
- Utilizar corretamente parâmetros.
- Distinguir os diferentes tipos de subprogramas.
- Conhecer os mecanismos de utilização de bibliotecas de subprogramas.

3.4.1 Competências a desenvolver

- Efetuar a análise e desenvolvimento de sistemas de informação.
- Conceber programas através da divisão de problemas.
- Estimular o raciocínio lógico e o pensamento computacional.
- Saber escolher e adequar as soluções tecnológicas aos problemas a resolver.
- Estimular a reflexão, a observação e autonomia.

3.4.2 Conteúdos Programáticos

- Funções:
    - Estrutura e argumentos de uma função.
    - Variáveis locais.
    - Tipos de funções.

3.4.3 Análise crítica do currículo escolar – Escola, disciplina e turma de Intervenção

O currículo proposto para a intervenção está adequado na Unidade– Subprogramas em C e encontra-se de acordo com o Catálogo Nacional de Qualificações com o código (UFCD): 0784. A unidade de formação de curta duração (UFCD) – Subprogramas está alinhada com as necessidades do curso profissional de Técnico de Informática – Sistemas porque tem como principal objetivo o foco nos conceitos fundamentais da programação em linguagem C. O Agrupamento (AESA) caracteriza-se por ser composto por uma população diversa de jovens estudantes tanto em termos socioculturais como em termos socioeconómicos. A turma é composta por 14 alunos, predominantemente masculina - 12 rapazes e 2 raparigas, incluindo alguns alunos estrangeiros que necessitam de apoio adicional nas aprendizagens. Estes alunos frequentam um curso técnico-profissional de informática o que revela uma predisposição para a aprendizagem da programação. No entanto, a presença de alunos com necessidades educativas específicas exige a implementação de estratégias de ensino diferenciadas. A desigualdade no nível de conhecimentos entre os alunos torna-se um desafio para o professor que terá de garantir um ensino eficaz. Uma parte significativa destes alunos enfrenta

dificuldades uma vez que se sentem desmotivados, e cabe ao professor o papel de introduzir metodologias ativas e motivadoras.

4 Planificações das aulas supervisionadas

A planificação das aulas tem como objetivo principal estruturar e organizar a prática supervisionada, garantindo que os conteúdos são abordados de forma progressiva e adaptada ao nível de conhecimento dos alunos. Segundo Alves (2019), a planificação das aulas permite uma aprendizagem mais eficiente, gerindo as atividades planeadas a desenvolver, com o objetivo de aumentar competências em programação em C. Para o autor, a planificação define claramente os objetivos de cada aula e assegura uma abordagem sequencial e coerente dos temas abordados. Garante a progressão das aprendizagens dos alunos, uma vez que prevê atividades de introdução de novos conceitos e consolidação de conceitos adquiridos anteriormente.

Além disso, permite a preparação de recursos adequados para cada aula específica, selecionando os materiais e ferramentas que favoreçam a aprendizagem e possibilita a distribuição estratégica dos tempos letivos, uma vez que organiza a carga horária de forma equilibrada para otimizar o envolvimento e a participação dos alunos. Como destaca Alves (2019), a planificação estabelece um cronograma detalhado das aulas, assegurando que os alunos têm tempo suficiente para desenvolver e aplicar os conhecimentos adquiridos. Assim, a planificação das aulas, contribui para uma prática pedagógica mais eficaz, promovendo a autonomia, o pensamento crítico e a motivação dos alunos durante a aprendizagem da programação.

4.1 Planificação da Aula 1

Duração da aula: (50 minutos)

Introdução | Desenvolvimento

- Programação modular:
    - Conceito.
    - Vantagens.

- Ciclo de desenvolvimento de um programa em linguagem C.
- Estrutura de um programa em linguagem C.

Dinamização da aula:

- A aula é dinamizada com recurso a exemplos práticos e que reflitam situações reais.

4.2 Planificação das Aulas 2 e 3

Duração das aulas: (50 minutos + 50 minutos)

Introdução | Desenvolvimento

- Conceito de Função.
- Nomenclatura de uma Função.
- Estrutura de uma Função.
- A instrução `Return`.
- Função `Void`.
- Argumentos de uma Função.
- Resolução de exercícios conduzidos passo a passo: Definição e construção de funções e evocação das mesmas na função `main()`.

Dinamização da aula

- Através da utilização de exemplos práticos e situações reais.

4.3 Planificação das Aulas 4 e 5

Duração das aulas: (50 minutos + 50 minutos)

Introdução | Desenvolvimento

- Num 1.º momento os alunos irão explorar as funções com recurso à Inteligência Artificial. Irão criar uma calculadora com as quatro operações aritméticas com um menu de opções para o utilizador.
- Num 2.º momento a partir das soluções encontradas pelos alunos, o professor seleciona uma das soluções encontradas pelos alunos e irá desconstruir a solução obtida.

Dinamização da aula

- Utilização de exemplos práticos aliado a situações reais.

4.4 Planificação das Aulas 6 e 7

Duração das aulas: (50 minutos + 50 minutos)

Introdução | Desenvolvimento

- Prática computacional no ambiente Dev C++.
- Resolução de uma ficha formativa de aplicação de conceitos relativos a funções abordados nas aulas anteriores.
- Correção da ficha formativa.

Dinamização da aula

- Utilização de exemplos práticos e situações reais.

4.5 Planificação da Aula 8

Duração da aula: (50 minutos + 50 minutos)

Introdução | Desenvolvimento

- Resumo dos conceitos abordados nas aulas anteriores.

Aferição de aprendizagens:

- Aplicação de uma ficha com o objetivo de aferir aprendizagens ao longo das oito aulas.

4.6 Planificação de recursos a utilizar

- Quadro Branco.
- Retroprojetor.
- Computador.
- PowerPoint.
- Material de Escrita.
- Bloco de Notas ou Caderno do aluno.
- Fichas de exercícios (individuais ou em grupo).
- Manual técnico.
- Software de desenvolvimento Dev C++.
- Questionários para análise de perceções dos alunos.
- Ficha de aferição de aprendizagens.

4.7 Calendarização das aulas práticas supervisionadas

A calendarização das aulas práticas, teve como objetivo organizar e distribuir os tempos letivos de forma a garantir uma sequência pedagógica adequada e eficaz. O cronograma apresentado na (**Tabela 4**), estabeleceu as datas, horários e duração das aulas, permitindo um acompanhamento estruturado do progresso da turma. As aulas supervisionadas decorreram entre 4 e 11 de fevereiro de 2025, distribuídas ao longo de cinco dias. Todas as aulas estão programadas contemplando os momentos de abordagem teórica, essenciais para a introdução e reforço dos conceitos, assim como os momentos práticos, para que os alunos possam aplicar os conhecimentos adquiridos. De acordo com Alves (2019), a distribuição dos tempos letivos por dia e horário seguindo um plano estruturado, para garantir um equilíbrio entre as atividades propostas e o tempo a utilizar, permite a adoção de estratégias e métodos de ensino conforme

a evolução das aprendizagens dos alunos. A calendarização permite a observação contínua e o desempenho dos alunos, possibilitando ajustes pedagógicos sempre que necessário, de forma a garantir a progressão das aprendizagens dos alunos estabelecidos nos objetivos da disciplina.

Tabela 4

Cronograma da prática supervisionada

Aulas	**Dia da prática supervisionada**	**Hora de início e fim**	**Tempos**
Aula 1	4 de fevereiro de 2025	11:50 às 12:40	50 minutos
Aula 2 e 3	5 de fevereiro de 2025	8:55 às 10:50	50 minutos + 50 minutos
Aula 4 e 5	6 de fevereiro de 2025	11:50 às 13:35	50 minutos + 50 minutos
Aula 6 e 7	10 de fevereiro de 2025	10:00 às 11:45	50 minutos + 50 minutos
Aula 8	11 de fevereiro de 2025	11:50 às 12:40	50 minutos

5 Preparação das aulas supervisionadas

Neste capítulo, serão apresentados os objetivos, os recursos a utilizar e as estratégias a implementar durante as aulas práticas supervisionadas. A preparação é essencial para garantir e promover um ensino eficaz e adequado aos alunos da turma. É imperativo garantir a organização dos conteúdos e das metodologias de ensino, definir os objetivos de aprendizagem para assegurar que os alunos adquirem as competências fundamentais na programação em linguagem C, apresentar os recursos didáticos e tecnológicos (materiais físicos e digitais) que serão utilizados durante as aulas supervisionadas, descrever possíveis estratégias adotadas que incentivem a participação ativa dos alunos, que desenvolvam o pensamento crítico, a autonomia e a resolução de problemas e assegurando uma abordagem progressiva que respeite o ritmo da turma equilibrando sempre a transição entre conceitos teóricos e práticos. De acordo com Alves (2019), a progressão dos alunos e a melhoria do seu desempenho dependem da descrição clara dos objetivos de aprendizagem definidos e da escolha de metodologias e recursos adequados às características dos alunos.

5.1 Objetivos

No final desta unidade, os alunos devem ser capazes de:

- Entender o conceito de subprogramas.
- Identificar e aplicar corretamente as regras para declarar subprogramas.
- Conhecer e executar subprogramas de forma adequada.
- Utilizar argumentos de forma correta nos subprogramas.
- Diferenciar os diversos tipos de subprogramas existentes.

5.2 Estratégias a implementar

A disciplina de programação, combina teórica e prática, focando-se em atividades que promovam a resolução de problemas e codificação em linguagem C. Durante as aulas práticas

supervisionadas, será incentivada a participação ativa dos alunos, através do diálogo e resolução de exercícios práticos guiados em estilo "*step-by-step*". É imperativo iniciar as aulas transmitindo aos alunos os conceitos básicos de programação e lógica para criar bases sólidas, avançando gradualmente para os conteúdos mais complexos. A utilização de uma metodologia mais prática torna-se fundamental para aumentar a motivação dos alunos desta turma e a combinação com ferramentas de Inteligência Artificial pode enriquecer a aprendizagem. Manter um *feedback* contínuo será fundamental na medida em que permite apoiar os alunos a superar as suas dificuldades. De acordo com o autor Alves (2019), existem estratégias consideradas eficazes para o ensino da programação tais como: A Aprendizagem Baseada em Problemas (ABP), a resolução de exercícios no computador, a utilização de plataformas digitais de aprendizagem, a interação e o *feedback* constante.

Estratégias fundamentais a implementar durante a intervenção das aulas práticas supervisionadas após leitura e consulta do trabalho publicado por Alves (2019):

- Aplicação de conceitos teórico-práticos para consolidação de conhecimento dos alunos.
- Resolução de exercícios práticos conduzidos durante as aulas por forma a combinar a aplicação de conceitos teóricos com a prática.
- Foco na interação e no *feedback* contínuo durante as aulas mantendo um diálogo entre o professor e os alunos.

A disciplina em que a unidade de formação se insere tem um carácter predominantemente teórico-prático e experimental, pelo que durante esta unidade serão implementadas metodologias práticas através de atividades que incidam na resolução de fichas formativas com o objetivo de preparar e consolidar conhecimentos dos alunos ao longo das aulas, promover o trabalho entre pares e fortalecer a aprendizagem autónoma. Durante a resolução das mesmas o professor deve interagir com os alunos e prestar apoio aos que revelam maiores dificuldades personalizando o ensino através de um apoio individualizado tal como é referido por Alves (2019).

6 Reflexão sobre as aulas práticas supervisionadas

De acordo com Lopes e Bastos (2017), a Prática de Ensino Supervisionada (PES), é um processo não só de aplicação de conhecimentos teóricos, mas também um processo de aprendizagem contínua. O professor estagiário, deve refletir sobre o que funcionou, e sobre o que pode ser melhorado. Os registos feitos durante as aulas, permitem que o professor reconheça a necessidade de melhorar as suas práticas pedagógicas. Na sequência da análise daquilo que foi feito o professor pode proceder a vários ajustes e adaptações conforme vai identificando as necessidades dos alunos. As adaptações podem concretizar-se ao nível de conteúdos e atividades respeitando o ritmo de trabalho dos alunos da turma. Não só os alunos aprendem como o próprio professor estagiário aprende durante este processo, facto que lhe permite adquirir conhecimentos e melhorar a sua própria capacidade de adaptação. Assim sendo a PES é uma fase essencial na formação do professor ao qual é permitido refletir, experienciar, corrigir e assim progredir. O artigo ainda reforça o facto de que o processo do registo das aulas permite ao professor avaliar as metodologias que utiliza e também avaliar os impactos sobre o sucesso da aprendizagem dos seus alunos. A descrição das aulas permite o registo e análise das experiências vividas durante o processo da prática do ensino supervisionado.

6.1 Aula 1

No dia 4 de fevereiro de 2025 realizou-se a primeira aula prática supervisionada. O professor estagiário pôde refletir sobre diversos aspetos tanto de carácter pedagógico assim como sobre as estratégias que impactam diretamente nas aprendizagens dos alunos. Ao longo da aula foi possível identificar alguns pontos cruciais que podem e devem ser melhorados para tornar o ensino das aprendizagens mais eficaz e envolvente para os alunos. Foi utilizado o método dialogante, interrogatório e o método afirmativo demonstrativo. Durante as aulas foi sempre mantido um diálogo com os alunos, colocando questões e fazendo a demonstração através de exemplos práticos sempre que necessário, com o objetivo de que os alunos

compreendessem de forma correta os conceitos aplicados. A gestão de tempo de aula revelou-se um fator crítico uma vez que houve algum desequilíbrio entre o tempo usado na exposição teórica e na aplicação prática. A exposição teórica terminou antes do previsto, pelo que houve a necessidade de propor aos alunos um exercício prático. Contudo, esse facto permitiu aos alunos antecipar a atividade prática. O exercício tinha como objetivo a criação de uma função simples. Outro ponto essencial foi a utilização do reforço positivo, uma vez que o incentivo aos alunos e o reconhecimento dos seus progressos contribuiu para a motivação dos alunos. Expressões tais como: "É assim mesmo", ou "Boa resposta", funcionam como motivadores durante o processo de aprendizagem. Skinner, como citado por Brito (2024), destaca a importância destes reforçadores para o contexto educacional.

O professor estagiário sentiu a necessidade de explorar melhor as respostas que os alunos deram em sala de aula. Em vez de apenas validar ou corrigir, foi fomentada a reflexão entre os alunos, incentivando-os a justificar as suas respostas e a aprofundar o seu raciocínio. Esta estratégia parece estimular o pensamento crítico e o envolvimento ativo dos alunos na construção do seu processo de aprendizagem. Um dos temas abordados foi a diferença entre variáveis globais e locais. Foi possível observar que alguns alunos tiveram dificuldades em compreender este conceito e as suas diferenças, o que reforçou a necessidade de na aula seguinte, apresentar aos alunos exemplos mais claros e contextualizados baseados em situações reais.

Relativamente à postura do professor, concluiu-se que é importante adotar uma postura interativa com os alunos para tornar a aula mais acessível e estimulante para os alunos. Foram transmitidos conhecimentos de forma direta, mas os alunos também foram conduzidos à descoberta dos conceitos e definições, pelo incentivo à participação na formulação de novas hipóteses. Foi mantido um ritmo mais equilibrado e dinâmico durante as aulas de forma a evitar que os alunos perdessem o foco, contribuindo para um ambiente de aprendizagem mais produtivo. Durante a aula houve alguns períodos chamados "tempos mortos": o registo escrito do sumário e o registo escrito no caderno de alguns conceitos importantes sobre a matéria. Durante a aula houve sempre o cuidado de adequar o nível de exposição e de prática ao nível etário dos alunos da turma.

Refletir sobre estes aspetos permitiu reforçar que ensinar não se resume apenas à transmissão de conteúdos, mas também à forma como o professor interage com os alunos, adaptando sempre as suas metodologias e estratégias, promovendo um espaço de aprendizagem ativo e envolvente.

6.2 Aulas 2 e 3

No dia 5 de fevereiro de 2025 antes do início da aula realizou-se uma súmula da aula anterior para criar uma ponte com os conceitos transmitidos anteriormente de forma a ajudar os alunos a assimilar conceitos, mas também a esclarecer algumas dúvidas que os alunos ainda tivessem. Durante a aula prática supervisionada foi dada continuidade ao trabalho desenvolvido na aula anterior, dando prioridade à resolução do exercício anterior e posteriormente à sua correção. É importante referir que foi iniciada a matéria referente às aulas 2 e 3, e transmitido o sumário da aula aos alunos que o registaram nos seus cadernos. Ao invés de continuar a ser apresentada a teoria apenas através da apresentação digital, a aula acabou por ser baseada nos conteúdos expostos no sumário, mantendo um discurso fluido entre o professor e os alunos sem a utilização persistente do método expositivo. Foi mantido sempre um diálogo baseado na interação entre o professor e os alunos.

A opção por uma condução de aula diferente, reduzindo a dependência extrema do método expositivo e privilegiando apenas a comunicação oral e a interação direta com os alunos, teve como objetivo o aumento da concentração e do foco dos alunos. Durante a aula, foi utilizada uma apresentação digital. Foram selecionados apenas os tópicos de maior interesse e ao longo da aula, foram aplicados os métodos interrogativo e afirmativo, conduzindo a uma explicação dos conceitos de forma mais dinâmica. É importante destacar que o apoio através de apresentações digitais tem como objetivo fornecer aos alunos um guia de estudo. Desde o início, procurou-se captar a atenção dos alunos através do diálogo, recorrendo a perguntas estratégicas para estimular o pensamento crítico e promover a participação ativa. Com a redução da dependência extrema das apresentações digitais, os conteúdos foram explorados e transmitidos de forma clara e envolvente, ajustando-se o ritmo e a abordagem de acordo com as reações e necessidades dos alunos. Esta metodologia de ensino favoreceu um ambiente mais interativo, o que incentivou os alunos a prestar mais atenção à explicação oral, a refletir sobre

os conteúdos e a tomar apontamentos de forma autónoma, permitindo-lhes definir, por si próprios, os conceitos abordados em aula. Deve-se também referir que, devido à sua faixa etária e à pouca experiência nos conteúdos abordados, os alunos demonstraram dificuldades na compreensão de conceitos relacionados com a programação modular, subprogramas, entre outros.

Sem o apoio da apresentação digital, foi proporcionada aos alunos uma experiência diferente, na qual a compreensão dependia exclusivamente da escuta ativa e da construção coletiva do conhecimento. Esta abordagem estimulou a capacidade de concentração dos alunos e o debate espontâneo entre o professor e os estudantes. Para complementar e reforçar a aprendizagem, foram propostos aos alunos exercícios de interpretação sobre a identificação de funções com e sem retorno. Através do diálogo e do *feedback* com os alunos, proporcionou-se uma experiência diferenciada de aprendizagem, permitindo uma reflexão conjunta sobre o conhecimento transmitido ao longo da aula. A estratégia foi dirigida às capacidades do pensamento crítico dos alunos, bem como ao espírito de entreajuda entre os elementos da turma.

Relativamente aos conceitos abordados, foram discutidos o tipo de variáveis a serem declaradas, uma vez que podem existir variáveis globais e variáveis locais quando se trabalha com subprogramas em linguagem C. Quando são declaradas funções, é importante que os alunos sejam capazes de identificar que tipo de variáveis devem ser declaradas. A compreensão destes conceitos é fundamental na declaração e utilização de funções, uma vez que permite que os alunos sejam capazes declarar variáveis, bem como compreender melhor o conceito de programação modular.

Relativamente à postura do professor em sala de aula, notou-se uma melhoria significativa na forma como interagiu com os alunos, uma vez que se mostrou mais descontraído, o que permitiu uma maior interação com os alunos durante a prática letiva.

A experiência revelou-se enriquecedora tanto para o professor quanto para os alunos. Ao optar por uma abordagem menos centrada no apoio visual, foi reforçada a importância da comunicação e interação direta com os alunos. Este tipo de atividade pedagógica foi mantido e aprimorado nas aulas.

6.3 Aulas 4 e 5

Durante a aula no dia 6 de fevereiro de 2025, verificou-se a necessidade de reforçar os conteúdos relativamente aos diferentes tipos de variáveis, através de exemplos práticos de identificação de variáveis. Após a revisão, deu-se início às aulas 4 e 5. Durante estas aulas, os alunos resolveram exercícios práticos utilizando o ambiente de programação Dev C++, de forma a consolidarem os seus conhecimentos na programação da linguagem C. Além disso, foi proposto aos alunos que explorassem a utilização das ferramentas de Inteligência Artificial como forma de auxílio durante a aprendizagem da linguagem de programação em C. Através da observação direta, foi possível constatar a evolução significativa dos alunos na forma como interagem com os exercícios, demonstrando maior autonomia, interesse e motivação durante a aprendizagem da programação. A prática regular e o apoio da Inteligência Artificial têm contribuído para um maior envolvimento e autoconfiança na resolução de problemas desde que estes alunos iniciaram as aulas com a professora cooperante da escola.

Outro aspeto positivo foi a crescente naturalidade e à vontade dos alunos com a presença do professor na sala de aula, sendo que as regras estabelecidas foram cumpridas de forma criteriosa. Este fator pode indicar que os alunos estão a sentir-se mais confortáveis para esclarecer dúvidas, partilhar dificuldades e participar ativamente no processo de aprendizagem. De um modo geral, a aula decorreu de forma produtiva, reforçando não só a aprendizagem da linguagem C, mas também a confiança e o envolvimento dos alunos na disciplina. Durante a resolução dos exercícios práticos, o professor foi esclarecendo as dúvidas dos alunos e prestando apoio de forma individualizada. Houve ainda uma melhoria da gestão dos tempos, uma vez que o plano de aula programado foi cumprido com sucesso.

6.4 Aulas 6 e 7

Durante a aula de 10 de fevereiro de 2025, foram aprofundados conceitos e funções em linguagem C, com foco na programação modular e nos princípios da estrutura de um programa em C. A ficha de trabalho desafiava os alunos a refletir sobre a diferença entre funções com e sem retorno, bem como a aplicar na prática estes conceitos, recorrendo ao ambiente de programação Dev C++. No âmbito dos exercícios de codificação, os alunos desenvolveram

programas que permitem ao utilizador optar entre uma função sem retorno para verificar se um número introduzido é par ou ímpar e uma função com retorno para calcular a idade média entre dois alunos. Foi atribuída uma tarefa aos alunos para a realização de uma ficha de trabalho a pares, com o objetivo de consolidar conhecimentos sobre subprogramas em C, compreender o conceito de programação modular e as diferenças entre os diferentes tipos de funções em C. Pretende-se, com este exercício, incentivar a aprendizagem autónoma dos alunos, desafiando-os a explorar e aplicar os conceitos por si próprios, recorrendo à análise de problemas e à utilização do ambiente de programação Dev C++. Esta forma de aprendizagem desenvolve maior confiança na resolução de problemas e fortalece a capacidade de tomar decisões.

É importante referir que os alunos também foram incentivados a utilizar as ferramentas de Inteligência Artificial, não apenas como um apoio na resolução das suas dúvidas, mas também como um recurso que permite explorar diferentes abordagens no ensino da programação. A Inteligência Artificial permitiu-lhes identificar erros no código, obter explicações mais detalhadas e complementares e otimizar soluções, promovendo assim uma aprendizagem mais autónoma, personalizada e eficiente. Durante toda a aula, o professor acompanhou os alunos na realização da ficha de trabalho, oferecendo apoio individualizado durante a interpretação das questões, bem como na validação das respostas corretas, esclarecendo sempre as dúvidas apresentadas pelos alunos. Foi fornecido *feedback* sobre as soluções apresentadas, reforçando a importância da indentação (boas práticas e regras) essencial para tornar o código mais legível e organizado, assim como a escrita estruturada do próprio código cumprindo as boas práticas da programação. O professor reforçou de forma positiva sempre que os alunos realizavam o exercício corretamente. Para finalizar, foi recordada aos alunos a importância de submeter o trabalho na plataforma *Microsoft Teams*, cumprindo as normas estabelecidas para a entrega.

6.5 Aula 8

No dia 11 de fevereiro de 2025, na última aula foi aplicada uma ficha de aferição de aprendizagens, elaborada em colaboração com a professora cooperante, com o objetivo de

aferir os conhecimentos adquiridos ao longo das oito aulas práticas dedicadas à UFCD 0784 – Programação em C/C++ - Funções. A ficha de aferição foi preparada para uma duração de 50 minutos e estruturada em dois grandes grupos de questões, abordando tanto a compreensão teórica dos conceitos fundamentais da programação modular como a capacidade de interpretação e análise de código em linguagem C.

No grupo 1, os alunos responderam a questões de escolha múltipla, focadas na definição de subprogramas, vantagens e desvantagens da programação modular, diferenças entre funções com e sem retorno, variáveis locais, tipos de dados primitivos, estrutura de um programa em C, o ciclo de desenvolvimento de um programa, entre outros conceitos essenciais abordados ao longo das aulas práticas supervisionadas. No grupo 2, os alunos tiveram de analisar diferentes trechos de código em linguagem C, de modo a demonstrar a sua capacidade de identificar e interpretar diferentes tipos de funções, compreender a lógica do retorno de valores e justificar a utilização dos diferentes tipos de variáveis. Esta ficha de aferição permitiu medir o progresso individual dos alunos, mas também possibilitou a adequação de estratégias de ensino durante as aulas supervisionadas. Durante a sua realização, foi possível observar que alguns alunos demonstraram maior facilidade na aplicação dos conceitos teóricos, enquanto outros enfrentaram desafios na interpretação e implementação do código. Com este tipo de exercício pretendeu-se desenvolver o pensamento crítico e a capacidade de resolução de problemas dos alunos, competências essenciais para a aprendizagem da programação. A correção da ficha de aferição será realizada numa aula posterior, proporcionando uma oportunidade para refletir sobre os erros cometidos e reforçar os aspetos que necessitam de melhoria.

7 Um estudo sobre metodologias e estratégias, o papel da motivação e a integração da IA na aprendizagem da programação

7.1 Revisão da Literatura

A revisão da literatura teve um papel fundamental na justificação teórica deste estudo de investigação, permitindo compreender os principais conceitos, abordagens e estudos realizados sobre o ensino da programação em linguagem C. Através da análise de diferentes autores, é possível identificar metodologias, desafios e estratégias que contribuam para um ensino de qualidade. Neste sentido, esta revisão de literatura contribuiu para a exploração das dificuldades enfrentadas pelos alunos durante a aprendizagem da programação, bem como as metodologias e estratégias mais eficazes para promover a aprendizagem dos alunos desta turma. Compreender o impacto das ferramentas do Dev C++ aliado com as ferramentas de Inteligência Artificial durante o processo de ensino-aprendizagem, permitiu verificar efeitos nas aprendizagens dos alunos.

7.2 Identificação dos Problemas de Investigação

Creswell e Clark (2014), no livro *Research Design: Qualitative, Quantitative, and Mixed Methods Approaches,* discutem a importância da definição das questões de investigação. A definição do problema de investigação é um processo complicado, mas fundamental e essencial que exige o estabelecimento de prioridades iniciais de forma a direcionar o estudo aos objetivos da pesquisa. Este estudo científico proporciona a base para uma recolha de dados estruturada e uma análise criteriosa dos dados recolhidos. O relatório de investigação identifica duas questões que procuram identificar métodos e ferramentas para o ensino da programação com o objetivo de avaliar a eficácia e simplicidade do *software* Dev C++, a integração da Inteligência Artificial na aprendizagem da programação e o impacto da metodologia baseada em projetos como facilitadora e motivadora no processo de aprendizagem do aluno. Esta investigação pretendeu dar resposta às questões de investigação através de uma revisão bibliográfica, estudo

empírico e validação prática por meio da utilização nas aulas práticas supervisionadas. Esta abordagem permitiu um maior entendimento sobre o problema de investigação ao integrar as teorias e práticas de forma a enriquecer a formação dos futuros alunos do mestrado via ensino da informática e professores que estão a lecionar programação ao nível do ensino secundário.

Foram criadas e identificadas as seguintes questões de investigação que dão relevância a este estudo de investigação.

- De que modo o perfil e a motivação dos alunos influenciam a aprendizagem da programação em linguagem C?
- Poderão as ferramentas do Dev C++ e a Inteligência Artificial tornar mais fácil o processo de aprendizagem da programação em linguagem C?

7.3 O papel da motivação nas aprendizagens dos alunos

A motivação desempenha um papel crucial na aprendizagem deste tipo de disciplinas mais técnicas e práticas como o ensino da programação. De acordo com Ryan e Deci (2000), a motivação intrínseca, associada à curiosidade e interesse pessoal, é um fator determinante para o sucesso dos alunos. Por outro lado, a motivação extrínseca, ligada a recompensas externas, pode ser eficaz em contextos estruturados, como por exemplo nos cursos técnico-profissionais.

O artigo de Filomena Ribeiro (2011) destaca a importância da motivação para o sucesso no ensino-aprendizagem dos alunos. A autora salienta o papel da motivação dos estudantes nas disciplinas curriculares, uma vez que os fatores individuais e o contexto educativo influenciam diretamente o envolvimento dos alunos. Ao longo do artigo, diversas teorias da motivação são abordadas e diversos estudos são apresentados para demonstrar que a motivação pode prever e orientar o comportamento escolar. Distingue entre motivação extrínseca e motivação intrínseca. A motivação extrínseca, baseia-se numa recompensa e em punições externas, já a motivação intrínseca, decorre do interesse pessoal pela aprendizagem. Os alunos com maior motivação intrínseca, tendem a aprofundar os seus conhecimentos e a obter melhores resultados. Em contexto de sala de aula, o professor desempenha um papel fundamental na criação de um ambiente estimulante, utilizando estratégias que incentivam a curiosidade, a autonomia e a autoestima dos alunos. Outro ponto focado no artigo é a diferenciação dos

diversos estilos motivacionais dos alunos, na medida em que existem perfis distintos entre os alunos. Existem perfis que apenas procuram o sucesso, perfis que apenas são curiosos, perfis que são conscienciosos e ainda existem os socialmente motivados. É importante que os professores sejam capazes de reconhecer estas diferenças para que assim possam diversificar os seus métodos pedagógicos para atender às necessidades individuais dos alunos. Conclui-se que o ensino integra tanto aspetos cognitivos quanto afetivos, as aprendizagens são mais significativas, no entanto a motivação não é apenas um fator interno ao aluno, mas também é um elemento influenciado pelo professor e pelo ambiente escolar.

Brito (2024) aborda a motivação à luz das teorias da psicologia da aprendizagem para o ensino da programação e da robótica. Este trabalho foi desenvolvido, com o objetivo de destacar a importância da motivação para a aprendizagem e como as diversas teorias psicológicas podem ser aplicadas no ensino. Neste trabalho foram desenvolvidas algumas estratégias e metodologias para o ensino da programação. A motivação será abordada através das diversas teorias, incluindo as de Bruner e de Skinner, (como citados por Brito, 2024), uma vez que estas duas teorias são importantes para o ensino da programação. Segundo Brito (2024), Bruner desenvolveu a teoria da aprendizagem por descoberta e defendia que a motivação surge quando os alunos exploram e descobrem por si próprios. Este tipo de motivação desenvolve a aprendizagem ativa nos alunos. Ainda segundo Bruner (como citado por Brito, 2024), o professor apenas atua como um condutor, que permite que os alunos explorem e desenvolvam a sua curiosidade e autonomia. Este tipo de ensino por descoberta, aumenta os níveis de motivação na medida em que permite um maior envolvimento dos alunos na compreensão da lógica da programação através da prática.

Para Skinner (como citado por Brito, 2024), a teoria do condicionamento operante e a recompensa durante a aprendizagem baseiam-se nos reforços positivos e negativos na medida em que moldam o comportamento e motivam a aprendizagem. Para o ensino da programação, os alunos devem ser incentivados cada vez que obtiverem resultados, de forma a promover a motivação intrínseca e extrínseca. Através da motivação intrínseca os alunos obtêm o prazer em aprender e através da motivação extrínseca a aprendizagem ocorre através de elogios e reconhecimento. A aprendizagem baseada na experimentação e na tentativa e erro são fundamentais para que os alunos consigam sucesso na realização das suas tarefas. O trabalho

também referencia a importância do ambiente de sala de aula, para a motivação dos alunos. Os professores devem aplicar estratégias através de *feedback* positivos, desafios progressivos e a criação de ambientes de respeito e incentivo. Os ambientes motivadores, melhoram a participação ativa dos alunos e tornam a aprendizagem mais eficaz. Este trabalho demonstra que a motivação para a aprendizagem da programação exige dos alunos, curiosidade, autonomia, persistência e interação social. Promover a descoberta, a experimentação e o reforço positivo pode resultar em aprendizagens mais significativas e duradouras.

7.4 Estratégias e metodologias para o ensino na programação

Para Alves (2019) é dada prioridade a métodos como a aprendizagem baseada em problemas e cenários reais, os quais permitem desenvolver competências como o pensamento computacional, a resolução de problemas e a autonomia dos alunos. Estas metodologias permitem contextualizar os conteúdos aproximando-os mais da realidade com que se depararão, posteriormente a nível profissional, incentivando a comunicação e a gestão de conflitos em ambiente colaborativo segundo CAST (2011). De acordo com Alves (2019), existem estratégias pedagógicas mais adequadas para melhorar o ensino e a aprendizagem da programação em C. O autor recomenda a utilização de diversas estratégias. O Ensino baseado numa programação baseada em projetos permite que os alunos aprendam os conceitos através de problemas reais promovendo assim uma aprendizagem mais ativa. A resolução de exercícios no computador é outra estratégia recomendada pelo autor uma vez que combinam a aplicação de conceitos teóricos com a prática, permitindo assim que os alunos utilizem editores e compiladores para a resolução dos exercícios práticos uma vez que estas ferramentas permitem orientar os alunos na aprendizagem da sintaxe e facilitam a aprendizagem prática aliado à aprendizagem dos conceitos mais teóricos. A utilização de uma plataforma de aprendizagem digital permite um suporte contínuo na medida em que disponibiliza materiais e também permite monitorizar o progresso do aluno assim como a realização de diversos tipos de avaliações. Para o autor é extremamente importante o foco na interação e *feedback* contínuo durante as aulas, assim como estimular os debates e questões com o foco num espírito mais crítico e participação ativa dos estudantes. Relativamente à avaliação o autor defende a aplicação de estratégias baseadas numa avaliação contínua ao longo de todo o semestre incentivando assim os alunos a manterem um ritmo constante durante a aprendizagem. Este

tipo de estratégias torna o processo de aprendizagem mais dinâmico, prático e adaptado às necessidades dos estudantes promovendo assim um aumento da motivação, interesse e eficácia no ensino da programação.

Tendo em conta o tipo de disciplina e o programa a lecionar é fundamental nesta disciplina que o professor privilegie sempre aulas mais práticas com o objetivo de desenvolver nos seus alunos o pensamento crítico, o pensamento computacional, o raciocínio, a resolução de problemas e a pesquisa de forma autónoma. Este tipo de ensino baseia-se numa metodologia de aprendizagem baseada em projetos. De acordo com Terra (2022), este tipo de metodologia é construtivista porque envolve os estudantes na pesquisa e resolução de problemas. Este tipo de estratégia de ensino incentiva os alunos a serem responsáveis por planear e organizar as suas tarefas, assumindo um papel ativo nas suas decisões. Torna-se importante ainda referir que este tipo de metodologia incentiva o trabalho em grupo permitindo o desenvolvimento de competências a nível da comunicação e gestão de conflitos. Além destas valências, este tipo de metodologias permite o ensino das teorias aliada à prática.

De acordo com CAST (2011), o Desenho Universal de Aprendizagem é baseado em três princípios: Representação, Ação e Expressão, e o Envolvimento:

- Representação: Este princípio assenta na utilização de diversas ferramentas, tais como vídeo, áudio e texto. Permite também o apoio e a compreensão através de ilustrações, gráficos e ferramentas visuais.
- Ação e Expressão: Este princípio assenta na forma de expressar e comunicar através de ferramentas e tecnologias que ajudem na composição e construção das aulas.
- Envolvimento: Este princípio motiva e envolve os alunos através de diversas escolhas e autonomia durante o processo de aprendizagem.

Estes princípios têm como objetivo fornecer um ambiente de aprendizagem que tenha em conta todas as necessidades dos alunos de forma a promover a inclusão. O Desenho Universal de Aprendizagem constitui-se como uma estratégia de ensino que poderá ser utilizada por professores que adotem metodologias de projeto baseadas em cenários de aprendizagem.

No artigo de Mangas e Sousa (2020), sobre práticas de ensino-aprendizagem, os "cenários de aprendizagem" são utilizados para criar experiências educativas que refletem situações do mundo real. Os alunos trabalham em contexto de inclusão, divididos em pequenos grupos, analisam relatórios técnicos adaptados sobre pessoas com deficiência. A atividade estimula a capacidade dos estudantes de propor soluções para necessidades reais em áreas como a educação. Esta abordagem promove a aprendizagem prática, colaborativa e permite a inclusão dos alunos. Serão aplicados dois questionários aos alunos da turma a lecionar para caracterização da turma, avaliação dos conhecimentos e competências dos alunos na aprendizagem da linguagem de programação em C.

Bell (2010) aborda o papel da metodologia de projeto que se baseia na aprendizagem através de cenários de aprendizagem. A aplicação prática dos conceitos promove a motivação e o envolvimento ativo dos alunos durante o processo de aprendizagem.

Segundo Barron e Darling-Hammond (2008), a metodologia de projeto ou *Project-Based- learning* (PBL) é eficaz para as disciplinas em que o ensino envolve a prática e o desenvolvimento de competências técnicas como o caso do ensino da programação. Desenvolve ainda capacidades ao nível da resolução de problemas e trabalho em equipa, competências fundamentais para a formação de alunos através de ambientes de aprendizagem aplicada e colaborativa.

7.5 A influência do método Interrogativo e dialogante nas estratégias de ensino

Segundo a **Universidade do Porto (s.d.),** o método interrogativo é uma metodologia que assenta numa interação verbal entre o professor e os alunos, recorrendo a questões que ajudam a estimular a reflexão, a análise e a compreensão dos conteúdos aprendidos. Esta metodologia, foi criada tendo as suas bases no filósofo grego Sócrates (século IV AC), que incentivava os alunos a desenvolverem capacidades ao nível do pensamento crítico através do questionamento, mas sempre orientado pelo professor.

Os princípios fundamentais deste método podem ser caraterizados do modo seguinte:

O método do questionamento, na qual o professor faz recurso a perguntas abertas que valorizam o conhecimento prévio e as experiências dos estudantes, participação ativa dos alunos na medida em que estimula o envolvimento através de questões desafiadoras e pertinentes, desenvolvimento do pensamento crítico, uma vez que promove a capacidade dos estudantes para identificar informações relevantes, avaliar argumentos e formar opiniões fundamentais. Este método também desenvolve a aprendizagem centrada no estudante, uma vez que o aluno torna-se o centro dele próprio e vai construindo o seu conhecimento, através da exploração de novas ideias assim como estabelece ligações entre os diversos conceitos.

Vantagens da utilização do Método Interrogativo:

Estimula a reflexão e o pensamento crítico através de questões que desafiam os alunos a refletir sobre os diversos temas abordados, aumenta a motivação e o envolvimento, porque a utilização de questões, despertam interesse e incentivam a participação ativa dos alunos. Também fornecem *feedback* imediato por parte do professor porque permite ajustar o ensino conforme necessidade. Melhora a compreensão e a aprendizagem através de questões permitindo aos alunos assimilar mais facilmente os conteúdos e promove a aprendizagem colaborativa, proporcionando discussões e debates com os alunos permitindo assim aprenderem através uns dos outros.

Podemos afirmar que o método interrogativo é eficaz porque coloca o aluno como o centro do processo de aprendizagem, utilizando o método questionador para aumentar a aprendizagem ativa e significativa. Coelho et al. (2024), destacam a importância do diálogo na formação de professores a partir da pedagogia de Paulo Freire. A educação deve ser inclusiva, crítica e socialmente transformadora e o diálogo é uma ferramenta importante para este processo. Os professores devem utilizar a abordagem dialogadora na construção do conhecimento porque ajuda os alunos a desenvolverem uma consciência crítica sobre a realidade. O artigo utiliza como metodologia uma revisão sistemática da literatura, analisando artigos, livros e pesquisas académicas sobre o tema do diálogo na formação de professores a partir da perspetiva de Paulo Freire. Coelho et al. (2024) destacam-se pela crítica à "educação

bancária", um modelo tradicional em que o professor transmite conhecimento de forma unidirecional, sem interação com os alunos. No entanto, os autores destacam Paulo Freire para descrever a pedagogia dialogante, baseada no diálogo, na comunicação e no respeito mútuo, incentivando uma aprendizagem mais ativa e significativa. O diálogo, não deve ser visto apenas como uma estratégia didática, mas sim como um método essencial para o ensino. Ao promover o pensamento crítico e a autonomia dos estudantes, o professor assume o papel de facilitador, incentivando a reflexão e a participação ativa dos alunos no processo educativo. Na formação de professores, o diálogo desempenha um papel fundamental, pois permite valorizar as experiências dos alunos, desenvolver competências reflexivas e críticas, mas também preparar os professores para atuarem em ambientes colaborativos. Além disso, fortalece a relação entre professores e alunos, promovendo um ambiente de ensino baseado no respeito e na empatia.

A implementação da pedagogia dialogante enfrenta diversos desafios. Entre os principais obstáculos estão a resistência institucional, uma vez que muitas escolas ainda seguem métodos tradicionais de ensino, e os professores não recebem formação adequada. A falta de recursos também representa um problema, pois a implementação eficaz desta metodologia exige materiais didáticos apropriados e tempo suficiente para a sua preparação. Apesar destes desafios, a abordagem de Paulo Freire continua a ser uma referência para a construção de uma educação mais inclusiva, crítica e transformadora, onde o diálogo é a base para o desenvolvimento do conhecimento.

7.6 O papel do Dev C++ na aprendizagem da programação

Ainda no contexto das estratégias utilizadas no ensino da programação, Robins et al. (2003) desenvolvem um estudo sobre o impacto dos ambientes de desenvolvimento na formação de programadores principiantes. A utilização destes ambientes de programação proporciona aos alunos que estão a aprender programação um suporte valioso no desenvolvimento de competências essenciais permitindo um envolvimento maior na aprendizagem dos conceitos da linguagem de programação em C uma vez que permite escrever e estruturar o código de forma prática e interativa. Para os autores, a utilização destes ambientes de programação beneficia os alunos uma vez que oferecem uma *interface* que facilita a execução de programas em tempo real, proporcionando um *feedback* imediato dado que

permite a visualização imediata dos resultados através de uma janela chamada consola. Esta característica é essencial para o processo de aprendizagem que permite que os alunos identifiquem e corrijam os erros de forma rápida, desenvolvendo uma compreensão prática e profunda sobre os conceitos de programação. Lidar desde cedo com os ambientes de programação melhora a confiança dos estudantes e ajuda-os a lidar com problemas de lógica e a estruturar o código de forma mais eficiente.

O ensino da programação do C através do *software* Dev C++ é ideal porque apresenta uma *interface* simplificada e acessível, reduzindo a complexidade técnica que outros ambientes de programação (IDEs) apresentam. Como refere Damas (2011), o Dev C++ é uma ferramenta adequada para alunos principiantes, pelo seu carácter gratuito, simples e eficaz no ensino da programação em C. O recurso ao *software* Dev C++ é crucial para alunos que estejam a iniciar a aprendizagem da programação que podem assim centrar-se nos aspetos fundamentais da linguagem de programação em C e nos conceitos de desenvolvimento de *software* sem serem assoberbados por configurações mais avançadas que outros ambientes de programação apresentam. O Dev C++ apresenta um ambiente de programação simples e gratuito de código aberto. Este tipo de estratégia facilita a compreensão dos conceitos teóricos, mas também prepara os estudantes para o desenvolvimento de soluções práticas promovendo experiências de aprendizagem mais completas.

7.7 O papel da Inteligência Artificial no ensino da programação

A Inteligência Artificial tem desempenhado um papel crucial na transformação da educação, especialmente no ensino de programação. O estudo realizado por Yilmaz e Yilmaz (2023), explora as várias formas como a Inteligência Artificial pode ser integrada na aprendizagem da programação contribuindo para o desenvolvimento de competências práticas e melhoria na aprendizagem de conceitos complexos de programação. É necessário pensar num sistema que adapte os conteúdos conforme as dificuldades e o progresso do aluno. De acordo com o estudo realizado as ferramentas de Inteligência Artificial podem ajudar a analisar o código identificando erros comuns e sugerindo melhorias o que facilita a aprendizagem mais prática e orientada. A Inteligência Artificial pode também ser utilizada para desenvolver cenários práticos de programação baseados em problemas reais para promover uma

aprendizagem mais ativa e colaborativa. As aplicações de Inteligência Artificial não procuram apenas a memorização da sintaxe, mas a compreensão das estruturas dos dados e algoritmos essenciais durante a aprendizagem da programação. O método demonstrado neste artigo simula a utilização de um "tutor inteligente" através do qual os alunos recebem assistência personalizada funcionando como se fosse um explicador humano. A utilização de ferramentas com Inteligência Artificial na educação da programação é vantajosa para a criação de ambientes de aprendizagem interativos e adaptativos que atendem às necessidades individuais dos alunos promovendo a autonomia. Este estudo desenvolvido reforça a importância de capacitar os alunos que aprendem a programação com ferramentas avançadas as quais auxiliam um desenvolvimento contínuo e eficaz.

No livro *Artificial Intelligence in Education: Promise and Implications for Teaching and Learning*, Holmes et al. (2019) abordam o contexto da aprendizagem da programação e referem que a Inteligência Artificial é vista como essencial no suporte personalizado aos alunos e na forma como facilita a compreensão dos conceitos mais complexos. Discutem como a Inteligência Artificial pode transformar o ensino fornecendo sistemas inteligentes de apoio e feedback que se adaptam ao ritmo e nível de compreensão de cada estudante. Este tipo de sistema não só reforça a aprendizagem dos conceitos fundamentais, mas também promove o desenvolvimento de capacidades ao nível do pensamento computacional algo essencial na aprendizagem de linguagens de programação como o C. Os autores ainda sublinham a importância de uma abordagem mais equilibrada em que a utilização da Inteligência Artificial é utilizada apenas como ferramenta de apoio ao invés de substituir o papel do professor. A Inteligência Artificial é eficaz na medida em que ajuda a criar ambientes de aprendizagem mais inclusivos e interativos onde os alunos aprendem a explorar e a aplicar conceitos em diferentes contextos. Para os autores, este tipo de abordagem é extremamente importante para o ensino da programação porque facilita a prática e a experimentação como elementos cruciais na consolidação de capacidades a nível de linguagens de programação como o C.

No livro *Intelligence Unleashed: An argument for AI in Education*, Luckin et al. (2016) destacam o papel da Inteligência Artificial como sendo uma ferramenta promissora para a educação, especialmente no ensino da programação uma vez que oferece formas de personalizar o processo de aprendizagem e melhorar a compreensão dos conceitos mais

complexos. Os autores do livro também referem a importância das ferramentas de Inteligência Artificial na criação de ambientes de aprendizagem adaptativos que se ajustam ao ritmo e nível de compreensão de cada aluno. Através da Inteligência Artificial que disponibiliza sistemas de feedback em tempo real é possível aos estudantes receber apoio de forma a identificar os erros e orientação sobre como corrigi-los o que é crucial na aprendizagem da programação. Para os autores, este sistema inteligente é relevante para o ensino das linguagens de programação onde os alunos enfrentam dificuldades na compreensão da sintaxe, lógica e estrutura do código. A Inteligência Artificial fornece *feedback* imediato e promove uma aprendizagem ativa ajudando os estudantes a desenvolver as suas competências de forma autónoma e adaptada às suas necessidades. As ferramentas de Inteligência Artificial vieram transformar o ensino da programação através de uma experiência mais inclusiva e personalizada preparando os alunos para a resolução de problemas e consolidação de capacidades fundamentais.

De acordo com o relatório da OECD (2019) a metodologia abordada destaca-se por promover o desenvolvimento das competências digitais e cognitivas através da integração da Inteligência Artificial e de ferramentas digitais durante o processo de aprendizagem. Esta abordagem metodológica permite não só que os alunos aprendam de forma eficaz, mas também que desenvolvam capacidades críticas e práticas tais como o pensamento computacional e a resolução de problemas. A integração de tecnologias digitais na educação deve ser acompanhada de uma metodologia que facilite a aprendizagem. Estes tipos de metodologias focam-se na utilização da Inteligência Artificial para criar ambientes de aprendizagem interativos onde os alunos podem progredir ao seu próprio ritmo. A utilização de ferramentas de Inteligência Artificial na aprendizagem da programação, facilita a utilização de plataformas como o Dev C++ na medida em que melhora o processo de aprendizagem, oferecendo aos alunos a oportunidade de aprender os conceitos de programação de forma prática, mas também oferecem a oportunidade de desenvolver e testar código, contribuindo assim para o desenvolvimento das competências em programação.

7.8 Metodologia de Investigação

A utilização de metodologias mistas que combinam abordagens quantitativas e qualitativas para a recolha e análise de dados permite uma interpretação e compreensão mais

abrangente, de acordo com o descrito por Creswell e Clark (2014). Segundo Bell e Waters (2018) os questionários são instrumentos acessíveis e de fácil aplicação e interpretação sobretudo em contextos escolares porque permitem a recolha dos dados de forma estruturada e garantem uma elevada taxa de resposta. A abordagem reflete a utilização de metodologias mistas, integrando a recolha de dados quantitativos e qualitativos, para permitir uma análise mais abrangente e detalhada do fenómeno estudado, conforme descrito por Creswell e Clark (2014).

- **Instrumento**: Questionários estruturados, com perguntas fechadas (escalas de *Likert*) e abertas para análise de exploração de perceções.
- ***Software* utilizado**: O software utilizado para a aplicação dos questionários foi o *Google Forms* que pode ser acedido através de qualquer dispositivo com acesso à internet e também uma análise rápida e estruturada das respostas.
- **Participantes**: Alunos que estão a aprender pela primeira vez programação em C.
- **Turma:** 14 alunos que divergem em vários níveis de competência e conhecimentos, incluindo alunos com percursos de necessidades educativas especiais.
- **Método**: Aplicação de um questionário online com aspetos quantitativos e qualitativos.
- **Revisão da Literatura**: Revisão da literatura **extensa** para garantir uma fundamentação adequada e significativa da investigação realizada.

Este estudo aplica uma abordagem exploratória porque utiliza instrumentos quantitativos e qualitativos para a recolha dos dados sobre as perceções, motivações e experiências dos alunos.

A turma apresenta uma diversidade diferente em termos técnicos, níveis de motivação e necessidades educativas diferentes e os questionários permitem captar as especificidades individuais de cada aluno. A utilização do formato *online* facilita o acesso sobretudo para alunos com algumas dificuldades de expressão oral e que são mais introvertidos (Bryman, 2016).

Tendo como objetivo a preparação da prática pedagógica, aplicou-se um primeiro questionário antes do início das aulas supervisionadas e um segundo questionário após o término da prática. O questionário inicial servirá apenas como um diagnóstico, permitindo uma visão clara do perfil inicial da turma, com o objetivo de analisar o perfil, os níveis de motivação, as competências técnicas e as perceções dos alunos sobre as suas dificuldades na aprendizagem da programação em linguagem C.

Este relatório está de acordo com as recomendações de Creswell e Clark (2014), no livro *Research Design: Qualitative, Quantitative, and Mixed Methods Approaches*, uma vez que adota uma metodologia mista que combina abordagens quantitativas e qualitativas, que vão ao encontro dos objetivos desta pesquisa tendo em conta as características da turma a lecionar. A metodologia mista é amplamente reconhecida na literatura como uma forma robusta de investigar permitindo assim a recolha dos dados aumentando a validade e profundidade dos resultados obtidos. Os autores destacam a importância da metodologia mista como uma forma de pesquisa nas ciências sociais e humanas devido à sua capacidade de integrar os métodos quantitativos e qualitativos para uma análise completa do problema de investigação. A combinação dos dois métodos aumenta a compreensão do problema. A metodologia mista também adota um estudo empírico com o objetivo de avaliar o impacto dos cenários de aprendizagem. Este relatório também realizará um estudo empírico com o objetivo de avaliar as competências e os conhecimentos dos alunos relativamente à aprendizagem da linguagem de programação em C. Tendo em conta que a aprendizagem da linguagem de programação se apoia na resolução de problemas torna-se necessário realizar um estudo empírico na turma onde se realizará a prática supervisionada.

Os questionários são instrumentos acessíveis e de fácil aplicação e interpretação sobretudo em contextos escolares porque permitem a recolha dos dados de forma estruturada e garantem uma elevada taxa de resposta conforme defendido por Bell e Waters (2018).

Bryman (2016), no seu livro *Social Research Methods,* descreve a importância das perguntas categóricas como uma forma de recolha de dados através de questionários. As perguntas categóricas oferecem aos respondentes, opções de resposta através de categorias ou frases permitindo assim ao investigador recolher informações sobre as suas preferências e

atitudes sem estabelecer uma ordem hierárquica das respostas. Este método apresenta as seguintes vantagens: a simplicidade na recolha e análise dos dados, facilidade em resumir os dados através de tabelas de frequência ou até gráficos de barras.

7.8.1 Justificação da metodologia mista utilizada neste estudo de investigação defendida por Creswell e Plano Clark (2014)

A metodologia mista permite a recolha de dados quantitativos com o objetivo de fornecer uma visão geral de tendências e padrões. Ao mesmo tempo, a recolha de dados qualitativos permite-nos obter uma visão mais detalhada das perceções, experiências e sentimentos dos alunos relativamente à aprendizagem da programação. Quando identificamos padrões de motivação analisamos dados quantitativos e quando compreendemos as perceções e as dificuldades dos alunos estamos a analisar dados qualitativos. Este tipo de metodologia mista é muito útil porque permite explorar diversos tipos de perguntas e explorar os resultados com uma maior profundidade. Os questionários abordam questões sobre as dificuldades dos alunos em programação, juntamente com perguntas de resposta aberta para permitir identificar os motivos das dificuldades dos alunos. A combinação de diversos métodos aumenta a validade dos resultados. Ao cruzarmos os dados quantitativos com dados qualitativos é possível analisar e avaliar se os dados são consistentes. Quando cruzamos os dados podemos identificar contradições ou até discrepâncias ao nível das respostas dos alunos. Por exemplo: quando identificamos um grupo de alunos que está menos motivado para aprender programação em linguagem C, os dados quantitativos dizem-nos o número de alunos, no entanto são os dados qualitativos que nos permitem identificar as razões que levam os alunos a estar menos motivados para aprender programação em linguagem C.

Segundo os autores, os questionários são ferramentas fundamentais quando se utiliza uma metodologia quantitativa. As metodologias quantitativas permitem descrever tendências, atitudes e opiniões de uma determinada população. Os questionários devem ser cuidadosamente planeados e incluir perguntas fechadas que facilitem a análise estatística dos dados. Este tipo de questionário permite a recolha de dados mais objetivos economizando tempo e recursos em comparação com os métodos qualitativos como o caso das entrevistas. No entanto apresenta algumas limitações uma vez que pode existir falta de profundidade nas

respostas porque as opções são pré-definidas e os respondentes podem não interpretar as perguntas da mesma forma e até responder da forma mais honesta possível.

Para esta análise, foram utilizadas escalas de *Likert* para medir as atitudes, sentimentos e perceções de forma estruturada, conforme proposto por Likert (1932). Através das escalas os respondentes indicam o seu grau de concordância ou discordância numa escala desde *discordo totalmente* a *concordo totalmente*. Este tipo de escalas é convertido em sequências ordinais, permitindo assim a utilização de estatísticas descritivas.

Esta abordagem possibilitou confrontar os dados de ambos os questionários, destacando os impactos das estratégias pedagógicas implementadas. Esta análise comparativa oferecerá uma base sólida para identificar as práticas que facilitaram o processo de aprendizagem e como podem ser aprimoradas/melhoradas para futuras intervenções. A metodologia utilizada na recolha dos dados tem como objetivo avaliar o perfil, a motivação, as perceções e as experiências dos alunos relativamente à aprendizagem da programação em C durante as aulas supervisionadas. Recorreu-se a medições, baseadas nas escalas de *Likert* para medir as atitudes, sentimentos e perceções de forma estruturada.

No final da prática pedagógica foi aplicado um segundo questionário que terá como objetivo principal constatar como as ferramentas utilizadas do Dev C++ e a Inteligência Artificial influenciaram a aprendizagem, verificando se existiram mudanças nas atitudes e nos resultados desenvolvidos pelos alunos. A aplicação deste tipo de metodologia tem como objetivo principal dar resposta às duas questões de investigação formuladas neste relatório de estágio da prática supervisionada.

7.9 A Influência dos Encarregados de Educação na motivação dos alunos

De acordo com Lourenço (2008), o envolvimento dos encarregados de educação durante o processo de aprendizagem é um fator crucial para o sucesso dos alunos. Famílias de estratos sociais mais elevadas proporcionam aos seus filhos melhores recursos e possibilidades de sucesso. Famílias de estratos sociais mais baixas enfrentam maiores barreiras como a limitação de tempo, menor capacidade financeira e cultural. No entanto, as escolas desempenham um

papel fundamental e determinante na medida em que devem promover estratégias de inclusão de forma a permitir a estas famílias mais carenciadas ultrapassar estas barreiras tanto a nível financeiro como a nível cultural. Para Lourenço (2008), é também fundamental a participação ativa dos encarregados de educação durante todo o percurso escolar dos alunos.

Para Cardoso e Lamas (2021), "os encarregados de educação desempenham um papel crucial no acompanhamento escolar dos seus educandos" (p. 15). O envolvimento ativo dos pais durante todo o processo de ensino e aprendizagem contribui significativamente para uma melhoria no desempenho escolar. Famílias de estratos sociais mais elevados têm acesso a melhores recursos, capital cultural mais elevado e fornecem um apoio mais eficiente aos seus educandos. Pelo contrário, as classes menos favorecidas economicamente, culturalmente e socialmente enfrentam obstáculos no que se refere à prestação de apoio aos seus educandos devido às limitações culturais e económicas.

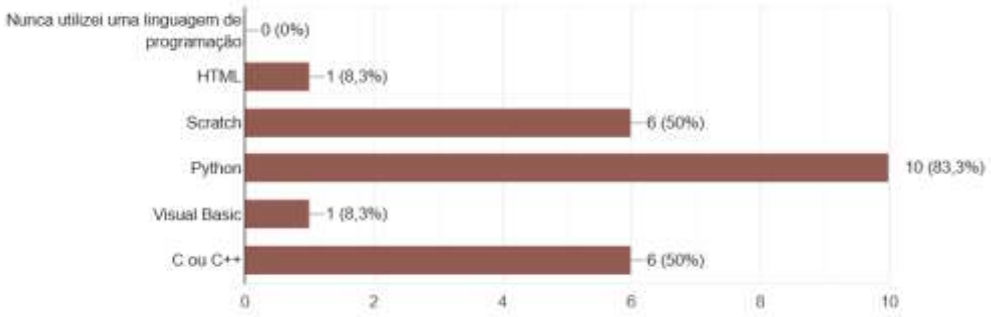
7.10 Fase diagnóstico na recolha de dados antes da prática supervisionada

Foi feito um diagnóstico inicial através do qual foi aplicado um questionário para analisar as perceções dos alunos relativamente aos conhecimentos da programação em linguagem C, relativamente ainda ao ambiente de programação e também na integração das ferramentas de Inteligência Artificial durante as aulas práticas. Os alunos inicialmente estavam a ter o primeiro contacto com a programação, a Inteligência Artificial e o programa DEV C++, o que permitiu compreender melhor as dificuldades e as necessidades individuais dos alunos durante o processo de aprendizagem. Após uma análise cuidada dos resultados obtidos através do questionário aplicado aos alunos da turma do 1.º L foi possível concluir que são diversos os fatores que influenciam o perfil, a motivação e as perceções dos alunos na forma como aprendem a linguagem de programação em C. A turma apresenta diversidade de conhecimentos e experiências diferentes o que afeta diretamente a forma como cada aluno aborda esta disciplina.

É possível através da (**Figura 3**) compreender que grande parte dos alunos já teve contacto com a programação em C e Python, assim como Scratch, no entanto o conhecimento

dos alunos é limitado, indicando uma forte necessidade de reforço relativamente aos fundamentos da programação. Ensinar os fundamentos de programação a esta turma tanto poderá ser uma vantagem como um desafio para o professor tendo em conta que terá de adaptar metodologias de ensino diferentes de acordo com o ritmo e a aprendizagem dos alunos.

Figura 3
Quais são os teus conhecimentos ao nível das linguagens de programação?



Relativamente à questão sobre a familiaridade com as linguagens de programação, uma pequena maioria (58,3%) dizem que já utilizaram linguagens de programação, no entanto uma minoria significativa (41,7%) diz já ter utilizado linguagens de programação, mas desconhecem quase tudo acerca do tema. Os dados permitem-nos concluir que apesar dos alunos terem contacto com a linguagem de programação, a maioria destes alunos carece de conhecimentos aprofundados. O professor deve adotar uma estratégia baseada numa revisão dos fundamentos da programação para os alunos menos confiantes e com mais dificuldades. A utilização de uma metodologia mais prática será essencial para aumentar o conhecimento dos alunos da turma.

Relativamente ao interesse e motivação demonstrado pelos alunos da turma, os dados recolhidos do questionário permitem inferir que existem diferentes interesses e tipos de motivação. Através da (**Figura 4**) podemos analisar que (41,7%) demonstram interesse em resolver problemas mais práticos e valorizam a resolução de desafios concretos, (25%) destes alunos vê a base da programação como um passo principal e essencial, (16,7%) vê a programação numa perspetiva de a explorar de forma geral e apenas (16,7%) vê a programação como uma oportunidade de desenvolvimento académico e profissional.

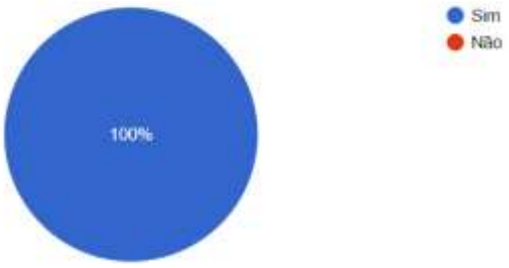
Figura 4

Quais dos seguintes aspetos da programação em C despertam o teu interesse e motivação?

A maioria dos alunos (100%) sugere a integração desta tecnologia durante o processo de ensino da prática supervisionada por forma a melhorarem a aprendizagem (**Figura 5**). O tema da Inteligência Artificial é visto por todos os alunos como uma ferramenta benéfica, na medida em que fornece apoio e feedback imediatos.

Figura 5

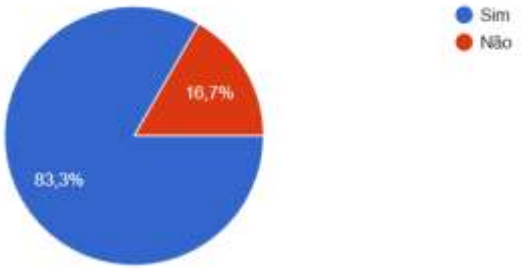
Consideras que as Plataformas de Inteligência Artificial podem ser uma mais-valia na aprendizagem da programação em C?



A maioria dos alunos (83%) gostaria que o professor utilizasse plataformas de Inteligência Artificial durante as suas aulas práticas supervisionadas enquanto apenas (16,7%) responderam que não querem que o professor utilize ferramentas de Inteligência Artificial durante a prática de ensino (**Figura 6**). Podemos concluir que há um consenso significativo para a utilização deste tipo de ferramentas durante as aulas.

Figura 6

Gostarias que o professor utilizasse alguma plataforma de Inteligência Artificial durante a prática de ensino?



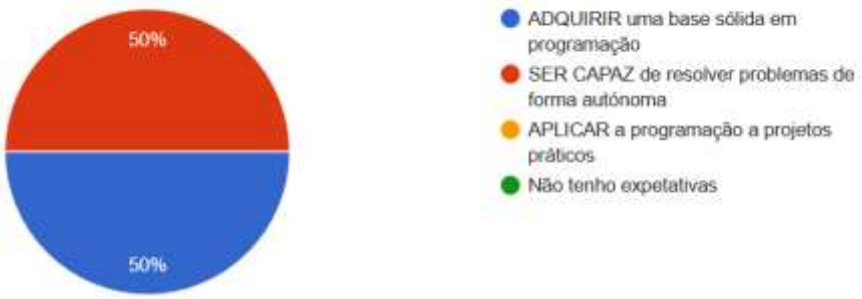
Quando questionados sobre a forma como gostariam que o professor lecionasse as suas aulas de programação, as opiniões divergem sendo que (33,3%) dos alunos preferem trabalho de grupo/projeto e outros (33,3%) preferem exercícios práticos com acompanhamento, (25%) dos alunos preferem aulas expositivas com explicações detalhadas e apenas (8,3%) optam pelo estudo autónomo. Claramente a maioria dos alunos prefere metodologias mais práticas através da resolução de exercícios. As aulas expositivas são menos valorizadas e os métodos ativos mais valorizados.

Relativamente à confiança e preparação para a aprendizagem da programação, (50%) dos alunos sentem-se preparados, mas têm receio de enfrentar dificuldades, (33,3%) não têm certeza, mas estão interessados e apenas (16,7%) estão entusiasmados e preparados. Torna-se importante que o professor seja capaz de criar um ambiente de aprendizagem que promova a confiança dos alunos e apoio contínuo. Este apoio é particularmente importante para aqueles alunos que se sentem menos confiantes e com mais dificuldades.

Quando questionados sobre as suas expectativas relativamente à aprendizagem da programação foi possível obter os seguintes resultados: metade da turma, (50%) dos alunos espera adquirir uma base sólida em programação e a outra metade da turma, (50%) esperam ser capazes de resolver problemas de forma autónoma. A turma espera ser capaz de utilizar ferramentas como o Dev C++ com o objetivo de desenvolver programas funcionais. As expectativas destes alunos estimulam a aprendizagem (**Figura 7**).

Figura 7

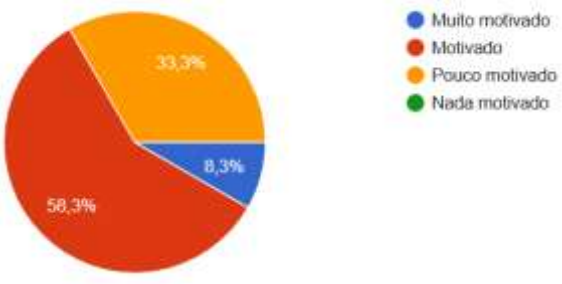
Quais são as tuas expetativas em relação às aulas de programação em C?



A maioria dos alunos da turma (58,3%) sente-se motivada para aprender diferentes tipos de dados, apenas (8,3%) se sentem muito motivados e (33,3%) sentem-se pouco motivados, (**Figura 8**). Podemos analisar que apesar de existir uma maioria de alunos motivada, existe ainda um grupo considerável de alunos com uma motivação baixa e que requer atenção por parte do professor. O professor deverá implementar abordagens mais inclusivas e interativas com o intuito de aumentar o interesse dos alunos focando-se mais nos alunos com mais dificuldades.

Figura 8

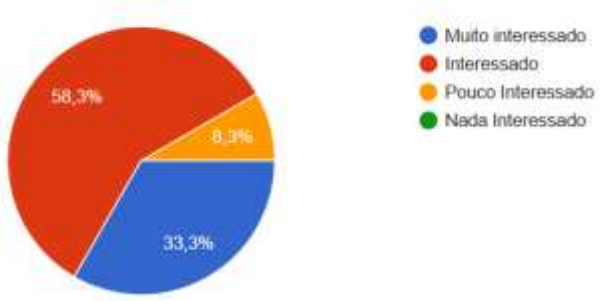
Quão motivado estás para aprender sobre os diferentes tipos de dados em C, como armazenar números decimais ou outros valores?



Mais de metade (58%) destes alunos sentem-se interessados na utilização do programa Dev C++ para a criação de funções em C, no entanto (33%) estão muito interessados e (8%) destes alunos sentem-se pouco interessados (**Figura 9**). Podemos concluir que a maioria dos alunos estão interessados em aprender programação através do programa Dev C++ e aprender a utilizar funções em linguagem C.

Figura 9

Estás interessado em aprender a exibir resultados no programa DeV C++ utilizando funções em C?



7.11 Dificuldades identificadas antes da prática supervisionada

Palavras-chave identificadas acerca das dificuldades encontradas:

- Falta de motivação dos alunos.
- Falta de conhecimentos a nível das linguagens de programação em C.
- Receio de enfrentar dificuldades na aprendizagem da linguagem de programação em C.
- Complexidade da aprendizagem da programação em C é sobretudo para alunos que estão a aprender pela primeira vez.
- Diversidades a níveis culturais e socioeconómicos.
- Necessidades educativas específicas de alguns alunos.

Através da análise dos questionários aplicados antes da intervenção das aulas supervisionadas foi possível identificar diversas dificuldades enfrentadas pelos alunos no início do processo de aprendizagem. Uma das dificuldades identificada como sendo a mais evidente foi a disparidade de conhecimentos em programação existente nesta turma. Alguns alunos já tiveram anteriormente contacto prévio com linguagens de programação, no entanto a maioria dos alunos possui uma experiência superficial e desconhecem quase tudo sobre o tema da programação. Esta disparidade de conhecimentos, dificultou a preparação e criação de materiais adequados, uma vez que alguns alunos necessitavam de uma abordagem mais

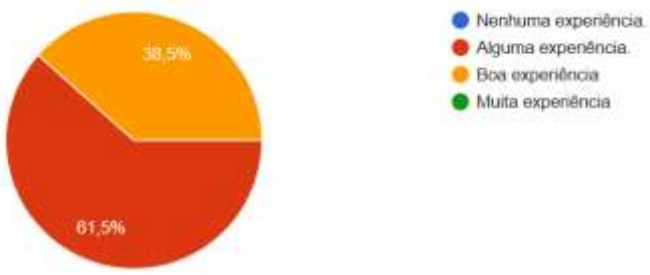
introdutória, sendo que outros já possuíam uma base inicial. Ainda foi possível identificar a falta de confiança, receio e dificuldades na aprendizagem da programação em linguagem C. Muitos alunos da turma demonstraram interesse, no entanto indicaram que se sentem inseguros quanto à sua capacidade de aprender programação. Este tipo de insegurança gera barreiras psicológicas e falta de motivação o que dificulta a assimilação dos conteúdos e a capacidade de enfrentar desafios mais complexos. Outro fator identificado foi a baixa motivação dos alunos para aprender a programar. Embora a maioria da turma se mostrasse motivada, um terço da turma indicou estar pouco motivada para aprender programação, sendo que uma pequena percentagem afirmou sentir-se altamente motivada. Esta falta de entusiasmo pode ser a perceção, de que a programação é uma disciplina difícil e inacessível, o que pode levar a um envolvimento fraco dos alunos durante as atividades práticas e tarefas realizadas durante as aulas. Ainda através dos resultados do questionário foi possível aferir que a ferramenta do Dev C++ não foi aceite de forma unânime por todos os alunos desta turma. Enquanto a maioria dos alunos aceitaram e demonstraram interesse em utilizá-la, ainda existe um pequeno grupo de alunos que revelou pouco entusiasmo na utilização da mesma. Este fator pode comprometer a prática de ensino supervisionada, caso os alunos não se sintam confortáveis com a utilização desta ferramenta de *software*. Por fim ainda foi possível identificar, que os fatores externos tais como o contexto socioeconómico e o acesso limitado a recursos tecnológicos, impactam diretamente no desempenho e aprendizagem dos alunos. É necessário ter atenção aos alunos que não possuem acesso a computadores e que em casa não recebem apoio familiar, o que pode dificultar a progressão das aprendizagens. Perante estas dificuldades, é necessário que o professor adapte estratégias diferenciadas que permitam atender às necessidades destes alunos assim como aos diferentes níveis de conhecimento da turma, com o objetivo de aumentar a confiança e estimular a motivação para a aprendizagem da programação.

7.12 Análise dos resultados após intervenção da prática supervisionada

Através da (**Figura 10**) podemos analisar o nível de experiência dos alunos relativamente à programação em C com base na recolha das respostas. A maioria dos alunos (61,5%) revela ter alguma experiência, o que sugere que houve aprendizagem desde o início do ano letivo e

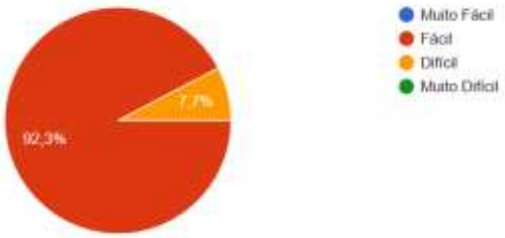
desde o primeiro momento que tiveram contacto com a linguagem de programação em C. (38,5%) destes alunos da turma revelaram ter boa experiência, o que indica que este grupo de alunos tem boa experiência com a linguagem de programação em C.

Figura 10
Nível de experiência com a programação em C?



Relativamente à questão sobre a frequência de utilização da ferramenta Dev C++, obteve-se uma maioria de alunos, (77%) que utilizam frequentemente ou sempre quando estão a programar C. Cerca de (23,1%) dos alunos desta turma indicaram que utilizam ocasionalmente, o que revela que estes alunos utilizam outras ferramentas ou que não programa com tanta frequência. Através da (**Figura 11**), podemos concluir que (92,3%) a maioria dos alunos da turma consideram a ferramenta do Dev C++ como sendo fácil de utilizar para aprenderem a programar. No entanto (7,7%) destes alunos considera o programa do Dev C++ como sendo difícil de utilizar para programar.

Figura 11
Sentes que o Dev C++ é mais fácil de utilizar para aprenderes a programar?

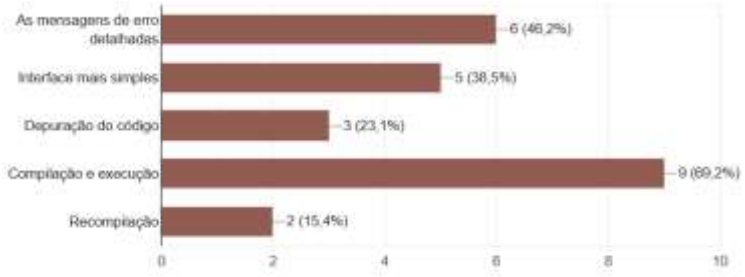


De forma a validar a coerências dos dados, foi realizada uma segunda questão sobre as dificuldades ao utilizar o Dev C++, sendo que (92,3%) dos alunos da turma responderam não sentem dificuldades e apenas (7,7%) relataram que sentem dificuldades. Obtiveram-se os mesmos resultados que na questão anterior, o que prova que os alunos tiveram coerência na forma como responderam ao questionário realizado.

Na (**Figura 12**), podemos analisar as diversas opiniões dos alunos sobre as funcionalidades do Dev C++ que consideram úteis na aprendizagem da programação em linguagem C. (69,2%) destacam o processo rápido e eficiente da compilação e execução do código, parte essencial para o programador testar e aprender C de forma prática. (46,2%) quase metade dos alunos da turma considera as mensagens de erro detalhadas como sendo uma funcionalidade muito útil. Identificar os erros ajuda os alunos iniciantes em programação a compreender problemas no código e a aprender a corrigir esses mesmos erros. (38,2%) considera a interface do Dev C++ como um fator importante na medida em que funciona como facilitador da aprendizagem uma vez que a interface é clara e objetiva. Apenas (23,1%) dos alunos consideram a funcionalidade essencial, o que pode indicar-nos que ainda não exploraram ou não utilizaram o depurador para identificar e corrigir erros lógicos de programação. (15,4%) consideraram a recompilação como a funcionalidade essencial o que pode significar que ainda não exploraram as outras funcionalidades que a ferramenta apresenta.

Figura 12

Quais as funcionalidades do Dev C++ que consideras mais úteis na aprendizagem do C?

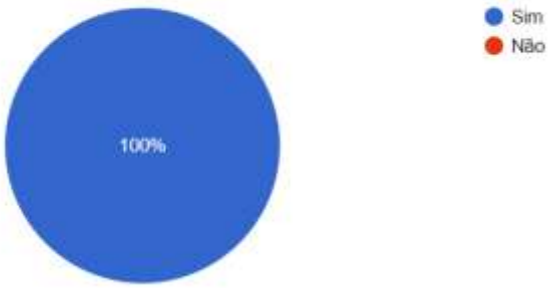


Através da (**Figura 13**) podemos concluir que (100%) todos os alunos da turma acreditam que as ferramentas de Inteligência Artificial são úteis durante a aprendizagem e para

resolver problemas em linguagem C. Estes resultados mostram uma tendência positiva e crescente na utilização deste tipo de tecnologias para o ensino da programação.

Figura 13

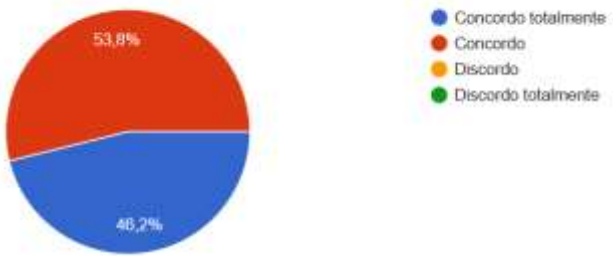
Sentes que as ferramentas de Inteligência Artificial te ajudam a aprender e a resolver problemas em C?



Ao analisarmos a (**Figura 14**), podemos constatar que existe por parte dos alunos uma unanimidade relativamente à utilização da Inteligência Artificial em sala de aula. (46,2%%) concordam totalmente que ao aprender programação deve-se complementar com o apoio deste tipo de ferramentas. No entanto existe um grupo mais cauteloso (53,8%) a maioria dos alunos que concordam e vêm a Inteligência Artificial como útil, mas ainda mantêm algumas reservas sobre a utilização dentro da sala de aula.

Figura 14

Concordas que as ferramentas de Inteligência Artificial devem fazer parte das aprendizagens de programação em C dentro da sala de aula?

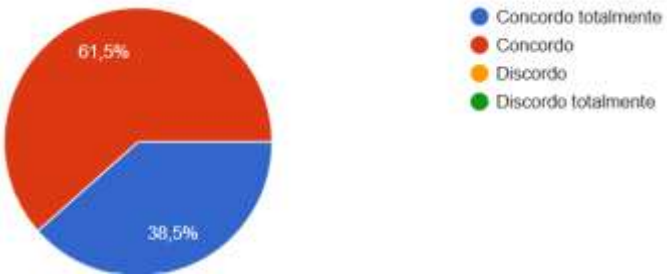


Quando questionados sobre a utilização de ferramentas de Inteligência Artificial (38,5%) dos alunos responderam que concordam totalmente e (61,5%) apenas responderam que concordam (**Figura 15**). Existe um consenso de que a Inteligência Artificial ajuda durante a

aprendizagem da programação e os alunos veem a ferramenta como sendo útil e eficaz, no entanto (61,5%) destes alunos reconhecem os benefícios da utilização deste tipo de ferramentas, mas acreditam que a Inteligência Artificial não substitui totalmente outras formas de aprendizagem.

Figura 15

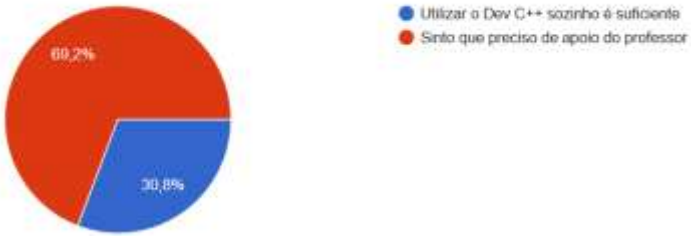
Na tua opinião a utilização de ferramentas de Inteligência Artificial tornam mais fácil a aprendizagem da programação?



Através da (**Figura 16**) podemos aferir que (30,8%) sentem que aprender sozinho a utilizar a ferramenta do Dev C++ é suficiente para aprender programação. Indica-nos que estes alunos sentem necessidade de aprender através de maior autonomia ou sentem-se confiantes e capazes de aprender por si próprios. No entanto a maioria dos alunos da turma (69,2%) considera necessário o apoio do professor, mostrando que a aprendizagem da programação não está dependente de uma ferramenta, mas também da orientação do professor dentro da sala de aula.

Figura 16

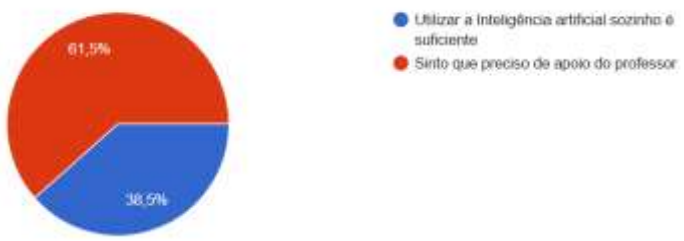
Consideras que o Dev C++ é suficiente para aprenderes programação? Ou sentes necessidade de apoio do teu professor?



Na (**Figura 17**) os alunos quando questionados sobre se sozinhos, por si próprios conseguem aprender a programar sem o apoio do professor foi possível aferir que (38,5%) dos alunos desta turma concordam e consideram que utilizar ferramentas de Inteligência Artificial sozinhos é suficiente para aprenderem programação. No entanto (61,5%), a maioria dos alunos da turma considera o contrário e acredita que precisam do apoio do professor para aprenderem a programar.

Figura 17

Consideras que sozinho com a ajuda da Inteligência artificial consegues aprender programação sem o apoio do teu professor?



Quando os alunos foram questionados sobre avaliação e satisfação em geral durante o processo de aprendizagem em C utilizando as ferramentas do Dev C++ em conjunto com as ferramentas de Inteligência Artificial, obtivemos uma taxa elevada de respostas positivas, sendo que (76,9%) dos alunos sentem-se satisfeitos e (23,1%) sentem-se muito satisfeitos. 100% dos alunos da turma estão satisfeitos ou muito satisfeitos com o processo de aprendizagem, no entanto (76,9%) a maioria da turma não se sente muito satisfeita o que poderá indicar que existe espaço para melhorias. No entanto nenhum aluno da turma indicou insatisfação mostrando que a combinação da ferramenta do Dev C++ aliada às ferramentas de Inteligência Artificial proporciona uma experiência de aprendizagem muito positiva.

7.13 Dificuldades identificadas após prática supervisionada

Palavras-chave identificadas acerca das dificuldades encontradas:

- Disparidade no nível de experiência e conhecimento dos alunos.
- Dificuldades na adoção total do Dev C++ por todos os alunos.

- Algum receio na utilização de ferramentas de Inteligência Artificial.
- Dependência da presença do professor para o processo de aprendizagem.
- Dificuldade na interpretação de mensagens de erro no Dev C++ por alguns alunos da turma.
- Falta de confiança e motivação de alguns alunos da turma relativamente à programação.

Através da análise dos resultados do questionário aplicado aos alunos da turma, foi possível identificar diversas dificuldades enfrentadas durante a intervenção das aulas práticas supervisionadas. Uma das principais dificuldades encontradas foi a disparidade no nível de experiência e conhecimento dos alunos. A turma diferencia-se na medida em que alguns alunos demonstraram evolução e maior familiaridade com a linguagem C, enquanto outros mantiveram as dificuldades em acompanhar o ritmo de aprendizagem, tornando importante o suporte individualizado prestado pelo professor durante as aulas. Outra dificuldade identificada está relacionada com a adoção do *software* utilizado pelo professor para a aprendizagem da programação em linguagem C. Apesar da maioria destes alunos ter utilizado sempre a ferramenta do Dev C++ durante o seu processo de aprendizagem, ainda houve um grupo de alunos que preferiu utilizar a ferramenta de forma ocasional. Podemos interpretar que estes alunos têm dificuldades na sua utilização ou até a preferência por outras plataformas, o que afeta a uniformidade do ensino. Ainda foi possível analisar através dos resultados do questionário que existe um pequeno grupo de alunos que sentiu dificuldades na interpretação das mensagens de erro apresentadas pelo programa do Dev C++ o que dificulta a correção de problemas identificados no código e gera frustração durante o processo de aprendizagem. Além disso, foi observada uma dependência excessiva da execução do código sem análise prévia, indicando falta de autonomia na programação. Em vez de antecipar e corrigir os erros antes da compilação, muitos alunos confiaram no compilador para identificar as falhas, sem compreender a origem dos problemas. Para superar esta dificuldade, é essencial desenvolver primeiro capacidades ao nível da depuração *(*`debugging`*)* e incentivar os alunos a ler e interpretar as mensagens de erro antes de executar o código. Relativamente à utilização de ferramentas de Inteligência Artificial, verificou-se que apesar de todos os alunos da turma

reconheceram a utilidade deste tipo de ferramenta para o ensino da programação, ainda têm receio na sua utilização. Os resultados do questionário demonstram que estes alunos não confiam plenamente nas sugestões fornecidas pela Inteligência Artificial, o que demonstra que se preocupam com a precisão das respostas e com a forma como podem validar os conteúdos apresentados pela Inteligência Artificial. A turma analisa de forma critica e responsável as respostas fornecidas pela Inteligência Artificial. Apesar da existência de ferramentas tecnológicas tais como a Inteligência Artificial, a maioria dos alunos desta turma preferem e sentem necessidade de apoio constante do professor para aprenderem a programar, o que nos indica que estes alunos ainda não são capazes de aprender sozinhos de forma autónoma. Outra dificuldade identificada é a falta de confiança e motivação de alguns alunos relativamente à aprendizagem da programação. A complexidade da linguagem de programação em C, aliada à insegurança e falta de confiança dos alunos tem impacto no desempenho de alguns estudantes que frequentam esta turma. Perante estas dificuldades, torna-se evidente a necessidade de adaptar estratégias para incentivar um ensino inclusivo, com o objetivo de promover o equilíbrio entre a autonomia do aluno e o apoio prestado pelo professor dentro da sala de aula. A integração de ferramentas de Inteligência Artificial com a utilização do Dev C++, ajuda a minimizar os riscos e desafios encontrados pelos alunos proporcionando uma experiência de aprendizagem acessível a todos os estudantes.

7.14 Resultados da Investigação

A realização de uma comparação entre estes dois momentos principais de investigação, permite-nos perceber se houve evolução no processo de aprendizagem dos alunos relativamente ao ensino da programação em linguagem C e se as dificuldades inicialmente identificadas foram superadas. Esta comparação entre os resultados iniciais e os obtidos após a intervenção das aulas práticas supervisionadas, permitiu confirmar o impacto positivo nas aprendizagens dos alunos relativamente à programação em linguagem C. Através desta investigação, foi possível demonstrar que o perfil e a motivação dos alunos influenciam diretamente o seu desempenho durante a aprendizagem. A utilização de ferramentas como o Dev C++ aliadas com ferramentas de Inteligência Artificial contribuiu para tornar a aprendizagem mais simples e prática, no entanto a presença do professor continua a ser fundamental para garantir o equilíbrio do ensino. Os resultados da investigação, reforçam a

necessidade de introdução de metodologias flexíveis e diferenciadas prevendo assim as dificuldades individuais dos alunos e assim promover um ambiente de aprendizagem inclusivo e estimulante.

As principais melhorias observadas foram:

- Aumento do conhecimento dos alunos e da confiança.
- Aumento de níveis de motivação.
- Adoção e utilização frequente do *software* Dev C++.
- Utilização de ferramentas de Inteligência Artificial durante as aulas.
- Redução das desigualdades de acesso às tecnologias digitais.

Os alunos da turma perderam o receio que tinham inicialmente da programação e começaram a sentir-se mais preparados para enfrentar desafios complexos. A experiência de aprendizagem tornou-se mais envolvente, refletindo-se na satisfação geral dos alunos. A ferramenta do Dev C++ passou a ser utilizada regularmente, facilitando a prática e a autonomia dos alunos. Os alunos reconheceram as vantagens de utilizarem a Inteligência Artificial como um recurso útil para a aprendizagem, no entanto sentem a necessidade do apoio do professor durante o processo de ensino. Durante a prática supervisionada foram garantidos acessos às tecnologias digitais a todos os alunos da turma, permitindo aos alunos a resolução dos exercícios durante os períodos da aula, minimizando assim desigualdades.

7.14.1 Impacto na motivação dos alunos antes e depois da intervenção

O baixo nível de motivação dos alunos desta turma para aprender programação era evidente e era um fator preocupante durante a fase inicial da investigação. Após a aplicação do questionário antes da prática supervisionada, identificou-se (33,3%) dos alunos desta turma relataram baixa motivação para aprenderem programação. Este desinteresse inicial, podia ter impacto no envolvimento destes alunos durante as atividades práticas e na progressão dos alunos ao longo da disciplina de programação. No entanto após a intervenção das aulas práticas supervisionadas, os níveis de motivação dos alunos melhoraram significativamente. Houve um aumento na aceitação das ferramentas utilizadas o que se refletiu na perceção dos alunos, que

passaram a considerar que a programação afinal é acessível desde que bem preparada e explicada pelo professor em sala de aula. Obtivemos uma percentagem de 76,9% dos alunos desta turma que relataram estar muito satisfeitos ou satisfeitos com o processo de aprendizagem. Este aumento significativo demonstra que a aula preparada e planeada antecipadamente permite a adequação de ferramentas de acordo com as necessidades e ritmos dos alunos, contribuindo para uma experiência positiva durante o ensino da programação. No entanto há espaço para melhorias assim como aperfeiçoar estratégias que mantenham o interesse dos alunos aumentando a predisposição inicial para aprender.

7.14.2 Superação de Barreiras de Acesso e Desigualdades

Durante a fase inicial, foram identificadas algumas barreiras externas que dificultaram o progresso de alguns alunos, tais como as diferenças socioeconómicas e acesso limitado à tecnologia fora da sala de aula. Apesar destas dificuldades não terem sido completamente eliminadas, a intervenção das aulas práticas supervisionadas minimizaram o impacto destas barreiras, através da resolução de exercícios sempre em sala de aula. A introdução deste tipo de metodologia favorece a prática e o apoio contínuo do professor, permitindo assim a mitigação de desigualdades, mas também que os alunos consigam acompanhar o ritmo da turma.

7.14.3 A adoção e utilização do *software* Dev C++

No início da investigação, nem todos os alunos estavam entusiasmados com a utilização desta ferramenta para programar, e alguns alunos relataram dificuldades durante a utilização da mesma. A falta de familiaridade com ambientes de desenvolvimento e a alta complexidade pode ter constituído uma barreira inicial para alguns estudantes, dificultando a sua adaptação. Contudo, ao longo das aulas práticas supervisionadas, os alunos pouco a pouco começaram a aceitar o ambiente de programação Dev C++. Após a intervenção, (77%) dos alunos da turma passaram a utilizá-lo frequentemente ou sempre para programar, e (92,3%) consideraram a ferramenta fácil de utilizar, relatando a importância das mensagens de erro detalhadas e a sua interface simples. Entre as diversas funcionalidades que o *software* apresenta as mais valorizadas pelos alunos são o processo de compilação e execução do código em C. Esta evolução transmite-nos que a prática e a habituação com o *software* permitiram aos alunos

sentirem-se mais confiantes. No entanto existe ainda um pequeno grupo de alunos que apresentaram dificuldades com as funcionalidades avançadas do Dev C++ tais como a depuração do código, o que indica a necessidade de estratégias adicionais para auxiliar estes alunos.

7.14.4 A Evolução no nível de conhecimento e confiança para programar em C

Durante a fase diagnóstica, foi possível identificar a disparidade nos níveis de conhecimento científico entre os alunos da turma. Alguns alunos já tinham tido contacto prévio com linguagens de programação e um grupo significativo de alunos desconhecia quase tudo sobre a linguagem C. Este desnível era preocupante uma vez que alguns alunos necessitavam de abordagens mais introdutórias, enquanto outros já possuíam uma base inicial. A falta de confiança era evidente, cerca de mais de metade dos alunos da turma expressaram receio e dificuldades com a programação, no entanto os restantes alunos demonstravam confiança e entusiasmo. Após as aulas práticas supervisionadas, verificou-se uma evolução significativa no nível do conhecimento e confiança dos alunos. A maioria começou a sentir-se mais confortável com a linguagem de programação em C, sendo que mais de metade da turma reconhecem que adquiriram alguma experiência e os restantes alunos da turma sentem-se ainda mais confiantes e preparados para enfrentar desafios ao nível da programação. O receio inicial foi superado, e houve evolução das aprendizagens destes alunos.

7.14.5 O Impacto da Inteligência Artificial na Aprendizagem

Na fase inicial, as ferramentas de Inteligência Artificial ainda não eram utilizadas em contexto de sala de aula, tornando-se uma novidade na abordagem pedagógica. Contudo a sua introdução ao longo da prática supervisionada foi amplamente aceite por todos os alunos. Após a intervenção, 100% dos alunos reconheceram que a Inteligência Artificial ajudou durante a sua aprendizagem e resolução de problemas em C, confirmando assim a sua eficácia como ferramenta de apoio durante o ensino. No entanto, verificou-se que 53,8% dos alunos ainda consideram que a Inteligência Artificial deve ser apenas um complemento, o que demonstra que, apesar de ser uma ferramenta útil, não substitui o papel do professor dentro da sala de aula. Os alunos compreenderam que a Inteligência Artificial pode ser utilizada para o esclarecimento de dúvidas e apoiar o raciocínio lógico, no entanto consideram fundamental

analisar criticamente e questionar as respostas fornecidas pela Inteligência Artificial em vez de as aceitarem de forma passiva. No entanto ainda existe uma necessidade de garantir a compreensão dos conceitos fundamentais da programação, dado que a turma está a aprender pela primeira vez a programar.

7.14.6 Aferição do progresso das aprendizagens dos alunos após prática supervisionada

A ficha de aferição teve como objetivo principal avaliar os conhecimentos dos alunos ao nível da programação modular em linguagem C, adquiridos ao longo das oito aulas. A ficha estava estruturada em duas partes: No primeiro grupo, os alunos tinham de responder questões de resposta múltipla, que testava os conceitos teóricos fundamentais e o segundo grupo era composto por questões de interpretação e análise de código, exigindo conhecimentos práticos adquiridos durante as aulas. Os resultados da ficha revelaram um desempenho global positivo, com uma média final de 16,15 valores, demonstrando a consolidação das aprendizagens. A maioria dos alunos apresentou resultados elevados no primeiro grupo da ficha de aferição, indicando que compreenderam os conceitos fundamentais da programação modular, tais como a estrutura de funções, a utilização correta de procedimentos, subprogramas assim como a utilização das bibliotecas padrão do C. No entanto no segundo grupo houve algumas dificuldades identificadas pelo facto das questões práticas exigirem a análise e interpretação de linhas de código, o que sugere ao professor a necessidade de rever mais exercícios práticos semelhantes com os alunos. Apesar das dificuldades encontradas, os alunos demonstraram progressos significativos em relação ao diagnóstico inicial. O envolvimento ativo destes alunos na resolução de problemas e aliado à utilização do Dev C++ e das tecnologias emergentes contribuiu para os resultados obtidos na ficha de aferição, provando a eficácia das metodologias utilizadas em sala de aula e destacando a importância da combinação da teoria com a prática para o ensino da programação.

8 Conclusões

A intervenção das aulas práticas supervisionadas teve como principal objetivo avaliar as diferentes estratégias de ensino durante a aprendizagem da programação, com especial destaque para a utilização das ferramentas do Dev C++ e da Inteligência Artificial como ferramenta auxiliar na medida em que constituiu um apoio auxiliar para o desenvolvimento das competências dos alunos. Esta experiência permitiu aferir os desafios enfrentados pelos alunos durante a aprendizagem, particularmente devido à falta de bases sólidas uma vez que esta turma é iniciante em programação.

A implementação de metodologias ativas, como a aprendizagem baseada em problemas e a prática sistemática através de exercícios conduzidos, permitiu um progresso significativo na compreensão dos conceitos abordados durante as aulas. Um dos aspetos desta intervenção é, sem dúvida, o impacto positivo da Inteligência Artificial nas aprendizagens. Através do apoio destas tecnologias emergentes, os alunos puderam resolver as suas dúvidas de forma autónoma, identificar erros no código e até otimizar soluções, no entanto foram críticos nas respostas fornecidas pela Inteligência Artificial o que indica que os alunos se preocupam em validar os conteúdos apresentados ao invés de simplesmente aceitar a resposta fornecida pela Inteligência Artificial, tornando-os mais confiantes e motivados a aprender. É indispensável a supervisão e a presença do professor para garantir a correta interpretação dos conceitos e do código de forma a garantir as boas práticas de ensino da programação em linguagem C.

A motivação destes alunos foi outro fator importante para o seu desempenho. Os resultados dos questionários aplicados antes e depois da intervenção indicam um aumento da confiança dos alunos para aprender programação, bem como uma maior predisposição para enfrentar desafios. A abordagem interativa e dinâmica adotada nas aulas contribuiu para um ambiente de aprendizagem mais estimulante, promovendo a participação ativa dos alunos e a colaboração entre os grupos de trabalho. Também foram evidenciadas algumas limitações e desafios nomeadamente ao nível da heterogeneidade da turma, uma vez que é composta por diferentes níveis de conhecimentos e ritmos de aprendizagem diferente, o que exigiu uma constante adaptação de estratégias pedagógicas por parte do professor estagiário. Inicialmente

alguns alunos demonstraram alguma resistência à programação, o que reforçou a necessidade de adaptar novas estratégias motivacionais capazes de captar a atenção e a motivação desses alunos.

A prática supervisionada permitiu validar a eficácia de uma abordagem equilibrada entre a teoria e a prática, aliada à utilização de ferramentas tecnológicas emergentes. Na última aula prática supervisionada, o professor aplicou uma ficha de aferição com o objetivo principal de aferir aprendizagens dos alunos. A turma atingiu uma média de 16,15 valores o que permite concluir um progresso significativo destes alunos ao longo da intervenção. Estes resultados demonstram uma consolidação dos conceitos principais abordados durante as aulas, nomeadamente na programação modular e na utilização de funções em C.

Apesar de terem sido detetadas dificuldades ao nível da interpretação e análise de código em C, a turma sofreu uma evolução notória, refletindo evolução nas aprendizagens nomeadamente ao nível da sintaxe e das estruturas lógicas da linguagem C. O professor deve estar apto e ser capaz de adaptar sempre as suas metodologias de ensino de acordo com as necessidades especificas dos alunos e da própria turma, para garantir que todos consigam desenvolver competências essenciais a nível da resolução de problemas computacionais e assim aprendam a programar. Através deste estudo é possível concluir que o ensino da linguagem de programação em C, pode e deve ser facilitado através de metodologias ativas com o foco de aumentar o envolvimento dos alunos e o sucesso académico.

Os resultados obtidos através deste estudo investigativo, contribui para uma reflexão sobre a importância de renovar o ensino da programação sugerindo que futuras investigações e que o próprio professor aprofunde conhecimentos sobre as novas tecnologias emergentes durante o processo de ensino-aprendizagem, inclusive no desenvolvimento de novas metodologias que personalizem ainda mais a experiência educativa.

9 Atividades de Escola

Neste capítulo pretende-se descrever as diferentes atividades em que o professor estagiário participou ao longo do semestre enquanto realizou a prática supervisionada na Escola Secundária de Santo André. Estas atividades têm como objetivo principal complementar o ensino-aprendizagem, facilitando a integração do professor na dinâmica escolar e no funcionamento da própria instituição de ensino. Estas experiências foram fundamentais para a formação do professor estagiário, proporcionando-lhe um contacto direto com a realidade escolar, permitindo assim uma reflexão crítica sobre o papel do docente para além da sala de aula.

9.1 Participação na observação da turma a lecionar

No dia 23/09/2024 durante as duas aulas de (50 min + 50 min) o professor estagiário pôde observar os comportamentos e atitudes dos alunos. Todos os alunos colocam os telemóveis em cima da mesa do professor antes de a aula iniciar. Esta medida foi tomada pela própria professora. Todos os alunos revelaram ter comportamento adequado à sala de aula. No decorrer da aula a professora manteve-se sempre atenta às atitudes e reações de todos os alunos. A professora cumpriu o que estava planificado. Sempre que considerou necessário a docente permitiu a utilização dos telemóveis como recurso, sempre com o acompanhamento da professora e supervisão, de forma a manter o ambiente necessário à aprendizagem.

Nas aulas do dia 16/10/2024 (50 min + 50 min), o professor estagiário observou que os alunos estiveram a resolver um exercício prático sobre a pesquisa de um trabalho científico sobre a linguagem C. Este exercício teve como objetivo promover o trabalho autónomo dos alunos com o acompanhamento e supervisão da professora durante o desenvolvimento da tarefa. Foi observado que toda a turma apresentou dificuldades nas aprendizagens, o que exigiu medidas de reforço através de explicações individualizadas.

Na aula do dia 22/10/2024 (50 min), a professora registou as presenças dos alunos em sala de aula realizando a chamada. A professora esteve a conversar com os alunos sobre a

importância da disciplina e rigor para o sucesso académico dado que detetou que alguns alunos têm vindo a demonstrar comportamentos menos assertivos.

Na aula do dia 24/10/2024, (50 min + 50 min) os alunos realizaram um trabalho sobre o desenvolvimento de algoritmia. Na 1.ª parte da aula a professora colocou questões aos alunos sobre a matéria lecionada na aula anterior, fazendo uma breve revisão da matéria dada. A professora utilizou diversas vezes o quadro para relembrar conceitos e tópicos dados nas aulas anteriores de forma que os alunos pudessem rever a matéria. Diversas vezes a professora se levantou e se dirigiu à turma, procurando sempre ensinar, estabelecendo diálogo com os alunos sem recurso à utilização de qualquer apoio audiovisual. Ao utilizar o quadro a professora constantemente a insistiu para que os alunos respondessem às suas questões de forma a incentivar e motivar os alunos. Sempre que um aluno aparentava revelar menos atenção ou interesse, a professora imediatamente intervinha para lhe chamar a atenção. Também se observou que não existem problemas de indisciplina nas aulas da professora. Os alunos estavam atentos e quando intervinham, comunicavam com a professora sobre os tópicos desenvolvidos nas aulas. Na 2.ª parte da aula, os alunos utilizaram os computadores para realizar um pequeno exercício prático sobre algoritmia, utilizando o programa `VisualG` que simula um ambiente de programação. A professora também autorizou a resolução do exercício com o auxílio do *ChatGPT*. Relativamente à avaliação das aulas através da observação, a professora Margarida Batista procedeu, no final da aula, ao preenchimento da grelha de observação sobre a avaliação dos alunos a nível da sua participação durante a aula.

Nas aulas do dia 30/10/2024 (50 min + 50 min), foi realizada uma simulação prática de mecanismos de controlo sobre algoritmia, utilizando o programa `VisualG.` A professora questionou os alunos sobre se a utilização do `VisualG` facilitou a estrutura do raciocínio lógico. Os alunos concordaram que se tornou mais fácil para a compreensão dos erros. Deve ser também referido que os alunos realizaram fichas de trabalho sobre algoritmia. Todas as fichas de trabalho foram partilhadas com os alunos através da plataforma *Microsoft Teams*.

Na aula do dia 12/11/2024 (50 min), o sumário da aula foi: entrega e correção dos testes de avaliação e introdução à linguagem C. A professora realizou a correção dos testes sobre algoritmia utilizando o programa do `VisualG`. O objetivo foi levar os alunos a aprender a

partir dos seus próprios erros. Em vez do trabalho de correção ter sido feito pela professora, os alunos corrigiram por eles próprios o teste. Nesta aula a professora fez uma pequena introdução teórica à linguagem em C começando por apresentar a breve história do C.

9.2 Participação do professor estagiário em reuniões de escola

No dia 09/10/2024 o professor estagiário participou numa reunião do grupo disciplinar 550 que contou com a presença dos professores do ensino secundário e do 2.º e 3.º ciclos. A reunião foi conduzida pelo professor Guilherme Batista, coordenador do grupo de informática e teve a duração de cerca de 1 h 30 min e foram abordados quatro pontos:

- Planificação dos Critérios de avaliação.
- Propostas para a calendarização das FCP e PAP.
- Debate sobre os protocolos/empresas dos alunos que vão para estágio.
- Debate sobre a manutenção dos equipamentos informáticos nas salas de aula.

No dia 12/02/2025 o professor estagiário participou na reunião dos Laboratórios de Educação Digital (LED) tendo como coordenadora a professora Margarida Batista, que conduziu a referida reunião que teve como objetivo principal a criação do regulamento interno da escola dos LED. Durante a reunião foram abordados diversos pontos:

- Criação de folha para a receção de equipamentos.
- Requisição, funcionamento e utilização de instalações e equipamentos.
- Criação de um formulário via email institucional para a requisição dos equipamentos.
- Organização do material/arrumação do material na sala pelos armários.
- Criação de avaliação e monitorização através de questionários para alunos e professores.
- Criação de relatórios de uso e impacto nas práticas pedagógicas.
- Criação de uma declaração de responsabilidade para as requisições de material das salas LED.
- Definição de ações e atividades a implementar com os LED.
- Criação de ações de formação para elementos da equipa saberem trabalhar com os LED.

- Criação de guiões/workshops.
- Criação de uma equipa via Teams para debates sobre os LED.

9.3 Participação no seminário sobre SmartCity

No dia 05/02/2025**,** decorreu no AESA a palestra sobre Cidades Inteligentes (*Smart Cities*)**,** ministrada pelos professores Guilherme Batista e Fábio Varanda**.** A palestra abordou a monitorização urbana através de tecnologias como **se**nsores, Internet das Coisas (*IoT*), Inteligência Artificial (IA) e *Big Data*. Foi destacado que esses recursos permitem recolher e transformar dados em métricas úteis para a gestão urbana. Um dos exemplos discutidos foi o (*SmartGrids)***,** um sistema de distribuição inteligente de energia nas cidades. A apresentação enfatizou as vantagens das cidades inteligentes, demonstrando como a integração de tecnologias digitais pode otimizar infraestruturas e melhorar a qualidade de vida urbana.

9.4 Atividade “Martinica”

O documento "Guião Martinica" é o roteiro para um encontro entre alunos e professores da Escola de Santo André (Portugal) e um grupo de 11 alunos e 3 professoras da Martinica (França), no âmbito do Projeto Erasmus+ “Conta-me a tua Europa”. O objetivo da visita é melhorar as competências linguísticas dos alunos em português e promover o intercâmbio cultural.

O guião inclui:

- Introdução: Explicação do propósito da visita e breve apresentação do grupo da Martinica.
- Perguntas para os alunos: Questões sobre as suas impressões de Portugal, diferenças culturais, experiências gastronómicas e escolares, e motivos para estudar português.
- Perguntas para os professores: Questões sobre a disciplina que lecionam, a experiência em Portugal, diferenças entre os países e escolas, e recomendações para alunos portugueses que possam estudar na Martinica.

O documento serve como uma estrutura para orientar a interação entre os participantes, promovendo um intercâmbio linguístico e cultural. Durante a realização das diversas atividades o professor estagiário participou e colaborou ativamente com os professores responsáveis pelas mesmas.

9.5 Atividade Net segura realizada com a turma da prática supervisionada

O documento "Guião do professor - Desafio Ensino Secundário" propõe uma atividade para a Semana da Net-Segura, focada no tema *Deep Learning* – Como as Máquinas Aprendem?

Objetivo da Atividade:

- Sensibilizar os alunos para a utilização responsável da internet.
- Desenvolver competências na seleção e análise de fontes credíveis.
- Demonstrar o potencial da Inteligência Artificial (IA) para pesquisas académicas.
- Ensinar as regras sobre referências e citações segundo as normas APA (7.ª edição).
- Incentivar a leitura crítica e a segurança digital.

Estrutura da Atividade:

1ª Parte – Pesquisa e Resumo

- Os alunos devem pesquisar um artigo sobre *Deep Learning* (máx. 15 páginas) de fontes credíveis (ex.: universidades).
- Fazer o download do artigo e carregá-lo numa ferramenta de IA (ChatGPT).
- Solicitar um resumo e trabalhar o conteúdo.

2ª Parte – Elaboração do Trabalho

Criar um documento Word seguindo esta estrutura:

- Capa (Nome, número, turma, escola).

- Resumo.
- Introdução.
- Desenvolvimento.
- Conclusão.
- Referências (APA 7.ª edição).
- O texto não deve ultrapassar duas páginas, excluindo capa e referências.
- Aplicar citações académicas corretamente.

Impacto Esperado:

- Desenvolver nos alunos uma visão crítica sobre a IA e sobre as pesquisas académicas.
- Promover a segurança digital e a credibilidade das fontes.
- Integrar ferramentas de IA de forma ética e responsável durante a aprendizagem.

A atividade foi realizada em parceria entre o professor estagiário e a professora cooperante da escola com os alunos onde ocorreu a prática supervisionada.

10 Portal Educativo: Recursos e Materiais

Este capítulo tem como objetivo principal apresentar um portal criado no âmbito do Mestrado via Ensino da Informática onde estão reunidos e disponibilizados todos os recursos, materiais, trabalhos e planos de aula desenvolvidos durante as aulas práticas supervisionadas. A criação deste portal procura facilitar o acesso organizado e estruturado aos conteúdos utilizados durante o estágio, promovendo a transparência e permitindo o acesso e a partilha de conhecimentos com outros docentes e colegas.

Através do *Link* do portal será possível aceder a:

- Todos os materiais pedagógicos utilizados durante a prática supervisionada.
- Atividades realizadas na escola fora do âmbito letivo.
- Todos os conteúdos dos questionários, planos de intervenção, planificações das aulas, materiais didáticos utilizados nas aulas e fichas realizadas.
- Todos os recursos educativos estão disponíveis para consulta futura, tanto por alunos como por docentes e interessados no desenvolvimento do tema deste relatório.
- A uma plataforma digital que permite fazer o download dos materiais.

Através desta ferramenta, pretende-se não só documentar todo o desenvolvimento do percurso da prática supervisionada, mas também proporcionar um instrumento útil, online e acessível a partir de qualquer browser, através de um simples clique no Link.

10.1 Link do portal educativo

- https://disciplina-de-iniciacao-a-pratica-profissional-iii---ensino-d.webnode.pt/?_gl=1*1sj7yon*_gcl_au*NjA4MjY5NjM0LjE3MzM0MTE5NDU.

10.2 Lista dos materiais disponibilizados no portal educativo

- Resultados dos Questionários.
- Calendarização do plano de Intervenção.
- Planificações aula a aula.
- Materiais das aulas.
    - Resolução dos exercícios das aulas.
- Ficha de trabalho formativa.
    - Correção da ficha.
- Ficha de aferição das aprendizagens.
    - Tabela dos resultados da ficha de aferição.
    - Correção da ficha.
- Trabalhos não publicados.
    - Trabalho Final de Psicologia.
    - Projeto de Intervenção.
- Atividades de Escola.

Referências

Agência Nacional para a Qualificação e o Ensino Profissional. (n.d.). *Programação estruturada - 769. Catálogo Nacional de Qualificações*. Recuperado a 27 de janeiro de 2025, de https://catalogo.anqep.gov.pt/ufcdDetalhe/769

Agrupamento de Escolas de Santo André. (n.d.). *Página institucional do agrupamento*. Recuperado a 27 de janeiro de 2025, de https://www.aesa.edu.pt/edu

Alves, L. M. (2019). *Ensino-aprendizagem de programação*. Instituto Politécnico de Bragança. Recuperado a 27 de janeiro de 2025, de https://bibliotecadigital.ipb.pt/handle/10198/27880

Barron, B., & Darling-Hammond, L. (2008). *Teaching for meaningful learning: A review of research on inquiry-based and cooperative learning*. Edutopia. https://www.edutopia.org/inquiry-project-learning-research

Bell, J., & Waters, S. (2018). *Doing your research project: A guide for first-time researchers*. McGraw-Hill Education.

Bell, S. (2010). Project-based learning for the 21st century: Skills for the future. *The Clearing House, 83*(2), 39–43. https://doi.org/10.1080/00098650903505415

Brito, J. (2024). *Trabalho final de psicologia*. [Manuscrito não publicado]. Faculdade de Ciências Sociais, Educação e Administração, Mestrado em Ensino de Informática.

Brito, J. A. F. (2025). *Projeto de intervenção – Plano de trabalho* [Trabalho não publicado]. Universidade Lusófona de Humanidades e Tecnologias.

Bryman, A. (2016). *Social research methods* (5th ed.). Oxford University Press.

Cardoso, S. S. P. J., & Lamas, E. P. R. (2021). Educação no processo de ensino e aprendizagem: Pais e/ou encarregados – Formas de acompanhamento. *E-Revista de Estudos Interculturais do CEI – ISCAP*, (9), 1–20. https://doi.org/10.34630/erei.v3i9.4225

CAST. (2011). *Universal design for learning (UDL) guidelines version 2.0. CAST*. Recuperado em 27 de janeiro de 2025, de https://udlguidelines.cast.org

Catálogo Nacional de Qualificações. (n.d.). *Cursos profissionais*. Catálogo Nacional de Qualificações. Recuperado em 27 de janeiro de 2025, de https://catalogo.anqep.gov.pt

Coelho, K. C. L. dos S., Schmidt, F. L. A., Theodorovski, R., Viudes, M. M., Oliveira, E. A. R. de, Oliveira, J. D. de, Santos, M. do S. dos, Reis, D. S., Souza, C. R. da S., & Campos, D. R. de. (2024). *Paulo Freire e o diálogo como ferramenta de formação de professores*. *IOSR Journal of Humanities and Social Science, 29*(2), 31–37. https://www.researchgate.net/publication/378035295_Paulo_Freire_E_O_Dialogo_Como_Ferramenta_De_Formacao_De_Professores#fullTextFileContent

Creswell, J. W., & Clark, V. L. P. (2014). *Research design: Qualitative, quantitative, and mixed methods approach* (4.ª ed.). Sage Publications.

Damas, L. (2019). *Linguagem C* (24.ª ed.). Edições FCA.

Holmes, W., Bialik, M., & Fadel, C. (2019). *Artificial intelligence in education: Promises and implications for teaching and learning.* Center for Curriculum Redesign.

Kernighan, B. W., & Ritchie, D. M. (2018). *The C programming language (ANSI C)* (3.ª ed.). Prentice Hall.

Kuhl, P. K., Lim, S.-S., Guerriero, S., & van Damme, D. (2019). *Developing minds in the digital age: Towards a science of learning for 21st century education*. OECD Publishing. https://doi.org/10.1787/562a8659-en

Likert, R. (1932). *A technique for the measurement of attitudes* (Vol. 22, No. 140). Archives of Psychology.

Lopes, N., & Bastos, A. M. (2017). *A prática de ensino supervisionada na formação inicial de professores do 1.º CEB: Dinâmicas na UTAD*. Revista Practicum, 2(2), 69–83. https://revistapracticum.com/index.php/iop/article/view/38.

Lourenço, L. P. R. (2008). *Envolvimento dos encarregados de educação na escola: Conceções e práticas* [Dissertação de Mestrado, Universidade de Lisboa].

Luckin, R., Holmes, W., Griffiths, M., & Forcier, L. B. (2016*). Intelligence unleashed: An argument for AI in education*. Pearson Education.

Mangas, C., & Sousa, J. (2020). *Práticas de ensino-aprendizagem com base em cenários reais na formação superior em inclusão*. INNODOCT/20 - I*nternational Conference on Innovation*, Documentation and Education. Editorial Universitat Politècnica de València. https://doi.org/10.4995/INN2020.2020.11838

OpenAI. (2025). ChatGPT (versão do modelo GPT-4) [Modelo de linguagem de IA]. Recuperado a 27 de janeiro de 2025, de https://openai.com

Robins, A., Rountree, J., & Rountree, N. (2003). Learning and teaching programming: A review and discussion. *Computer Science Education*, 13(2), 137–172. https://doi.org/10.1076/csed.13.2.137.14200

Ribeiro, F. (2011). *Motivação e aprendizagem em contexto escolar*. Revista PROFFORMA (03). Escola Secundária de São Lourenço. Disponível em https://cefopna.edu.pt/revista/revista_03/es_05_03_FR.htm

Ryan, R. M., & Deci, E. L. (2000). Self-determination theory and the facilitation of intrinsic motivation, social development, and well-being. *American Psychologist, 55*(1), 68–78. https://doi.org/10.1037/0003-066X.55.1.68

Sentance, S., Barendsen, E., Howard, N. R., & Schulte, C. (2023). *Computer science education: Perspectives on teaching and learning in school*. Bloomsbury Academic.

Teixeira, M., & Reis, M. F. (2012). A organização do espaço em sala de aula e as suas implicações na aprendizagem cooperativa. *Revista Meta Avaliação, 4*(2), 162–187.

Terra, A. L. (2022). *Aprendizagem baseada em projetos: Uma experiência pedagógica na área da Ciência da Informação.* Páginas a&b, (17). https://hdl.handle.net/10316/101406

UNESCO (2023). *Global Education Monitoring Report Team. Relatório de monitoramento global da educação: Resumo, 2023: A tecnologia na educação: uma ferramenta a serviço de quem?* (ED/GEMR/MRT/2023/S1). https://doi.org/10.54676/CUYC7902

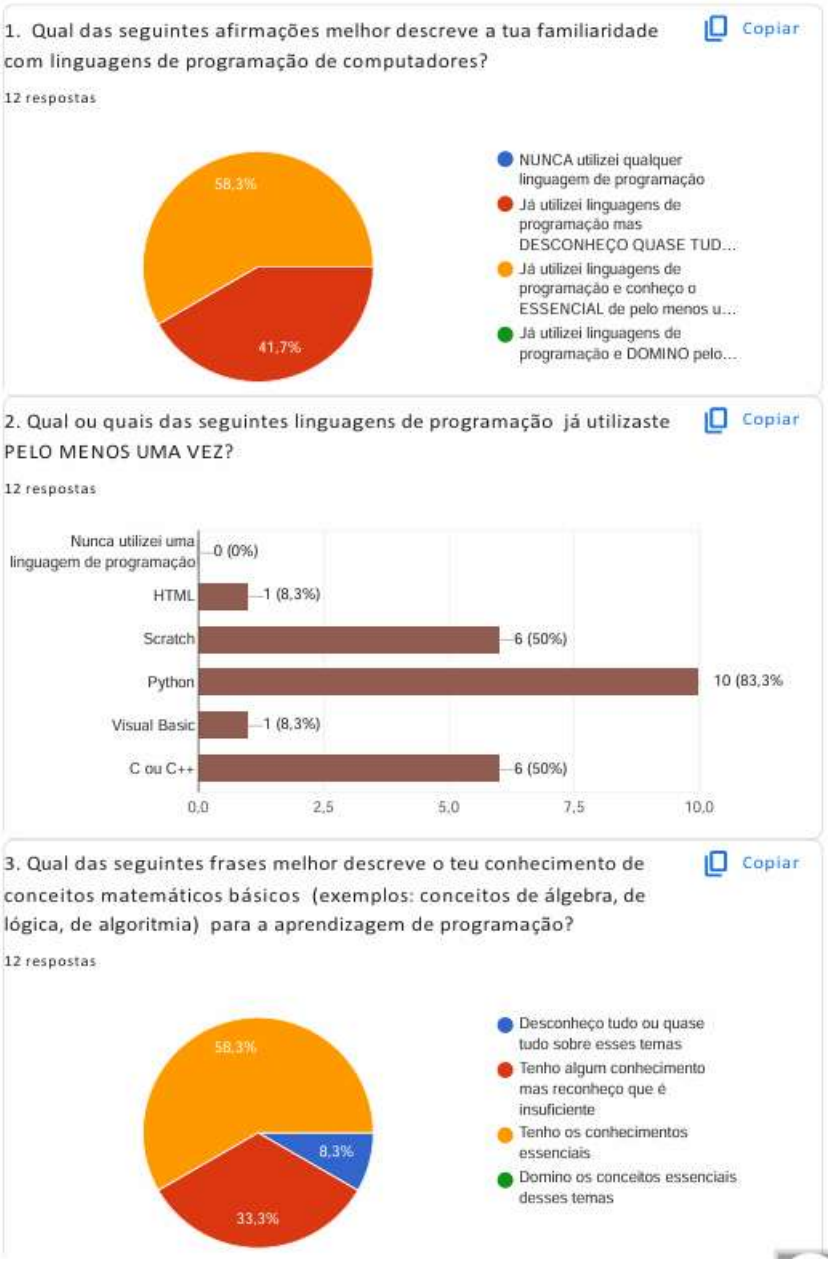
Universidade do Porto. (n.d.). *Método interrogativo*. Recuperado a 21 de fevereiro de 2025, de https://www.up.pt/portal/pt/inovacao-educativa/ensino-e-aprendizagem/abordagens/m%C3%A9todo-interrogativo

Wing, J. M. (2006). *Computational thinking*. Communications of the ACM, 49(3), 33–35. https://doi.org/10.1145/1118178.1118215

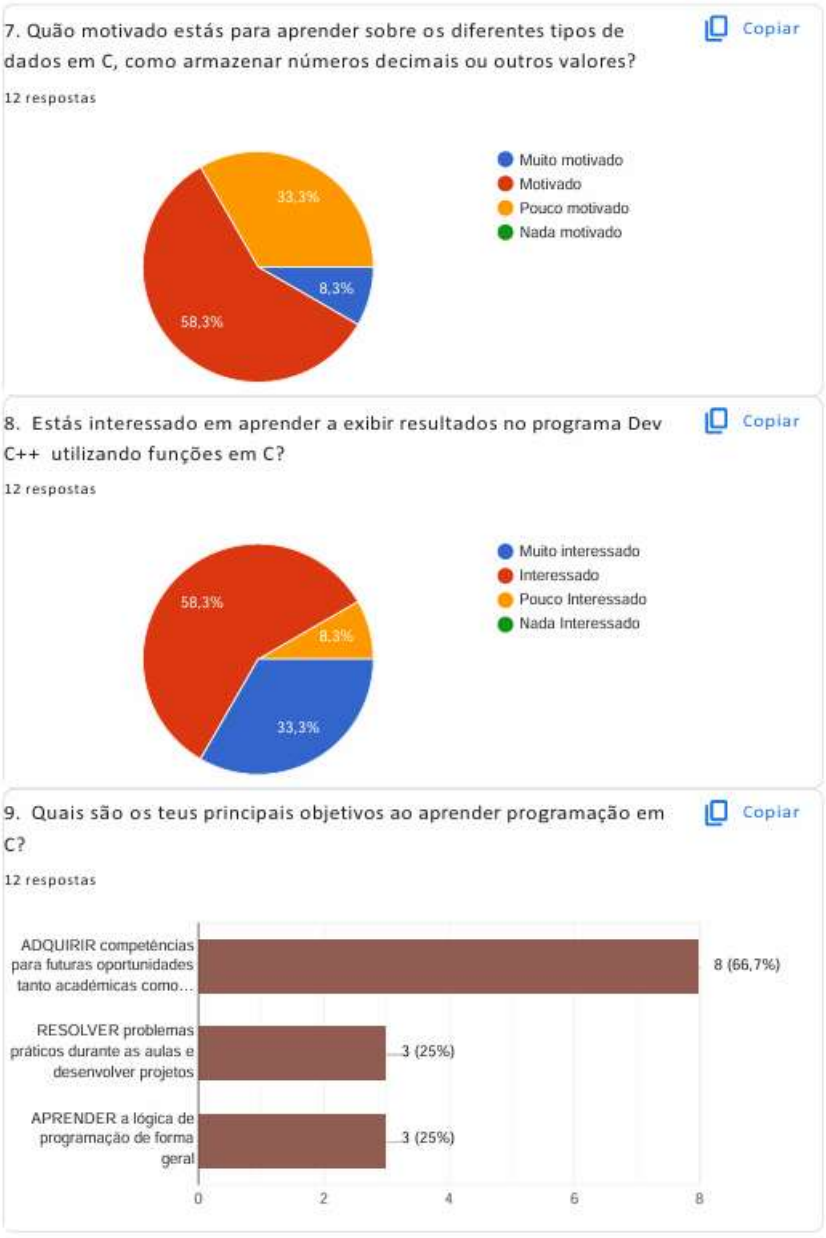
Yilmaz, R., & Karaoglan Yilmaz, F. G. (2023). The effect of generative artificial intelligence (AI)-based tool use on students' computational thinking skills, programming self-efficacy and motivation. *Computers and Education: Artificial Intelligence, 4*, 100147. https://doi.org/10.1016/j.caeai.2023.100147

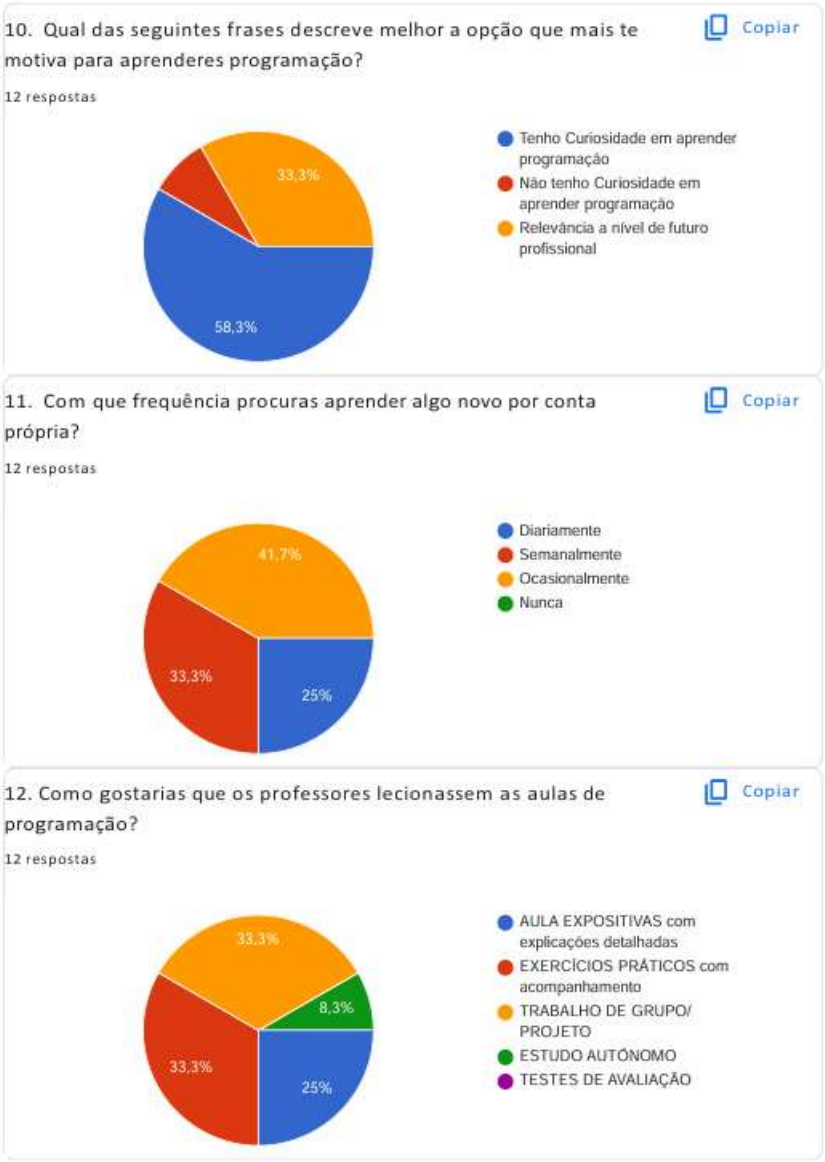
Apêndices

Questionários Diagnóstico


1. Qual das seguintes afirmações melhor descreve a tua familiaridade com linguagens de programação de computadores?
Copiar
12 respostas
58,3%
41,7%
NUNCA utilizei qualquer linguagem de programação
Já utilizei linguagens de programação mas DESCONHEÇO QUASE TUD...
Já utilizei linguagens de programação e conheço o ESSENCIAL de pelo menos u...
Já utilizei linguagens de programação e DOMINO pelo...
2. Qual ou quais das seguintes linguagens de programação já utilizaste PELO MENOS UMA VEZ?
Copiar
12 respostas
Nunca utilizei uma linguagem de programação
0 (0%)
HTML
1 (8,3%)
Scratch
6 (50%)
Python
10 (83,3%
Visual Basic
1 (8,3%)
C ou C++
6 (50%)
0,0
2,5
5,0
7,5
10,0
3. Qual das seguintes frases melhor descreve o teu conhecimento de conceitos matemáticos básicos (exemplos: conceitos de álgebra, de lógica, de algoritmia) para a aprendizagem de programação?
Copiar
12 respostas
58,3%
8,3%
33,3%
Desconheço tudo ou quase tudo sobre esses temas
Tenho algum conhecimento mas reconheço que é insuficiente
Tenho os conhecimentos essenciais
Domino os conceitos essenciais desses temas


4. Consideras que as Plataformas de Inteligência Artificial podem ser uma mais valia na aprendizagem da programação em C?
Copiar
12 respostas
Sim
Não
100%
5. Gostarias que o professor utilizasse alguma Plataforma de Inteligência Artificial durante a prática de ensino?
Copiar
12 respostas
Sim
Não
16,7%
83,3%
6. Qual dos seguintes aspetos da programação em C despertam mais o teu interesse e a tua motivação?
Copiar
12 respostas
16,7%
16,7%
41,7%
25%
Aprender a estrutura básica de um programa, como a função main().
Resolver problemas práticos utilizando algoritmos.
Desenvolver competências para futuras oportunidades académicas ou profissionais.
Explorar a lógica de programação de forma geral.

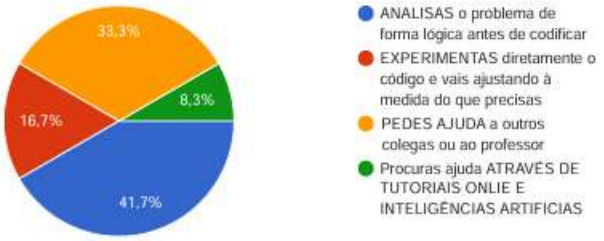

7. Quão motivado estás para aprender sobre os diferentes tipos de dados em C, como armazenar números decimais ou outros valores?
Copiar
12 respostas
33,3%
8,3%
58,3%
Muito motivado
Motivado
Pouco motivado
Nada motivado
8. Estás interessado em aprender a exibir resultados no programa Dev C++ utilizando funções em C?
Copiar
12 respostas
58,3%
8,3%
33,3%
Muito interessado
Interessado
Pouco Interessado
Nada Interessado
9. Quais são os teus principais objetivos ao aprender programação em C?
Copiar
12 respostas
ADQUIRIR competências para futuras oportunidades tanto académicas como...
8 (66,7%)
RESOLVER problemas práticos durante as aulas e desenvolver projetos
3 (25%)
APRENDER a lógica de programação de forma geral
3 (25%)
0
2
4
6
8



Aprendizagem e Preferências Pessoais

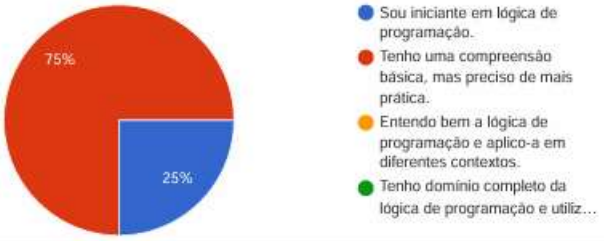
13. Quando te deparas com um problema em programação, como o resolves? Copiar

12 respostas



14. Qual o teu nível de familiaridade com a lógica de programação em geral? Copiar

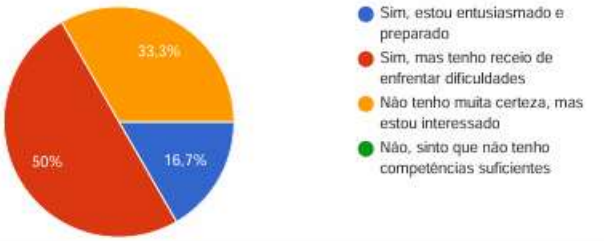
12 respostas



Expetativas relativamente à disciplina

15. Sentes-te preparado para aprender programação em C? Copiar

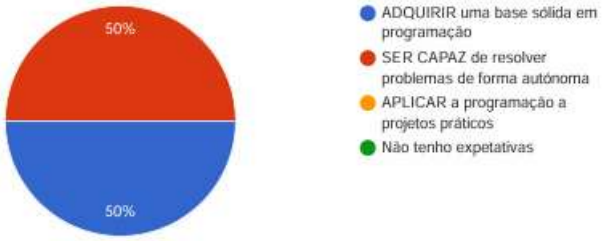
12 respostas



16. Quais são as tuas expectativas em relação às aulas de programação em C?

Copiar

12 respostas



17. Há algo específico que gostarias de aprender ou desenvolver durante as aulas de programação em C?

9 respostas

Não

Não á nada especifico

Quero desenvolver as minhas competências, e realizar trabalhos de grupo e projetos para desenvolver as minhas competências

sim pode ser

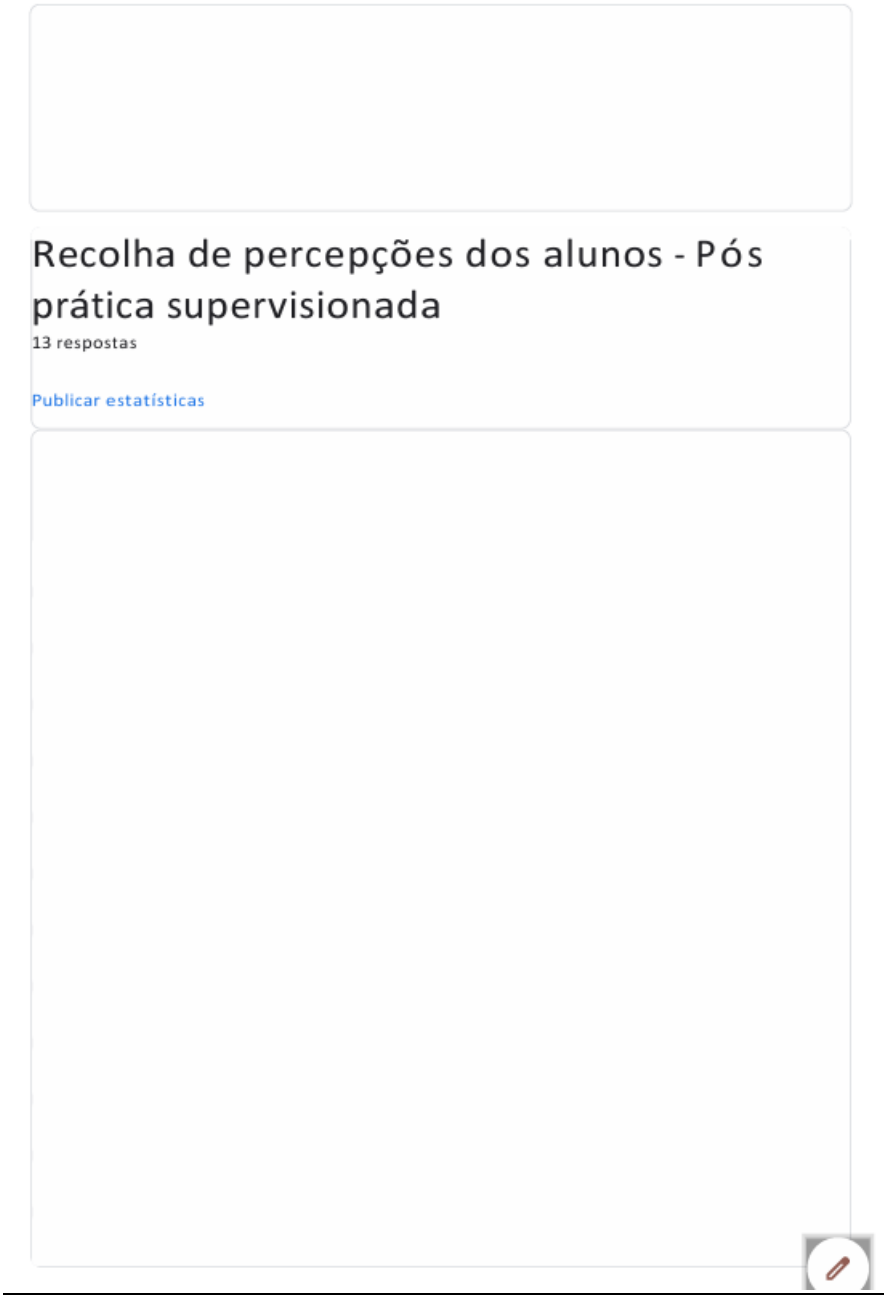
gostaria de ter aulas com o professor/a a mostrar powerpoints e a explicar os conteúdos no projetor.

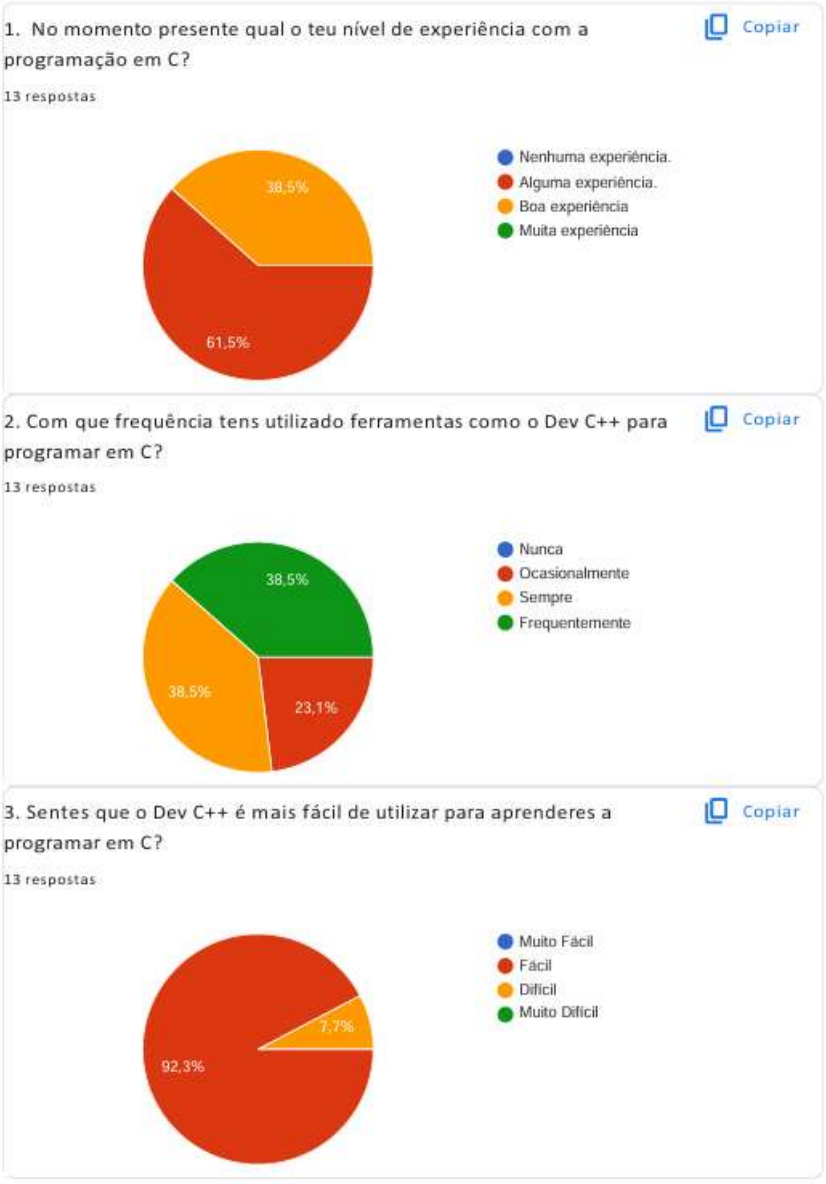
Funções e formas de depuração (deixar o código mais limpo e curto).

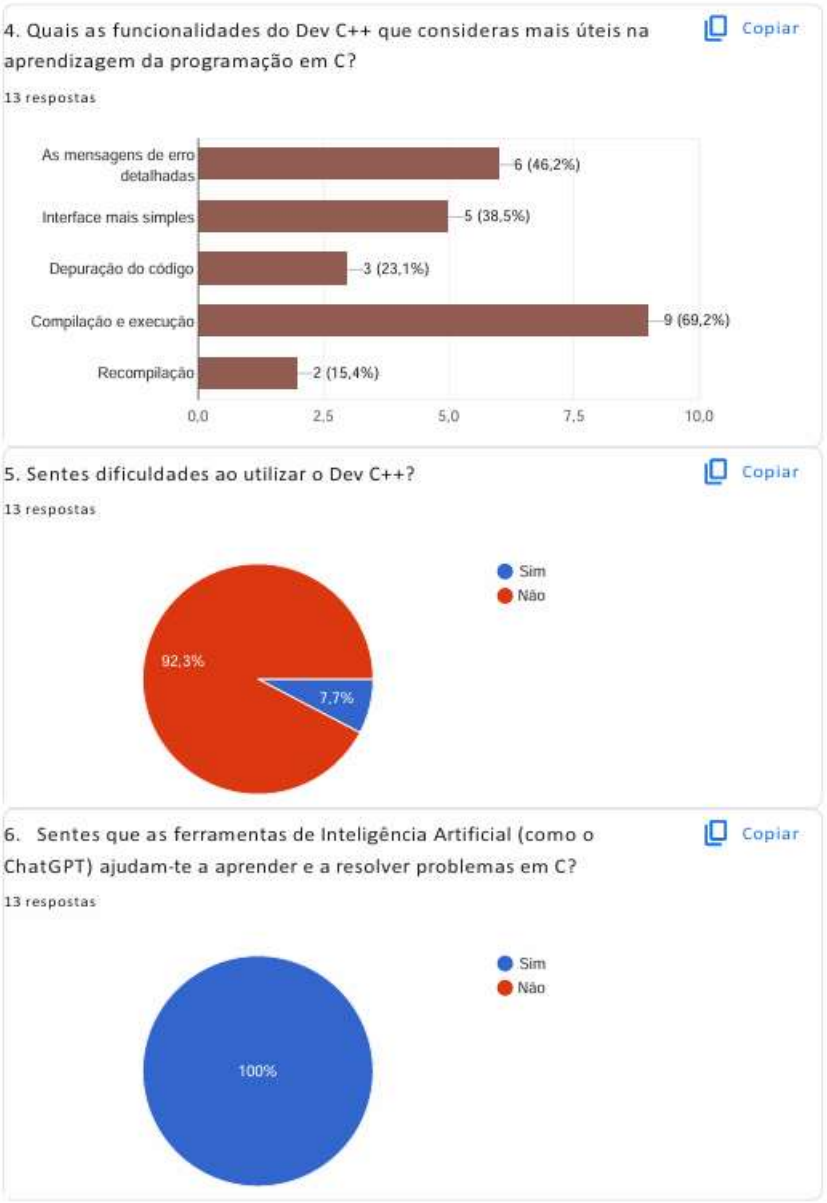
não

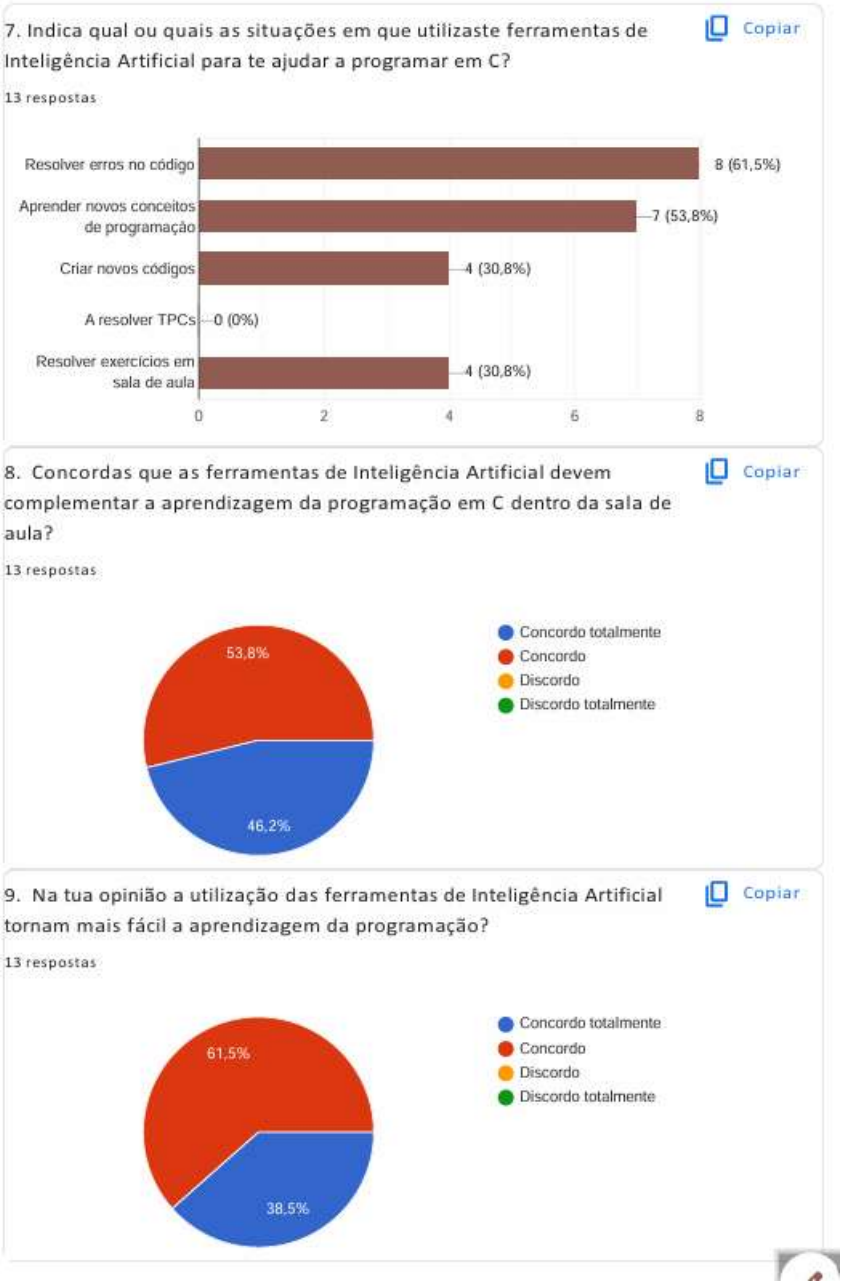
Programas que ajudam no desenvolvimento do conhecimento.

Fim do Questionário.

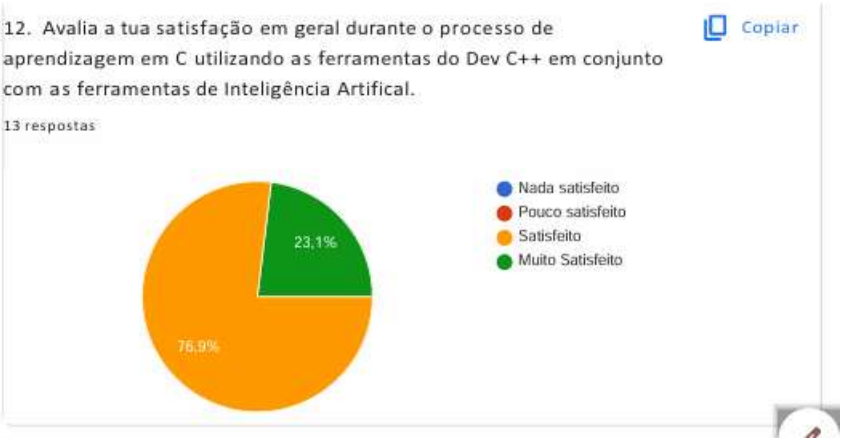

Recolha de percepções dos alunos - Pós prática supervisionada
13 respostas
Publicar estatísticas

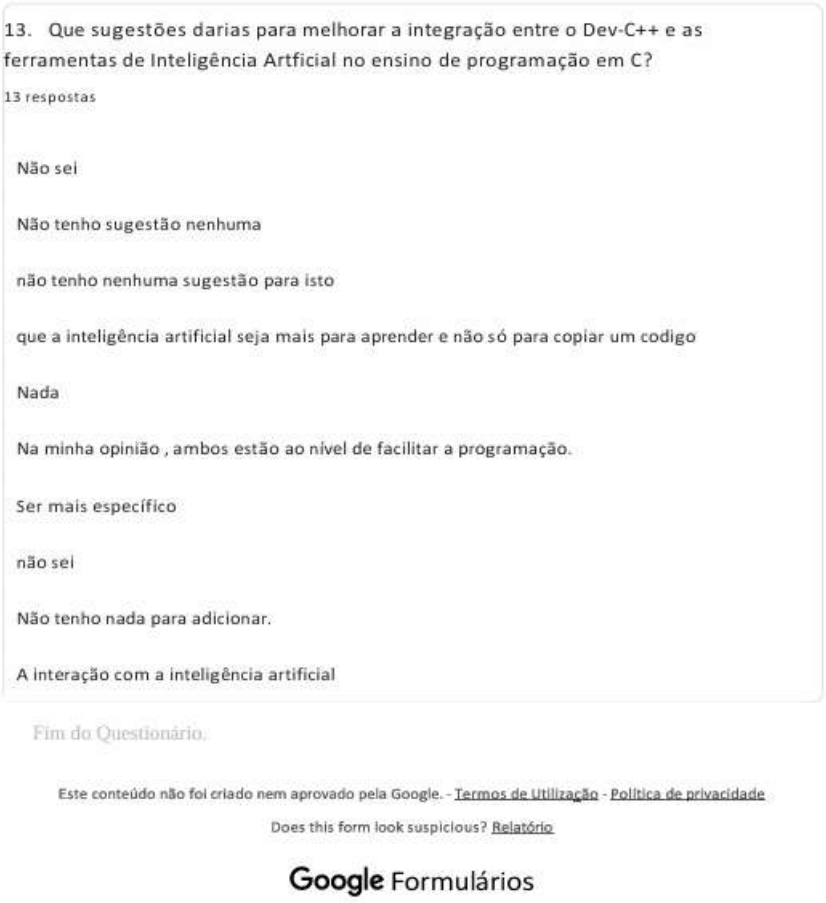

1. No momento presente qual o teu nível de experiência com a programação em C?
Copiar
13 respostas
38,5%
61,5%
Nenhuma experiência.
Alguma experiência.
Boa experiência
Muita experiência
2. Com que frequência tens utilizado ferramentas como o Dev C++ para programar em C?
Copiar
13 respostas
38,5%
38,5%
23,1%
Nunca
Ocasionalmente
Sempre
Frequentemente
3. Sentes que o Dev C++ é mais fácil de utilizar para aprenderes a programar em C?
Copiar
13 respostas
7,7%
92,3%
Muito Fácil
Fácil
Difícil
Muito Difícil


4. Quais as funcionalidades do Dev C++ que consideras mais úteis na aprendizagem da programação em C?
Copiar
13 respostas
As mensagens de erro detalhadas
6 (46,2%)
Interface mais simples
5 (38,5%)
Depuração do código
3 (23,1%)
Compilação e execução
9 (69,2%)
Recompilação
2 (15,4%)
0,0
2,5
5,0
7,5
10,0
5. Sentes dificuldades ao utilizar o Dev C++?
Copiar
13 respostas
Sim
Não
92,3%
7,7%
6. Sentes que as ferramentas de Inteligência Artificial (como o ChatGPT) ajudam-te a aprender e a resolver problemas em C?
Copiar
13 respostas
Sim
Não
100%


7. Indica qual ou quais as situações em que utilizaste ferramentas de Inteligência Artificial para te ajudar a programar em C?
Copiar
13 respostas
Resolver erros no código
8 (61,5%)
Aprender novos conceitos de programação
7 (53,8%)
Criar novos códigos
4 (30,8%)
A resolver TPCs
0 (0%)
Resolver exercícios em sala de aula
4 (30,8%)
0
2
4
6
8
8. Concordas que as ferramentas de Inteligência Artificial devem complementar a aprendizagem da programação em C dentro da sala de aula?
Copiar
13 respostas
53,8%
46,2%
Concordo totalmente
Concordo
Discordo
Discordo totalmente
9. Na tua opinião a utilização das ferramentas de Inteligência Artificial tornam mais fácil a aprendizagem da programação?
Copiar
13 respostas
61,5%
38,5%
Concordo totalmente
Concordo
Discordo
Discordo totalmente

Avaliação e Satisfação





13. Que sugestões darias para melhorar a integração entre o Dev-C++ e as ferramentas de Inteligência Artficial no ensino de programação em C?

13 respostas

Não sei

Não tenho sugestão nenhuma

não tenho nenhuma sugestão para isto

que a inteligência artificial seja mais para aprender e não só para copiar um codigo

Nada

Na minha opinião , ambos estão ao nivel de facilitar a programação.

Ser mais específico

não sei

Não tenho nada para adicionar.

A interação com a inteligência artificial

Fim do Questionário.

Planificações das aulas supervisionadas

AGRUPAMENTO DE ESCOLAS DE SANTO ANDRÉ - 120340 -

Prática Supervisionada

2024/2025

Identificação do Estagiário

Nome: João Alberto Fontinha de Brito

Agrupamento: de Escolas de Santo André

Departamento: Matemática e Informática **Grupo de Recrutamento:** 550

Disciplina: Programação **Ano:**1º **Turma:** L

Data: xx/xx/2025

João Brito Plano de sessão 1 Página 1 de 7

AGRUPAMENTO DE ESCOLAS DE SANTO ANDRÉ - 120340 -

Contextualização (escola, turma e unidade didática)

A turma 1.ºL está inserida na escola Secundária de Santo André, a qual pertence ao Agrupamento de escolas de Santo André, no Barreiro. A turma frequenta o Curso Profissional de Técnico de Sistemas - Informática. A turma é composta por 13 alunos, sendo que doze alunos são do sexo masculino e um do sexo feminino. A média de idades da turma ronda os 15 anos de idade. Existem dois alunos estrangeiros que beneficiam do apoio à língua não materna. Existe um aluno com RTP (Relatório Técnico Pedagógico). Um outro aluno é paquistanês e não fala a língua portuguesa, e é um aluno de nível de língua portuguesa de iniciação, A1 e A2. Beneficia de adaptações curriculares não significativas e também beneficia de adequações ao processo de avaliação assim como a antecipação e reforço das aprendizagens. É importante também referir que esta turma metade são pertencentes aos cursos Técnico profissionais de Informática – Sistemas e a outra metade pertencente ao curso de Gestão e programação de Sistemas Informáticos. Perante algumas observações que realizei durante as aulas da professora Margarida Batista, pude observar que a turma conta alguns alunos que apresentam dificuldades de aprendizagem.

João Brito Plano de sessão 1 Página 2 de 7

AGRUPAMENTO DE ESCOLAS DE SANTO ANDRÉ - 120340 -

Departamento de Matemática e Informática | Grupo de Recrutamento: 550 – Informática

Ano Letivo: 2024/2025

Plano de Aula nº 1			
Ano: 1.º **Turma:** L	**Disciplina**: Programação		**Aula nº**: 121
Data: xx/01/2025	**Hora:** 11:50 – 12:40	**Duração:** 50 min	
PROFESSOR:	João Brito		
UFCD 0784	Funções e Estruturas		
SUMÁRIO	• Introdução ao conceito de Programação Modular e suas principais vantagens. • Ciclo de desenvolvimento de um programa em C. • Estrutura de um programa em C • A função main(). • Discussão sobre a importância da: função main() e módulos para o desenvolvimento de software. • Aprendizagem através de exemplos práticos.		
CONTEÚDOS	• Rever os conceitos de • Programação Modular • Conceito • Vantagens • Ciclo de desenvolvimento de um programa em C • Estrutura de um programa em C • Exemplos práticos		

João Brito Plano de sessão 1 Página 3 de 7

AGRUPAMENTO DE ESCOLAS DE SANTO ANDRÉ - 120340 -

OBJECTIVOS GERAIS	• Rever os conceitos fundamentais de programação modular. • Analisar o ciclo de desenvolvimento de um programa em C. • Estruturar programas de forma organizada. • Utilizar parâmetros em subprogramas. • Exemplificar a estrutura de programas em C. • Promover a prática de programação estruturada.					
Objetivos Específicos	**Conteúdos**	**Metodologias**	**Atividades**	**Recursos**	**Duração**	**Avaliação**

AGRUPAMENTO DE ESCOLAS DE SANTO ANDRÉ - 120340 -

Introdução	Reconhecer os objetivos específicos para o módulo e sessão Refletir acerca dos conhecimentos que têm acerca dos conteúdos em análise	Apresentação dos conteúdos do módulo ao grupo Referência aos critérios de avaliação Apresentação dos objetivos específicos para a sessão	Método Expositivo Exposição Dialogada Método Interrogativo – Colocação de Questões	Plenário: Diálogo Apresentação de diapositivos	Computador Projetor de vídeo Tela de projeção PowerPoint Papel e caneta	10 minutos	Observação não focada (postura e atenção)

AGRUPAMENTO DE ESCOLAS DE SANTO ANDRÉ - 120340 -

Desenvolvimento	Consolidar conhecimentos ao nível da programação modular, conceitos e vantagens. Análise do Ciclo de desenvolvimento de um programa em C. Análise da estrutura de um programa em C. Apresentação de exemplos práticos.	Programação modular, conceitos principais, vantagens, a estrutura de um programa em C, Ciclo de desenvolvimento de um programa, a Função main(), exemplo de um programa modular em C.	Método Expositivo – Exposição Dialogada Método Interrogativo – Colocação de Questões Método Afirmativo Demonstrativo	Plenário: Diálogo Apresentação de diapositivos tendo por base exemplos de código em C.	Computador Projetor de vídeo Tela de projeção PowerPoint Quadro branco Papel e caneta	35 minutos	Observação não focada (postura e atenção) Avaliação continua formativa Avaliação continua
Conclusão	Aferir competências adquiridas durante a sessão Estabelecer a ponte com a sessão seguinte	Conteúdos desenvolvidos ao longo da sessão	Método Expositivo –Exposição Dialogada Método Interrogativo –	Plenário: Diálogo Sumula da sessão	Computador Projetor de vídeo Tela de projeção	5 minutos	Observação não focada (postura e atenção) Avaliação contínua

AGRUPAMENTO DE ESCOLAS DE SANTO ANDRÉ - 120340 -

			Colocação de Questões		PowerPoint Papel e caneta		

Assinatura da docente: _____

AGRUPAMENTO DE ESCOLAS DE SANTO ANDRÉ - 120340 -

Prática Supervisionada

2024/2025

Identificação do Estagiário			
Nome: João Alberto Fontinha de Brito			
Agrupamento: de Escolas de Santo André			
Departamento: Matemática e Informática			**Grupo de Recrutamento:** 550
Disciplina: Programação	**Ano:**1º	**Turma:** L	
Data: xx/xx/2025			

AGRUPAMENTO DE ESCOLAS DE SANTO ANDRÉ - 120340 -

Contextualização (escola, turma e unidade didática)

A turma 1.ºL está inserida na escola Secundária de Santo André, a qual pertence ao Agrupamento de escolas de Santo André, no Barreiro. A turma frequenta o Curso Profissional de Técnico de Sistemas - Informática. A turma é composta por 13 alunos, sendo que doze alunos são do sexo masculino e um do sexo feminino. A média de idades da turma ronda os 15 anos de idade. Existem dois alunos estrangeiros que beneficiam do apoio à língua não materna. Existe um aluno com RTP (Relatório Técnico Pedagógico). Um outro aluno é paquistanês e não fala a língua portuguesa, e é um aluno de nível de língua portuguesa de iniciação, A1 e A2. Beneficia de adaptações curriculares não significativas e também beneficia de adequações ao processo de avaliação assim como a antecipação e reforço das aprendizagens. É importante também referir que esta turma metade são pertencentes aos cursos Técnico profissionais de Informática – Sistemas e a outra metade pertencente ao curso de Gestão e programação de Sistemas Informáticos. Perante algumas observações que realizei durante as aulas da professora Margarida Batista, pude observar que a turma conta alguns alunos que apresentam dificuldades de aprendizagem.

AGRUPAMENTO DE ESCOLAS DE SANTO ANDRÉ - 120340 -

Departamento de Matemática e Informática	Grupo de Recrutamento: 550 – Informática Ano Letivo: 2024/2025

Plano de Aula nº 1			
Ano: 1.º **Turma:** L	**Disciplina**: Programação		**Aula nº**: 122 e 123
Data: xx/01/2025	**Hora:** 8:55 – 10:50	**Duração:** 50 min + 50 min	
PROFESSOR:	João Brito		
UFCD 0784	Funções e Estruturas		
SUMÁRIO	• Conceito de função e diferenças entre subprogramas e funções. • Tipos de funções: • Com retorno de valor. • Do tipo void (sem retorno de valor). • Nomenclatura e características das funções. • Parâmetros, argumentos e passagem de valores. • Resolução de exercícios práticos conduzidos.		
CONTEÚDOS	• Conceito de Função • Nomenclatura de uma Função • Estrutura de uma Função • A instrução Return • Função Void • Argumentos de uma Função • Exemplos práticos		

AGRUPAMENTO DE ESCOLAS DE SANTO ANDRÉ - 120340 -

OBJECTIVOS GERAIS	• Compreender o Conceito de uma Função. • Aprender a Nomenclatura de uma Função. • Analisar a Estrutura de uma Função. • Compreender o funcionamento da Instrução return. • Explorar a Função void. • Trabalhar com Argumentos de uma Função. • Aplicar os Conceitos Através de Exemplos Práticos.					
Objetivos Específicos	**Conteúdos**	**Metodologias**	**Atividades**	**Recursos**	**Duração**	**Avaliação**

AGRUPAMENTO DE ESCOLAS DE SANTO ANDRÉ - 120340 -

Introdução	Reconhecer os objetivos específicos para o módulo e sessão Refletir acerca dos conhecimentos que têm acerca dos conteúdos em análise	Apresentação dos conteúdos do módulo ao grupo Apresentação do sumário da aula aos alunos.	Método Expositivo Exposição Dialogada	Plenário: Diálogo Apresentação de diapositivos	Computador Projetor de vídeo Tela de projeção PowerPoint Papel e caneta Utilização do ambiente de programação DEV C++ Papel e caneta	10 minutos	Observação não focada (postura e atenção)

AGRUPAMENTO DE ESCOLAS DE SANTO ANDRÉ - 120340 -

Desenvolvimento	Compreender a importância das funções na programação em C como ferramentas de modularidade e organização do código. Identificar os objetivos de cada tópico abordado na sessão, relacionando-os com a prática de programação. Relacionar conhecimentos anteriores com os tópicos da sessão para construir uma base sólida de aprendizagem.	. Introduzir o conceito de funções e sua importância na programação. Ensinar a estrutura básica de uma função e as regras para sua nomenclatura. Demonstrar a utilização da instrução return para o retorno de valores. Explicar e diferenciar o uso de funções void. Aprender a utilizar argumentos Aplicar os conceitos através de exemplos práticos	Método Expositivo – Exposição Dialogada Método Interrogativo – Colocação de Questões Método Afirmativo Demonstrativo através de exercícios conduzidos.	Plenário: Diálogo Apresentação de diapositivos tendo por base exemplos de código em C.	Computador Projetor de vídeo Tela de projeção PowerPoint Quadro branco Papel e caneta Utilização do programa Dev C++- Papel e caneta	80 minutos	Observação não focada (postura e atenção) Avaliação continua formativa Avaliação continua

AGRUPAMENTO DE ESCOLAS DE SANTO ANDRÉ - 120340 -

Conclusão	Aferir competências adquiridas durante a sessão Estabelecer a ponte com a sessão seguinte	Conteúdos desenvolvidos ao longo da sessão	Método Expositivo –Exposição Dialogada Método Interrogativo – Colocação de Questões	Plenário: Diálogo Sumula da sessão	Computador Projetor de vídeo Tela de projeção PowerPoint Papel e caneta	10 minutos	Observação não focada (postura e atenção) Avaliação contínua

Assinatura da docente: _____

João Brito **Plano de sessão 1** **Página 7 de 7**

AGRUPAMENTO DE ESCOLAS DE SANTO ANDRÉ - 120340 -

Prática Supervisionada

2024/2025

Identificação do Estagiário			
Nome: João Alberto Fontinha de Brito			
Agrupamento: de Escolas de Santo André			
Departamento: Matemática e Informática			**Grupo de Recrutamento:** 550
Disciplina: Programação	**Ano:**1º	**Turma:** L	
Data: xx/xx/2025			

AGRUPAMENTO DE ESCOLAS DE SANTO ANDRÉ - 120340 -

Contextualização (escola, turma e unidade didática)

A turma 1.ºL está inserida na escola Secundária de Santo André, a qual pertence ao Agrupamento de escolas de Santo André, no Barreiro. A turma frequenta o Curso Profissional de Técnico de Sistemas - Informática. A turma é composta por 13 alunos, sendo que doze alunos são do sexo masculino e um do sexo feminino. A média de idades da turma ronda os 15 anos de idade. Existem dois alunos estrangeiros que beneficiam do apoio à língua não materna. Existe um aluno com RTP (Relatório Técnico Pedagógico). Um outro aluno é paquistanês e não fala a língua portuguesa, e é um aluno de nível de língua portuguesa de iniciação, A1 e A2. Beneficia de adaptações curriculares não significativas e também beneficia de adequações ao processo de avaliação assim como a antecipação e reforço das aprendizagens. É importante também referir que esta turma metade são pertencentes aos cursos Técnico profissionais de Informática – Sistemas e a outra metade pertencente ao curso de Gestão e programação de Sistemas Informáticos. Perante algumas observações que realizei durante as aulas da professora Margarida Batista, pude observar que a turma conta alguns alunos que apresentam dificuldades de aprendizagem.

AGRUPAMENTO DE ESCOLAS DE SANTO ANDRÉ - 120340 -

Departamento de Matemática e Informática | Grupo de Recrutamento: 550 – Informática

Ano Letivo: 2024/2025

Plano de Aula nº 1			
Ano: 1.º **Turma:** L	**Disciplina**: Programação		**Aula nº**: 124 e 125
Data: xx/01/2025	**Hora:** 11:50 – 13:35	**Duração:** 50 min + 50 min	
PROFESSOR:	João Brito		
UFCD 0784	Funções e Estruturas		
SUMÁRIO	• Desenvolvimento de um programa em C com o objetivo de criar uma calculadora em C utilizando as 4 principais operações básicas: A Soma, Subtração, Multiplicação e Divisão. • Criação de funções individuais para cada operação. • Implementação de um menu interativo com o objetivo de utilizar as estruturas de controlo (switch-case).		
CONTEÚDOS	• Exploração das Funções com Recurso à Inteligência Artificial • Utilizar ferramentas de IA para criar uma simulação de uma calculadora em C. • 2 - Desconstrução da Solução • Compreender os elementos do programa selecionado pelos alunos.		
OBJECTIVOS GERAIS	• Demonstrar como as ferramentas de IA podem ser utilizadas para gerar código e simular soluções práticas em C. • Incentivar a utilização da Inteligência Artificial como suporte no desenvolvimento de programas modulares. • Desenvolver uma aplicação em C utilizando funções para realizar operações matemáticas básicas.		

AGRUPAMENTO DE ESCOLAS DE SANTO ANDRÉ - 120340 -

	• Estruturar o programa de forma modular, implementando funções como a soma, subtração, multiplicação e divisão. • Identificar e compreender os elementos fundamentais do programa criado. • Analisar a estrutura de um programa, incluindo a utilização de funções, argumentos, retorno de valores e organização do código. • Estimular os alunos a refletirem sobre a eficácia da solução proposta. • Desenvolver habilidades de análise e depuração de código, reconhecendo boas práticas de programação. • Incentivar o pensamento crítico sobre o impacto da IA no processo de aprendizagem e no desenvolvimento de software. • Mostrar como a integração de IA pode otimizar o processo de criação, análise e melhoria de programas.					
Objetivos Específicos	**Conteúdos**	**Metodologias**	**Atividades**	**Recursos**	**Duração**	**Avaliação**

AGRUPAMENTO DE ESCOLAS DE SANTO ANDRÉ - 120340 -

Introdução	Reconhecer os objetivos específicos para o módulo e sessão Refletir acerca dos conhecimentos adquiridos nas aulas anteriores.	Apresentação dos conteúdos do módulo ao grupo Apresentação do sumário da aula aos alunos.	Método Expositivo Exposição Dialogada	Plenário: Diálogo Apresentação de diapositivos	Computador Projetor de vídeo Tela de projeção PowerPoint Papel e caneta Utilização do ambiente de programação DEV C++ Papel e caneta	10 minutos	Observação não focada (postura e atenção)

AGRUPAMENTO DE ESCOLAS DE SANTO ANDRÉ - 120340 -

Desenvolvimento	Demonstrar como ferramentas de IA podem ser utilizadas para criar código eficiente e funcional em C, aplicando os conceitos sobre as funções. Promover a compreensão crítica do código criado, identificando os seus elementos principais e analisando sempre a sua estrutura. Analisar detalhadamente os elementos do programa, incluindo o cabeçalho das funções, parâmetros, retorno e lógica interna.	Estrutura geral de um programa em C. Modularização e organização do código. Definição e implementação de funções individuais para: • Soma. • Subtração. • Multiplicação. • Divisão. Utilização de parâmetros e retorno de valores. Implementação do menu interativo com switch-case. Utilização de ciclos repetitivos para permitir	Método Expositivo – Exposição Dialogada Método Interrogativo – Colocação de Questões Método Afirmativo Demonstrativo através de exercícios conduzidos. Desconstrução da solução do exercício apresentado aos alunos.	Plenário: Diálogo Apresentação de diapositivos tendo por base exemplos de código em C.	Computador Projetor de vídeo Tela de projeção PowerPoint Quadro branco Papel e caneta Utilização do programa Dev C++. Papel e caneta	80 minutos	Observação não focada (postura e atenção) Avaliação continua formativa Avaliação continua

	Reconhecer a interação entre a função principal main() e as funções auxiliares.	múltiplas operações no programa. Leitura de números do utilizador com a função scanf. Verificação de erros, como divisão por zero. Teste e execução do programa para validar as funções implementadas. Ajuste de possíveis erros no código.					
Conclusão	Aferir competências adquiridas durante a sessão Estabelecer a ponte com a sessão seguinte	Conteúdos desenvolvidos ao longo da sessão	Método Expositivo –Exposição Dialogada Método Interrogativo –	Plenário: Diálogo Sumula da sessão	Computador Projetor de vídeo Tela de projeção	10 minutos	Observação não focada (postura e atenção) Avaliação

			Colocação de Questões		PowerPoint Papel e caneta		contínua

Assinatura da docente: _____

AGRUPAMENTO DE ESCOLAS DE SANTO ANDRÉ - 120340 -

Prática Supervisionada

2024/2025

Identificação do Estagiário

Nome: João Alberto Fontinha de Brito
Agrupamento: de Escolas de Santo André
Departamento: Matemática e Informática **Grupo de Recrutamento:** 550
Disciplina: Programação **Ano:**1º **Turma:** L
Data: xx/xx/2025

AGRUPAMENTO DE ESCOLAS DE SANTO ANDRÉ - 120340 -

Contextualização (escola, turma e unidade didática)

A turma 1.ºL está inserida na escola Secundária de Santo André, a qual pertence ao Agrupamento de escolas de Santo André, no Barreiro. A turma frequenta o Curso Profissional de Técnico de Sistemas - Informática. A turma é composta por 13 alunos, sendo que doze alunos são do sexo masculino e um do sexo feminino. A média de idades da turma ronda os 15 anos de idade. Existem dois alunos estrangeiros que beneficiam do apoio à língua não materna. Existe um aluno com RTP (Relatório Técnico Pedagógico). Um outro aluno é paquistanês e não fala a língua portuguesa, e é um aluno de nível de língua portuguesa de iniciação, A1 e A2. Beneficia de adaptações curriculares não significativas e também beneficia de adequações ao processo de avaliação assim como a antecipação e reforço das aprendizagens. É importante também referir que esta turma metade são pertencentes aos cursos Técnico profissionais de Informática – Sistemas e a outra metade pertencente ao curso de Gestão e programação de Sistemas Informáticos. Perante algumas observações que realizei durante as aulas da professora Margarida Batista, pude observar que a turma conta alguns alunos que apresentam dificuldades de aprendizagem.

AGRUPAMENTO DE ESCOLAS DE SANTO ANDRÉ - 120340 -

Departamento de Matemática e Informática | Grupo de Recrutamento: 550 – Informática

Ano Letivo: 2024/2025

Plano de Aula nº 1						
Ano: 1.º **Turma:** L	**Disciplina**: Programação			**Aula nº**: 126 e 127		
Data: xx/01/2025	**Hora:** 10:00 – 11:45		**Duração:** 50 min + 50 min			
PROFESSOR:	João Brito					
UFCD 0784	Funções e Estruturas					
SUMÁRIO	• Resolução de uma ficha formativa no ambiente de programação **Dev-C++** para aplicação prática dos conceitos abordados nas aulas anteriores. Correção e discussão dos exercícios da ficha formativa.					
CONTEÚDOS	• Prática computacional no ambiente de programação Dev C++. • Resolução de uma ficha formativa para aplicação de conceitos abordados nas aulas anteriores relativos a funções. • Correção da ficha de trabalho.					
OBJECTIVOS GERAIS	• Promover a prática computacional no ambiente Dev C++. • Consolidar o conhecimento sobre funções. • Resolver problemas de forma autónoma. • Refletir sobre as aprendizagens.					
Objetivos Específicos	**Conteúdos**	**Metodologias**	**Atividades**	**Recursos**	**Duração**	**Avaliação**

AGRUPAMENTO DE ESCOLAS DE SANTO ANDRÉ - 120340 -

Introdução	Reconhecer os objetivos específicos para o módulo e sessão Refletir acerca dos conhecimentos que têm acerca dos conteúdos em análise	Apresentação dos conteúdos do módulo ao grupo Apresentação dos objetivos específicos para a sessão	Método Expositivo Resolução de uma ficha formativa	Plenário: Diálogo Apresentação de diapositivos	Computador Projetor de vídeo Tela de projeção PowerPoint Papel e caneta	10 minutos	Observação não focada (postura e atenção)

AGRUPAMENTO DE ESCOLAS DE SANTO ANDRÉ - 120340 -

Desenvolvimento	Utilizar o ambiente de programação Dev C++ para implementar e testar programas relacionados a funções. Aplicar os conceitos de funções, incluindo estrutura, argumentos e retorno de valores, através da resolução de exercícios práticos. Resolver os exercícios de programação de forma autónoma, reforçando a compreensão dos conteúdos abordados. Analisar e corrigir erros através de uma ficha	Utilização e familiarização com a interface do Dev C++. Execução, teste e depuração de programas em C. Estrutura de uma função: cabeçalho, corpo e instrução return. Utilização de funções void e funções com retorno de valor. Passagem de argumentos para funções (por valor e referência).	Método Expositivo – Exposição Dialogada Método Interrogativo – Colocação de Questões e dúvidas Desenvolvimento de uma ficha formativa	Plenário: Diálogo Apresentação de diapositivos tendo por base exemplos de código em C.	Computador Projetor de vídeo Tela de projeção PowerPoint Quadro branco Papel e caneta	80 minutos	Observação não focada (postura e atenção) Avaliação continua formativa Avaliação continua

AGRUPAMENTO DE ESCOLAS DE SANTO ANDRÉ - 120340 -

	formativa, de forma a promover o ensino e a aprendizagem contínua.	Aplicação prática dos conceitos abordados em exercícios específicos. Desenvolvimento de programas modulares com subprogramas. Identificação de erros e inconsistências em programas. Revisão de soluções propostas na ficha formativa.					
Conclusão	Aferir competências adquiridas durante a sessão Estabelecer a ponte com a sessão seguinte	Conteúdos desenvolvidos ao longo de toda a sessão.	Método Expositivo Exposição de exercícios práticos.	Plenário: Diálogo Sumula da sessão	Computador Projetor de vídeo Tela de projeção	10 minutos	Observação não focada (postura e atenção) Avaliação

AGRUPAMENTO DE ESCOLAS DE SANTO ANDRÉ - 120340 -

			Colocação de Questões		Papel e canteta PowerPoint		contínua

Assinatura da docente: _____

AGRUPAMENTO DE ESCOLAS DE SANTO ANDRÉ - 120340 -

Prática Supervisionada

2024/2025

Identificação do Estagiário			
Nome: João Alberto Fontinha de Brito			
Agrupamento: de Escolas de Santo André			
Departamento: Matemática e Informática			**Grupo de Recrutamento:** 550
Disciplina: Programação	**Ano:**1º	**Turma:** L	
Data: xx/xx/2025			

AGRUPAMENTO DE ESCOLAS DE SANTO ANDRÉ - 120340 -

Contextualização (escola, turma e unidade didática)

A turma 1.ºL está inserida na escola Secundária de Santo André, a qual pertence ao Agrupamento de escolas de Santo André, no Barreiro. A turma frequenta o Curso Profissional de Técnico de Sistemas - Informática. A turma é composta por 13 alunos, sendo que doze alunos são do sexo masculino e um do sexo feminino. A média de idades da turma ronda os 15 anos de idade. Existem dois alunos estrangeiros que beneficiam do apoio à língua não materna. Existe um aluno com RTP (Relatório Técnico Pedagógico). Um outro aluno é paquistanês e não fala a língua portuguesa, e é um aluno de nível de língua portuguesa de iniciação, A1 e A2. Beneficia de adaptações curriculares não significativas e também beneficia de adequações ao processo de avaliação assim como a antecipação e reforço das aprendizagens. É importante também referir que esta turma metade são pertencentes aos cursos Técnico profissionais de Informática – Sistemas e a outra metade pertencente ao curso de Gestão e programação de Sistemas Informáticos. Perante algumas observações que realizei durante as aulas da professora Margarida Batista, pude observar que a turma conta alguns alunos que apresentam dificuldades de aprendizagem.

AGRUPAMENTO DE ESCOLAS DE SANTO ANDRÉ - 120340 -

Departamento de Matemática e Informática	Grupo de Recrutamento: 550 – Informática Ano Letivo: 2024/2025

Plano de Aula nº 1						
Ano: 1.º **Turma:** L	**Disciplina**: Programação			**Aula nº**: 121		
Data: xx/01/2025	**Hora:** 11:50 – 12:40		**Duração:** 50 min			
PROFESSOR:	João Brito					
UFCD 0784	Funções e Estruturas					
SUMÁRIO	• Resumo dos principais conceitos abordados nas aulas anteriores. • Aplicação de um questionário avaliativo para consolidar os conteúdos abordados.					
CONTEÚDOS	• Resumo dos conceitos abordados. • Aplicação de um questionário avaliativo					
OBJECTIVOS GERAIS	• Consolidar os principais conceitos abordados ao longo das aulas anteriores, relacionados com programação modular e funções. • Demonstrar a aplicação prática e teórica dos conteúdos através da revisão dos tópicos fundamentais. • Avaliar o nível de compreensão dos alunos sobre os conteúdos lecionados, utilizando um questionário avaliativo. • Promover a reflexão sobre a importância da organização e reutilização de código em programação.					
Objetivos Específicos	**Conteúdos**	**Metodologias**	**Atividades**	**Recursos**	**Duração**	**Avaliação**

AGRUPAMENTO DE ESCOLAS DE SANTO ANDRÉ - 120340 -

Introdução	Reconhecer os objetivos específicos para o módulo e sessão Refletir acerca dos conhecimentos que têm acerca dos conteúdos em análise	Apresentação dos conteúdos do módulo ao grupo Apresentação dos objetivos específicos para a sessão	Método Expositivo Exposição Dialogada Método Interrogativo – Colocação de Questões	Plenário: Diálogo Apresentação de diapositivos	Computador Projetor de vídeo Tela de projeção PowerPoint Papel e caneta	10 minutos	Observação não focada (postura e atenção)

AGRUPAMENTO DE ESCOLAS DE SANTO ANDRÉ - 120340 -

Desenvolvimento	Consolidar os conhecimentos sobre programação modular e funções, abordados nas aulas anteriores. Revisão dos conceitos fundamentais, como a estrutura de funções, tipos de funções, parâmetros e argumentos. Compreender a importância do ciclo de desenvolvimento de um programa e da função main(). Identificar as vantagens da modularização e da reutilização de código em programação.	Programação Modular: Conceito, estrutura de funções, parâmetros e vantagens. Ciclo de Desenvolvimento: Etapas de edição, compilação e execução. A Função main(): Ponto de entrada do programa. Vantagens das Funções: Reutilização, organização e manutenção. Avaliação Final: Questionário avaliativo sobre os conteúdos lecionados nas aulas anteriores.	Método Expositivo – Exposição Dialogada Método Interrogativo – Colocação de Questões Método Afirmativo Demonstrativo	Plenário: Diálogo Apresentação de diapositivos tendo por base exemplos de código em C.	Computador Projetor de vídeo Tela de projeção PowerPoint Quadro branco Papel e caneta	35 minutos	Observação não focada (postura e atenção) Avaliação continua formativa Avaliação continua

AGRUPAMENTO DE ESCOLAS DE SANTO ANDRÉ - 120340 -

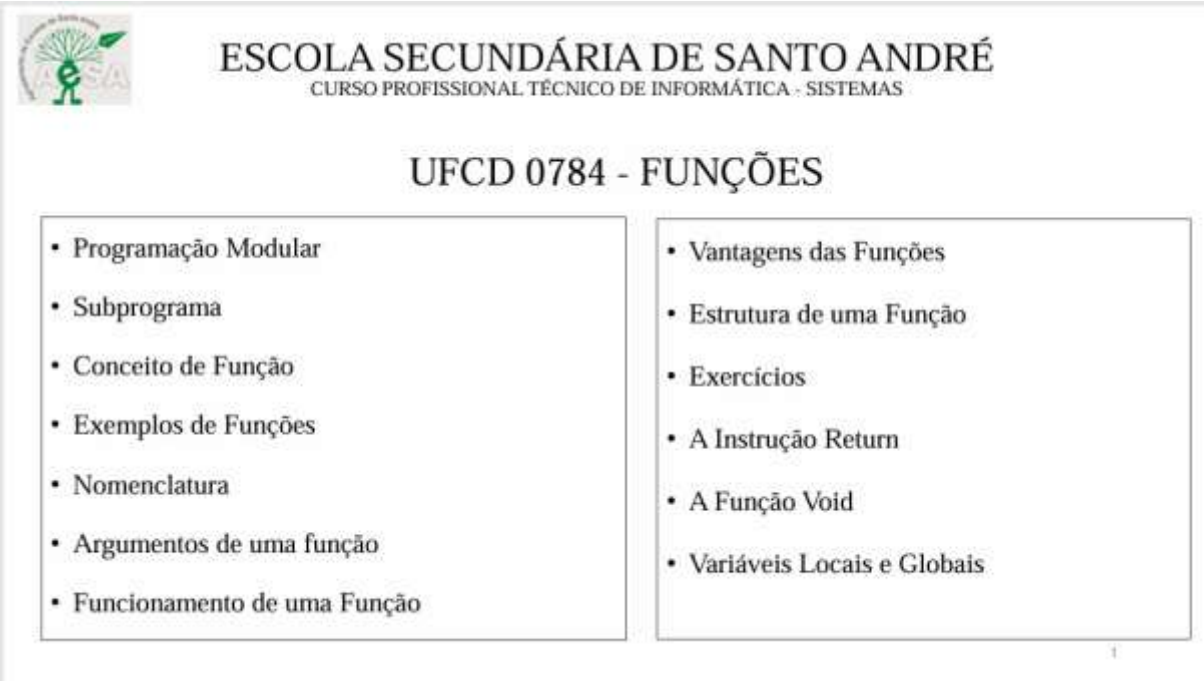
	Avaliar o nível de entendimento dos conteúdos através da realização de um questionário avaliativo. Estimular a reflexão crítica sobre os tópicos abordados e a aplicação prática dos conceitos em programação.						
Conclusão	Aferir competências adquiridas durante a sessão Estabelecer a ponte com a sessão seguinte	Conteúdos desenvolvidos ao longo da sessão	Método Expositivo –Exposição Dialogada Método Interrogativo – Colocação de Questões	Plenário: Diálogo Sumula da sessão	Computador Projetor de vídeo Tela de projeção PowerPoint Papel e caneta	5 minutos	Observação não focada (postura e atenção) Avaliação continua

AGRUPAMENTO DE ESCOLAS DE SANTO ANDRÉ - 120340 -

Assinatura da docente: _____

João Brito Plano de sessão 1 Página 7 de 7

Materiais expositivos das aulas supervisionadas



ESCOLA SECUNDÁRIA DE SANTO ANDRÉ

CURSO PROFISSIONAL TÉCNICO DE INFORMÁTICA - SISTEMAS

UFCD 0784 - FUNÇÕES

- Programação Modular
- Subprograma
- Conceito de Função
- Exemplos de Funções
- Nomenclatura
- Argumentos de uma função
- Funcionamento de uma Função

- Vantagens das Funções
- Estrutura de uma Função
- Exercícios
- A Instrução Return
- A Função Void
- Variáveis Locais e Globais

ESCOLA SECUNDÁRIA DE SANTO ANDRÉ

CURSO PROFISSIONAL TÉCNICO DE INFORMÁTICA - SISTEMAS

UFCD 0784 - Funções

Objetivos Gerais	Conteúdos
• Aprender a estruturar programas; • Adquirir a noção de subprograma; • Conhecer as regras de declaração de subprogramas; • Conhecer as regras de execução de subprogramas; • Utilizar corretamente argumento; • Distinguir os diferentes tipos de subprogramas; • Elaborar programas com recurso a subprogramas; • Conhecer as regras para a criação de bibliotecas de subprogramas; • Promover a resolução de exercícios através da programação estruturada.	• Noção de subprograma; • Regras de declaração de subprograma; • Regras de execução de subprograma; • Argumentos; • Tipos de subprogramas; • Programas com recurso a subprogramas;

2

ESCOLA SECUNDÁRIA DE SANTO ANDRÉ

CURSO PROFISSIONAL TÉCNICO DE INFORMÁTICA - SISTEMAS

AULA 1 (50 min)

Conteúdos Específicos

Tópicos a abordar na aula:
• Revisão dos conceitos:
  • Programação Modular:
  • Conceito
  • Vantagens
• Ciclo de desenvolvimento de um programa em C
• Estrutura de um programa em C
• Exemplos práticos

3

ESCOLA SECUNDÁRIA DE SANTO ANDRÉ
CURSO PROFISSIONAL TÉCNICO DE INFORMÁTICA - SISTEMAS

Sumário da aula 1

- Introdução ao conceito de Programação Modular e suas principais vantagens.
- Ciclo de desenvolvimento de um programa em C.
- Estrutura de um programa em C.
- A função main().
- Discussão sobre a importância da: função main() e módulos para o desenvolvimento de software.
- Aprendizagem através de exemplos práticos.

4

PROGRAMAÇÃO MODULAR

- **A programação modular é uma técnica de desenvolvimento de software que envolve a divisão de um programa em partes menores, chamados módulos ou subprogramas.**
- Cada módulo é responsável por realizar uma tarefa específica do programa que pode ser desenvolvido, testado e mantido de forma independente.
- Este conceito é essencial para organizar o código, torná-lo mais legível, facilitar a manutenção e permitir a reutilização de componentes noutros programas.

5

VANTAGENS

Vantagens da Programação Modular:

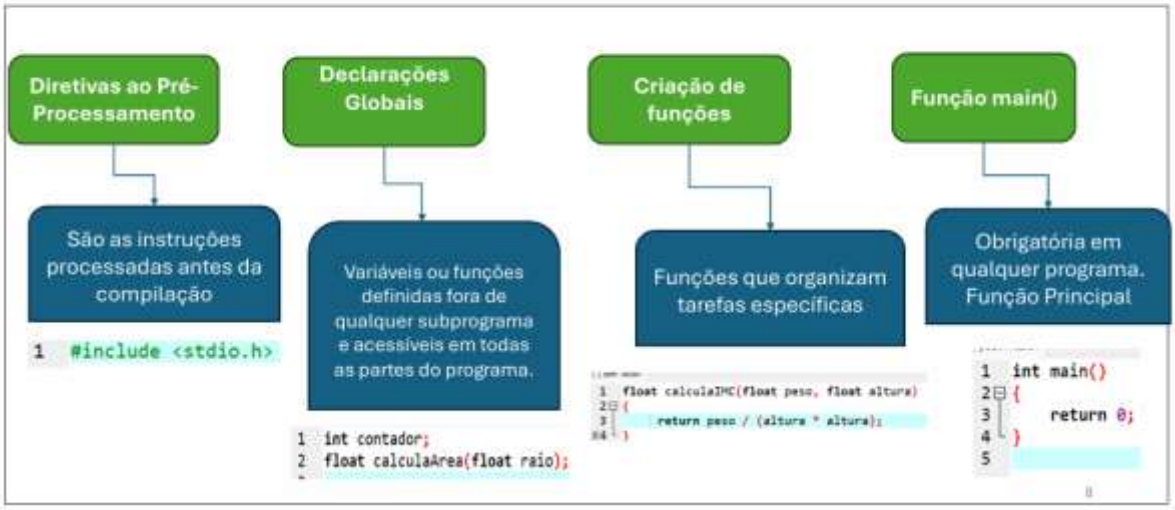
- **Divisão de tarefas** – A programação modular divide o programa em várias funções em vez de escrever todo o programa numa única função na main().
- **Reutilização de código** – Os módulos podem ser reutilizados em partes diferentes do programa reduzindo assim a redundância de código.
- **Facilidade na manutenção** – Como o código está dividido torna-se mais fácil encontrar erros e corrigir os mesmos assim como a sua atualização sem modificar todo o código já escrito.

VANTAGENS

Vantagens da programação modular:

- **Legibilidade** – O código fica mais organizado e fácil de compreender porque cada módulo possui uma função bem definida.
- **Facilidade na depuração** – Os problemas encontrados podem ser isolados e resolvidos em módulos individuais, sem influenciar todo o programa.
- **Trabalho de equipa** – Permite que diferentes programadores possam trabalhar em módulos diferentes em simultâneo o que facilita o trabalho em equipa.

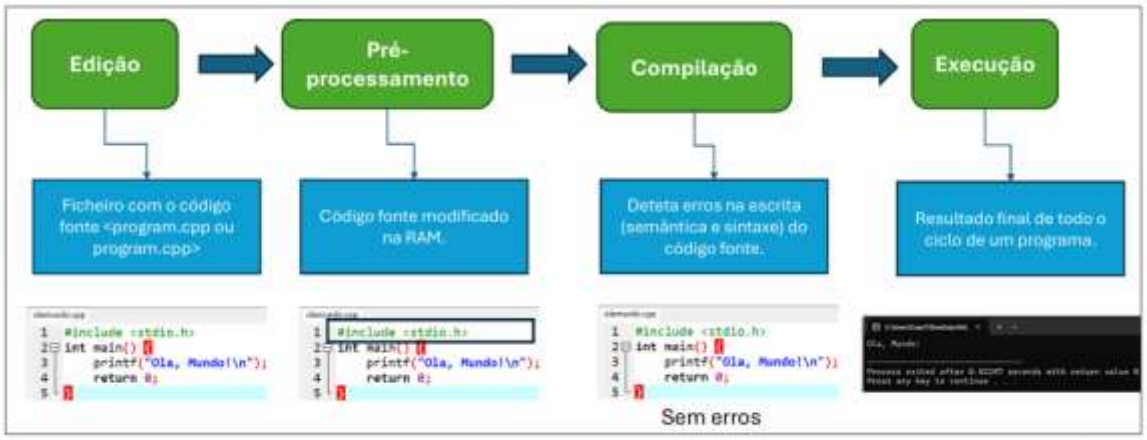
A ESTRUTURA DE UM PROGRAMA EM C



CICLO DE DESENVOLVIMENTO DE UM PROGRAMA

- **É um processo que transforma o código-fonte escrito pelo programador num programa executável.**
- É composto por várias etapas:
  - edição;
  - pré-processamento;
  - compilação;
  - execução.
- Tem como objetivo garantir que o programa funciona corretamente e sem erros.

CICLO DE DESENVOLVIMENTO DE UM PROGRAMA



10

EDIÇÃO

- Momento principal e fundamental em que o programador escreve o código num editor de texto ou IDE como por exemplo o Dev C++.
- É das etapas fundamentais em que o programador além de escrever o código cria o programa com uma determinada lógica e com as funcionalidades desejadas.
- O ficheiro criado deve conter sempre nome de program.cpp

olamundo.cpp

```
1  #include <stdio.h>
2  int main() {
3      printf("Ola, Mundo!\n");
4      return 0;
5  }
```

11

PRÉ-PROCESSAMENTO

- O pré-processador trabalha o código de forma a que fique preparado para ser compilado.
- Sem este procedimento do #include, o código não reconheceria a função printf().
- O pré-processador em C realiza uma série de transformações no código fonte antes que o compilador comece a compilar o programa.

olamundo.cpp

```
1  #include <stdio.h>
2  int main() {
3      printf("Ola, Mundo!\n");
4      return 0;
5  }
```

12

COMPILAÇÃO

- **Tradutor principal do código fonte para a linguagem máquina.**
- **Identifica erros de sintaxe e de semântica no código fonte.**
- Cria ficheiros intermediários para a próxima etapa.
- O código da máquina é uma sequência de instruções que o processador do computador executa.
- Funciona como um conversor uma vez que lê o código em C e converte-o para código de máquina.

olamundo.cpp

```
1  #include <stdio.h>
2  int main() {
3      printf("Ola, Mundo!\n");
4      return 0;
5  }
```

Sem erros

olamundo.cpp

```
1  #include <stdio.h>
2  int main() {
3      printf("Ola, Mundo!\n");
4      return 0;
5
```

Com erros

13

EXECUÇÃO

- **O programa executável (program.exe) está pronto nesta fase para ser executado.**
- **Testar o programa e verificar se funciona como esperado. Após cada etapa podem existir erros e é necessário corrigir.**
- O utilizador interage com o programa e valida a saída.
- Durante as etapas da edição, pré-processamento e compilação o programador pode corrigir erros encontrados.

14

A FUNÇÃO MAIN()

- **A função main() é o ponto de entrada de qualquer programa em C.**
- **Estrutura básica do C:**

```c
#include <stdio.h>
int main()
{
   printf("Olá, Mundo!\n");
   return 0;
}
```

- Inicio e término na função main().
- O valor de retorno (return 0) indica que o programa terminou com sucesso.

15

ESTRUTURA DE UM PROGRAMA EM C

```
/* Primeiro Programa em C */ comentários

#include <stdio.h> /*biblioteca de E/S */
 SUBPROGRAMAS
main()      /*função principal - inicio do
             programa*/
{            /*marca inicio da função*/
   printf("Meu primeiro programa em C\n");
            /*escreve no monitor e muda de
  linha*/
}            /*marca o fim da função*/
```

16

EXEMPLO DE UM PROGRAMA EM C

Neste exemplo, temos um programa que utiliza funções separadas para somar e subtrair dois números.

- Quantas funções se encontram neste exemplo?
- Existe alguma função da própria biblioteca de funções do C? Se sim qual?
- Quantas funções neste exemplo foram criadas?

```
[*] CODIGO.cpp
 1  #include <stdio.h>
 2  int a, b;
 3  int soma(int a, int b)
 4  {
 5      return a + b;
 6  }
 7  int subtrai(int a, int b)
 8  {
 9
10      return a - b;
11  }
12  int main()
13  {
14      //int num1 = 10, num2 = 5;
15
16      printf("Soma: %d\n", soma(a, b));
17      printf("Subtracao: %d\n", subtrai(a, a));
18
19      return 0;
20  }
```

17

RESUMO DA AULA

- Falámos no conceito de Programação Modular e suas vantagens.
- Falámos sobre o Ciclo de desenvolvimento de um Programa e suas etapas.
- Vimos como o código-fonte pode ser transformado num programa executável.
- Analisámos a Estrutura de um Programa em C.
- Introdução à função main().
- Vimos exemplos de estruturas e código simples.
- Realizámos exemplos práticos através de demonstrações simples para exemplificar os conceitos abordados.

PRÓXIMA AULA

- Iremos falar no conceito de Função vs Subprograma.
- Tipos de Subprogramas.
- Realizar um review sobre as três principais funções: main(); Printf() e scanf().
- Perceber a Nomenclatura das funções.
- Características de uma função.
- Ver Parâmetros e argumentos de uma função.
- Funcionamento de uma função e as suas vantagens.
- Funções com instruções e Estrutura de uma função.
- Exercícios conduzidos step by step.

18

FIM DA AULA 1

19

ESCOLA SECUNDÁRIA DE SANTO ANDRÉ
CURSO PROFISSIONAL TÉCNICO DE INFORMÁTICA - SISTEMAS

AULA 2 e 3

Conteúdos Específicos
• Conceito de Função • Nomenclatura de uma Função • Estrutura de uma Função • A instrução Return • Função Void • Argumentos de uma Função • Exemplos práticos

ESCOLA SECUNDÁRIA DE SANTO ANDRÉ
CURSO PROFISSIONAL TÉCNICO DE INFORMÁTICA - SISTEMAS

Sumário da aula 2 e 3

- Conceito de função e diferenças entre subprogramas e funções.
- Tipos de funções:
  - Com retorno de valor.
  - Do tipo void (sem retorno de valor).
- Nomenclatura e características das funções.
- Argumentos de uma função.
- Resolução de exercícios práticos conduzidos.

SUBPROGRAMA vs FUNÇÃO

Diferenças entre SUBPROGRAMA E FUNÇÃO

- Um **subprograma** é **um bloco de código independente**, ou seja fora da **função main()** e que **executa uma tarefa específica** como por exemplo calcular uma soma.
- Uma função é um tipo especifico de subprograma que retorna um valor - **Exemplo 1.**
- Em C os **subprogramas** são **funções** que retornam um valor (return) ou não retornam valor através da função (void) - **Exemplo 2.**

Exemplo 1

```
4  int soma(int a, int b) {
5      return a + b;
6  }
```

Exemplo 2

```
4  void saudacao()
5  {
6      printf("Olá! Bem-vindo!\n");
7  }
```

TIPOS DE SUBPROGRAMAS

Funções com Retorno de valor:

- **São funções** que executam uma tarefa e **retornam um valor** ao programa que as chamou.
- O **tipo de retorno** pode ser um **número inteiro**, **ponto flutuante**, **caracter**, entre outros.
- São usadas quando é necessário calcular ou obter um resultado que será usado posteriormente.
- Para serem executadas têm de ser chamadas.

Soma.cpp

```
1  #include <stdio.h>
2
3  int soma(int a, int b)
4  {
5      return a + b;
6  }
7
8  int main()
9  {
10     int resultado = soma(5, 3);
11     printf("Resultado da soma: %d\n", resultado);
12     return 0;
13 }
14
```

TIPOS DE SUBPROGRAMAS

Funções Void sem Retorno de valor:

- **São funções** que executam uma tarefa, mas **não retornam nenhum valor.**
- São utilizadas para:
  - Exibição de mensagens;
  - Alteração de variáveis;
  - Validações.

[*] Soma.cpp Void.cpp

```c
#include <stdio.h>

void exibirMensagem()
{
    printf("Ola, mundo!\n");
}

int main()
{
    exibirMensagem();
    return 0;
}
```

RELEMBRANDO

Embora ainda sem saber como escrever uma função, já as temos utilizado ao longo dos nossos programas.

Exemplos:

- main() - Função principal do programa.
- printf(…) – Função de **escrita associada à** biblioteca **stdio.h – (Output).**
- scanf(…) – Função de leitura (**também associada à biblioteca stdio.h**) e armazenamento – **(Input).**

```c
#include <stdio.h>

int soma(int a, int b) {
    return a + b;
}

int main() {
    int a, b;

    printf("Introduza o primeiro número: ");
    scanf("%d", &a);

    printf("Introduza o segundo número: ");
    scanf("%d", &b);

    int resultado = soma(a, b);
    printf("Resultado da soma: %d\n", resultado);

    return 0;
}
```

NOMENCLATURA DE FUNÇÕES

Nome de uma função:

- Deve seguir as mesmas regras das variáveis;
- Ser único no programa;
- Fácil leitura.

NOMENCLATURA DE FUNÇÕES

Relembrando as regras:

- O nome de uma variável/função pode ser constituído por letras do alfabeto (minúsculas, maiúsculas), dígitos e ainda pelo carácter underscore.
- O primeiro carácter não pode ser um digito. Terá de ser uma letra ou o carácter underscore.
- É desaconselhável a utilização do underscore para inicio da variável/função.
- Uma variável/função não pode nunca ter um nome que esteja reservado à própria linguagem C como por exemplo: float, if, for etc….

CARACTERISTICAS DE UMA FUNÇÃO

- Cada função precisa de ter um nome único para ser invocada no programa.
- Uma função pode ser chamada a partir de outras funções.
- A função deve realizar **uma única tarefa bem definida**.
- Deve comportar-se como uma "caixa negra": **o importante é o resultado**, não o processo interno.
- O código da função deve ser **o mais genérico possível** para ser reutilizável.
- Pode receber **argumentos** para adaptar-se a diferentes situações.
- Pode **retornar um valor** como **resultado** do seu processamento.

PARÂMETROS E ARGUMENTOS

Características:

- **Argumento:** Variável que serve de cálculo à função.
- **Passagem de Argumentos:** Os argumentos são passados para a função entre parênteses e separados por vírgulas.

  Exemplo : soma(10, 20).

ARGUMENTOS DE UMA FUNÇÃO

O que acontece neste exemplo?

Há retorno de valor?

Ou há impressão de um resultado?

```c
void saudacao(char nome[], int idade)
{
    printf("Olá, %s tens %d anos.\n", nome, idade);
}
int main()
{
    saudacao("João", 25);
    saudacao("Maria", 30);

    return 0;
}
```

ARGUMENTOS DE UMA FUNÇÃO

O que acontece neste exemplo?

Há retorno de valor?

Ou há impressão de um resultado?

```c
# include <stdio.h>

int soma(int a, int b)
{
        return a + b;
}

int main()
{
        printf("%d\n", soma(3,5));
        return 0;
}
```

VARIÁVEIS LOCAIS E GLOBAIS

Variável local:

- É definida **dentro da função**.
- A variável é válida e assume o valor atribuído dentro da função.

```c
void funcao() {
    int local = 10; // Variável local
    printf("Variável local: %d\n", local);
}
```

Variável Global:

- É definida **fora da função**.
- A variável é válida e assume valor atribuído em todo o programa.

```c

int global = 20; // Variável global

void funcao() {
    printf("Variável global na função: %d\n", global);
}
```

NOMENCLATURA DE FUNÇÕES

Exercício prático conduzido:

1. Função que permite calcular a soma de dois números inteiros.
2. Função sem retorno que imprime uma mensagem de boas-vindas.
3. Função que permite verificar se um número é par ou ímpar.

1

```c

int soma(int a, int b)
{
    return a + b;
}
```

2

```c

void mensagemBoasVindas()
{
    printf("Bem-vindo ao programa de exemplos de funções!\n");
}
```

3

```c
void verificaParImpar(int numero)
{
    if (numero % 2 == 0)
        printf("%d é par.\n", numero);
    else
        printf("%d é ímpar.\n", numero);

}
```

PARÂMETROS E ARGUMENTOS

Exercício explicativo:

- Escreva um programa em C que utilize uma função que receba dois argumentos para calcular a área do retângulo e retornar o resultado.
- Identificação das variáveis.
- Escreve as funções necessárias da biblioteca de funções do C para que o utilizador possa na consola calcular os valores pretendidos.

```c
#include <stdio.h>

float calculaAreaRetangulo(float largura, float altura)
{
    return largura * altura;
}

int main()
{
    float largura, altura;

    printf("Digite a largura do retângulo: ");
    scanf("%f", &largura);

    printf("Digite a altura do retângulo: ");
    scanf("%f", &altura);

    float area = calculaAreaRetangulo(largura, altura);

    printf("A área do retângulo é: %.2f\n", area);

    return 0;
}
```

FUNCIONAMENTO DE UMA FUNÇÃO

- **Execução sob Invocação:** O código de uma função só é executado quando esta é invocada (chamada) no programa.
- **Suspensão Temporária do Programa Principal:** Quando a função é chamada, a execução do programa principal é "suspensa" temporariamente. A função executa as suas instruções.
- **Retorno ao Ponto de Invocação:** Após a função terminar, o controlo de execução volta ao local no programa onde a função foi invocada.
- **Recebimento de Valores:** A função pode receber valores que influenciam o seu comportamento.
- **Devolução de Resultado:** A função pode retornar ou não um resultado para o programa que a chamou.

VANTAGENS DE UTILIZAR FUNÇÕES

- **Reaproveitamento de Código:** Permite reutilizar código já escrito, seja por si ou por outros programadores.
- **Evita Repetição de Código:** Previne a repetição de blocos de código em várias partes do programa.
- **Facilidade de Alteração:** Ao modificar uma função, a alteração é feita apenas dentro desta, tornando a manutenção mais rápida e eficiente.
- **Organização do Programa:** Mantém os blocos do programa menores e mais organizados, facilitando a compreensão.
- **Leitura Facilitada:** Melhora a legibilidade do código fonte.
- **Separação de Tarefas:** Permite separar o programa em partes independentes, tornando-o mais modular.

CONCEITO DE FUNÇÃO

- Na programação em C, **todos os programas** devem ter a **função main()**, independentemente de outras funções que incluam.
- **A execução de um programa começa sempre na main()**, mesmo que esta não esteja no início do código.
- **Todas as funções só serão executadas se forem chamadas pela main().**

CONCEITO DE FUNÇÃO

- **Uma função é um bloco de código** que pode ser chamado para executar uma tarefa.
- Permite **agrupar instruções** o que facilita a leitura do código.
- Permite **decompor códigos extensos** em pedaços mais pequenos e reutilizáveis.
- Existem funções mais simples e outras mais complexas. As funções podem conter uma instrução ou várias instruções.

O que significa isto?

Sintaxe em C:

```
nomeDaFuncao()
{
        < bloco de instruções ; >
}
```

FUNÇÕES COM INSTRUÇÕES

- Uma função com **uma instrução:**

```
int soma(int a, int b)
{
    return = a + b;
}
```

- Uma função com **mais instruções:**

```
int calculo(int a, int b)
{
    int soma;
    soma = a + b;
    soma = soma + 1;
    return soma;
}
```

ESTRUTURA De UMA FUNÇÃO

- Uma função com várias instruções:

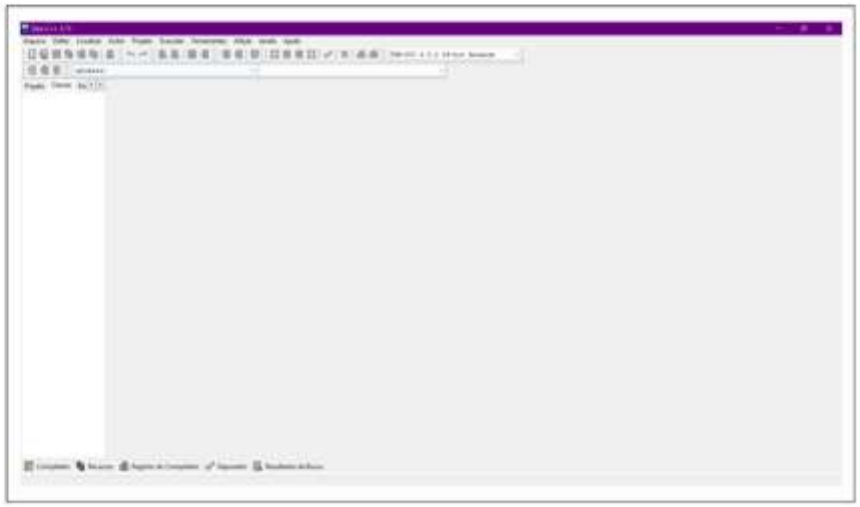
```
int calculo(int a, int b)
{
    int soma;
    soma = a + b;
    soma = soma + 1;
    return soma;
}
```

- Nome da função: calculo
- Cabeçalho da função: int calculo(int a, int b).
- O cabeçalho inclui o tipo de retorno (int), o nome da função (calculo), e os argumentos de entrada (int a, int b).
- As variáveis – São valores que são passados para a função.

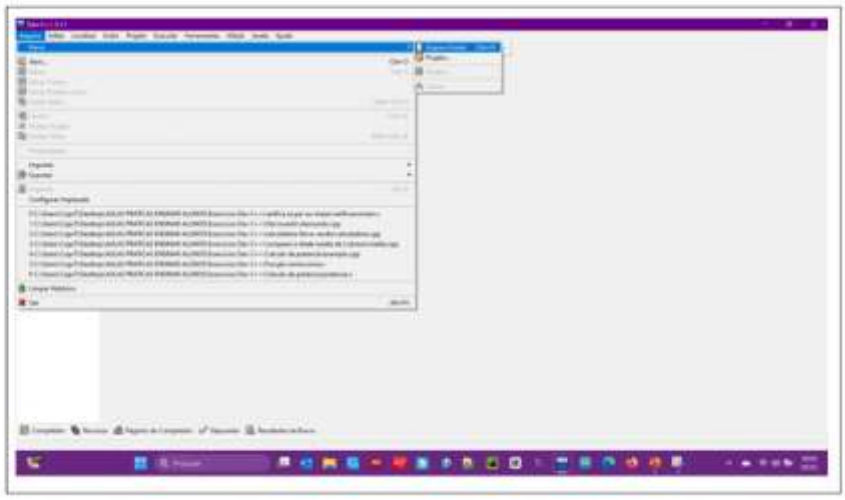
INICIO DO EXERCICIO 1 CONDUZIDO STEP BY STEP

- Vamos criar um programa em C que utilize uma função para calcular a soma de dois números inteiros à escolha do utilizador.

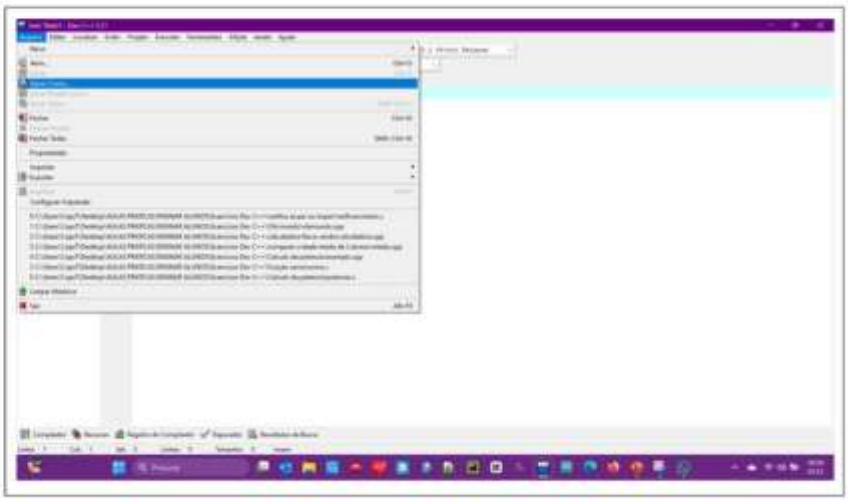
Passo 1 – Abrir o programa Dev C++



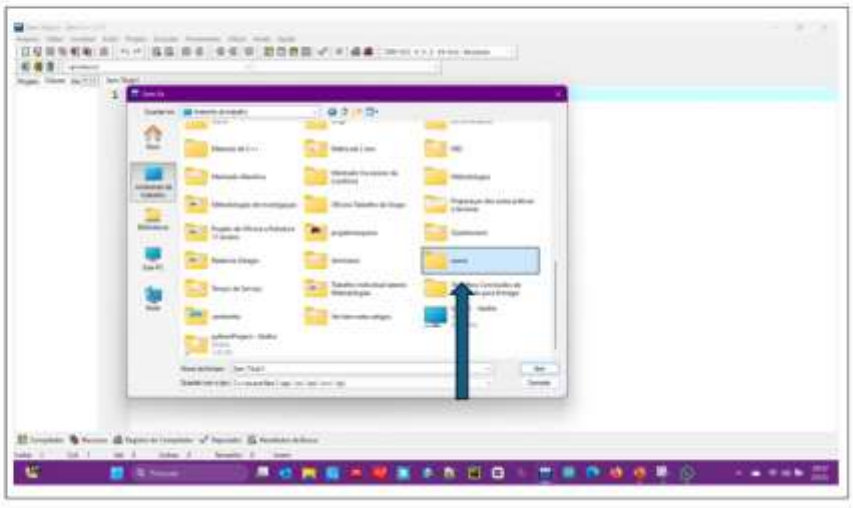
Passo 2 – Vamos a arquivo – Novo – Arquivo Fonte



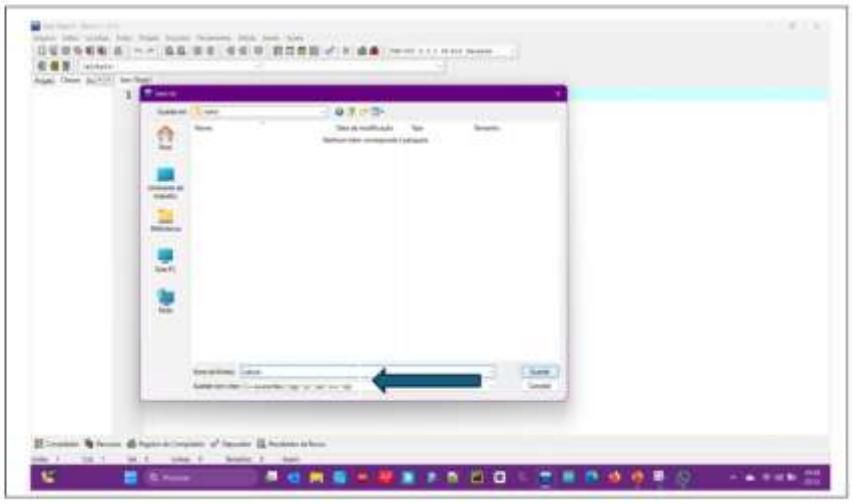
Passo 3 – Vamos agora salvar o ficheiro e vamos a salvar como…



Passo 4 – Vamos agora salvar o ficheiro numa pasta chamada "soma"

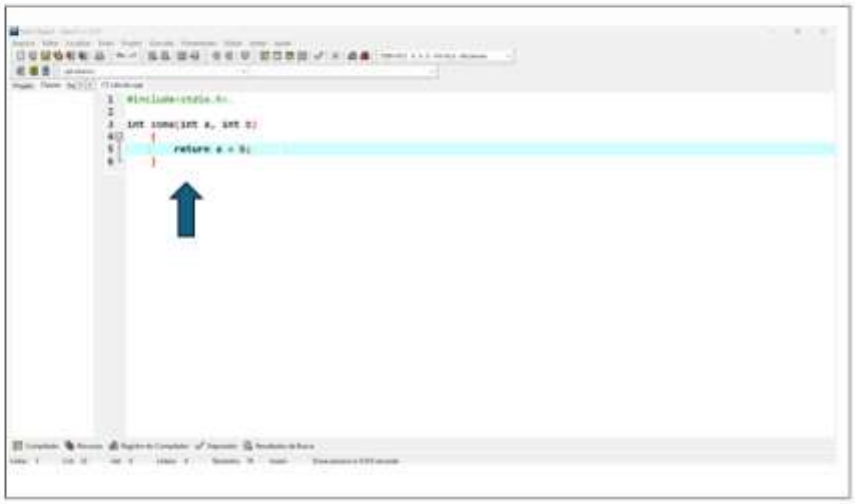


Passo 5 – Vamos agora salvar o ficheiro dentro da pasta soma e dão lhe o nome "calculo"

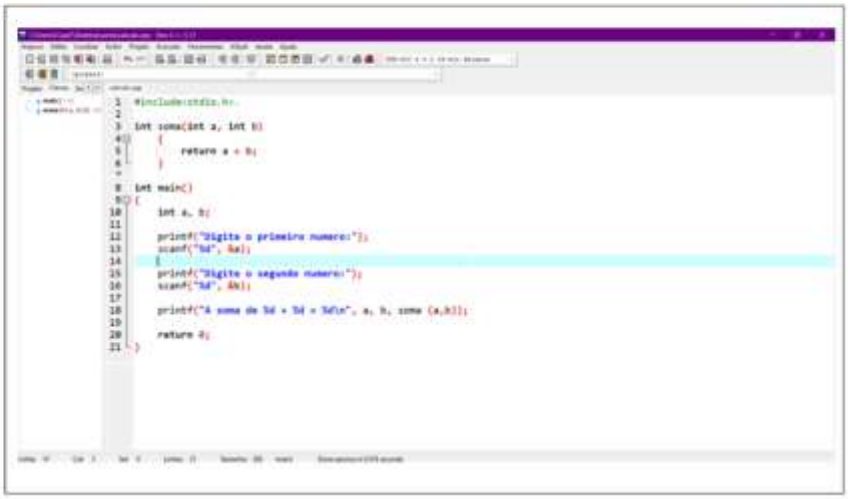


Passo 6 – No topo do programa do Dev C++ linha 1 devemos incluir a biblioteca - #include<stdio.h>

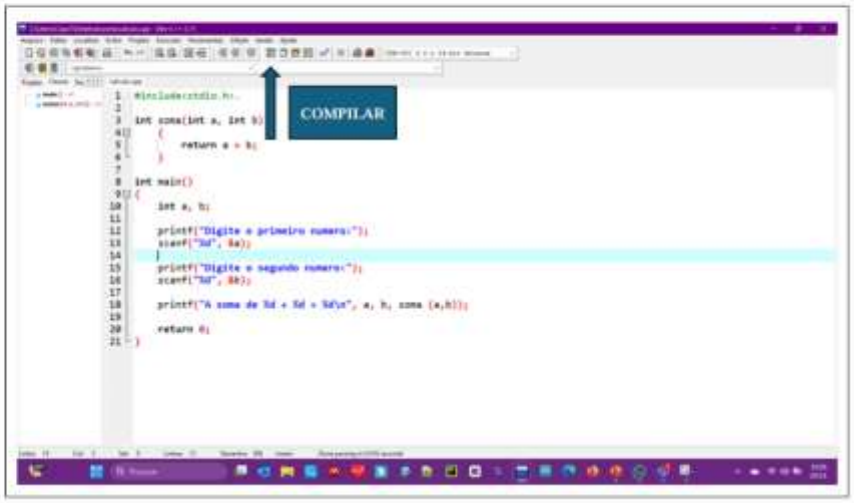
Passo 7 – Vamos declarar a função int soma(int a, int b) antes da função main()



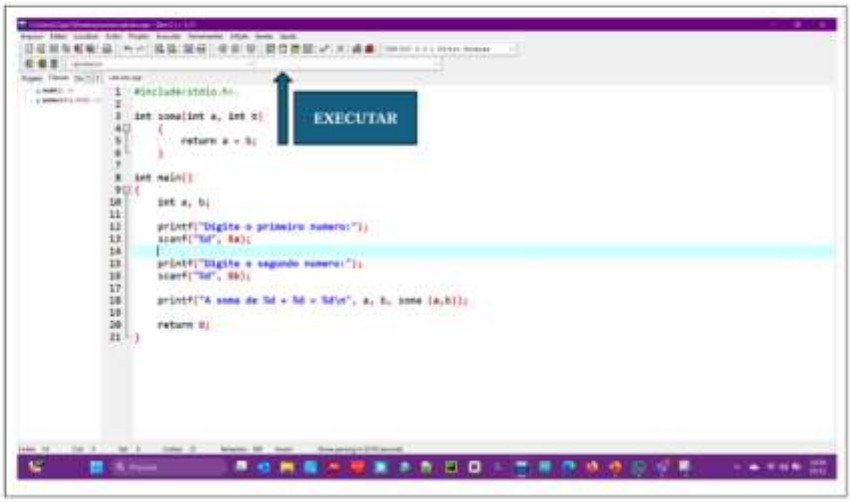
Passo 8 – Vamos implementar a função soma na função main()



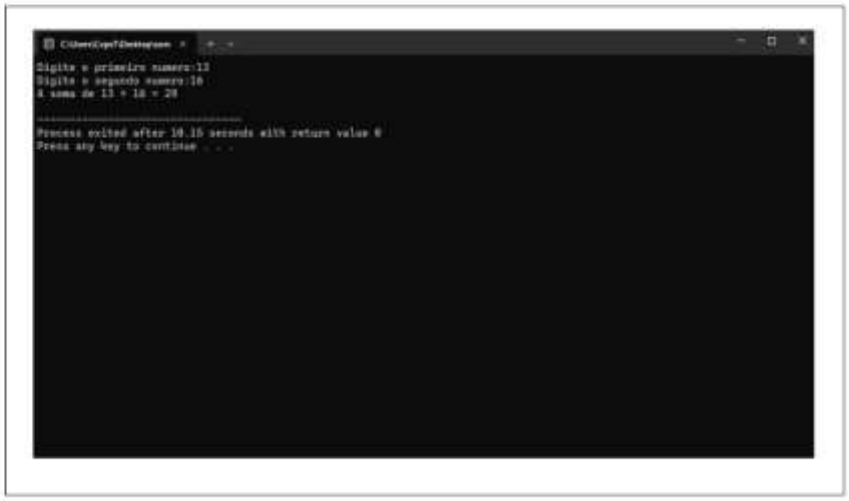
Passo 9 – Agora vamos compilar o código para verificarmos se existem erros



Passo 10 – Agora vamos executar o nosso código



Passo 11 – Programa a correr na consola

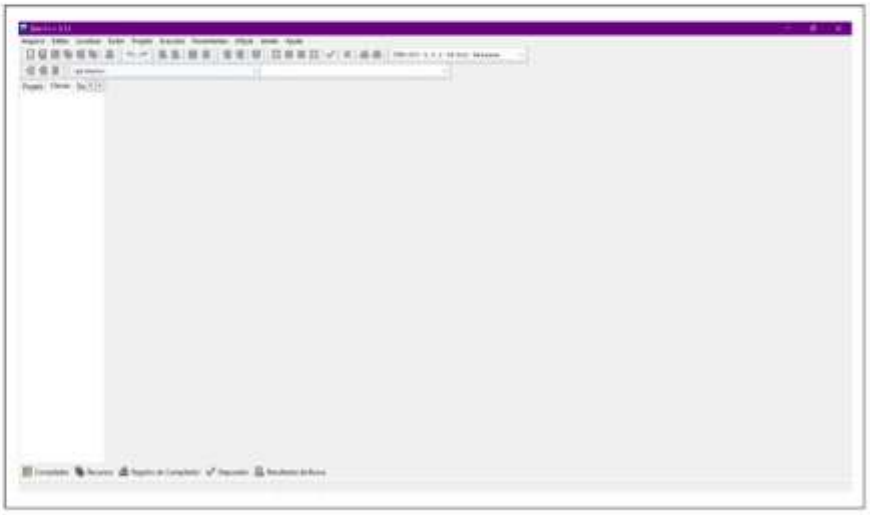


FIM DO EXERCICIO 1 CONDUZIDO STEP BY STEP

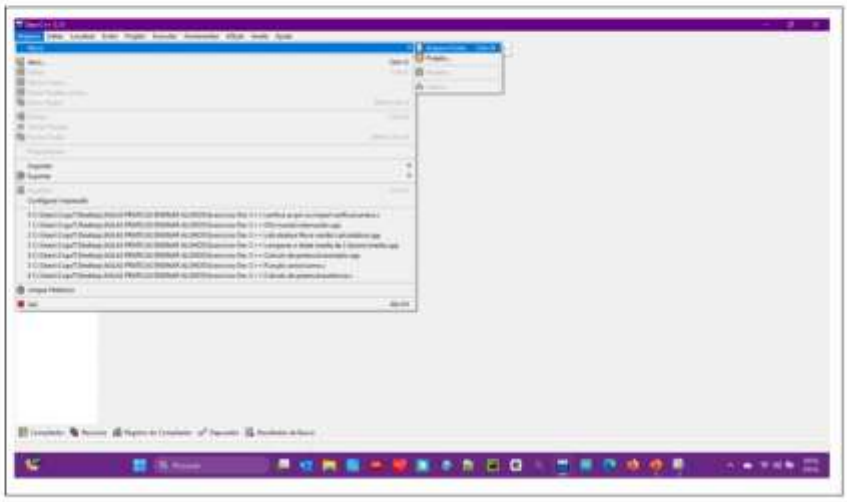
INICIO DO EXERCICIO 2 CONDUZIDO STEP BY STEP

- Desenvolva um programa que calcule a média de três notas de um aluno. Crie uma função com o nome calculaMedia() e que retorne o valor da média. O programa deve exibir a média no ecrã

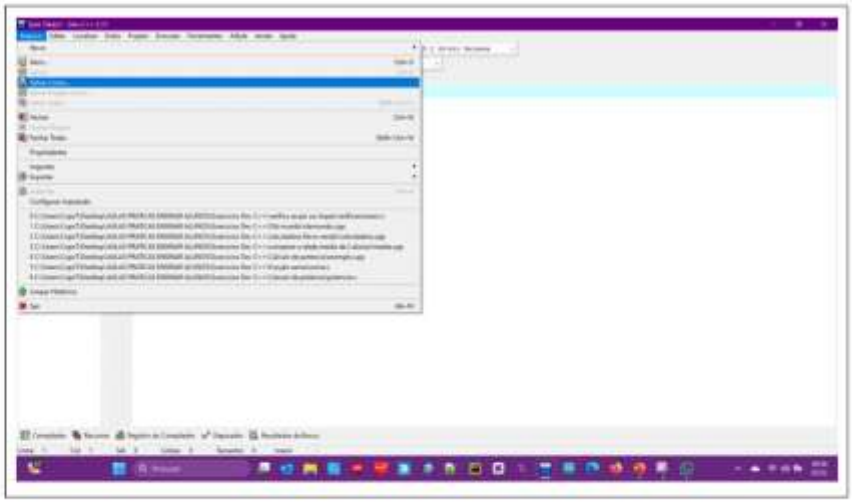
Passo 1 – Abrir o program Dev C++



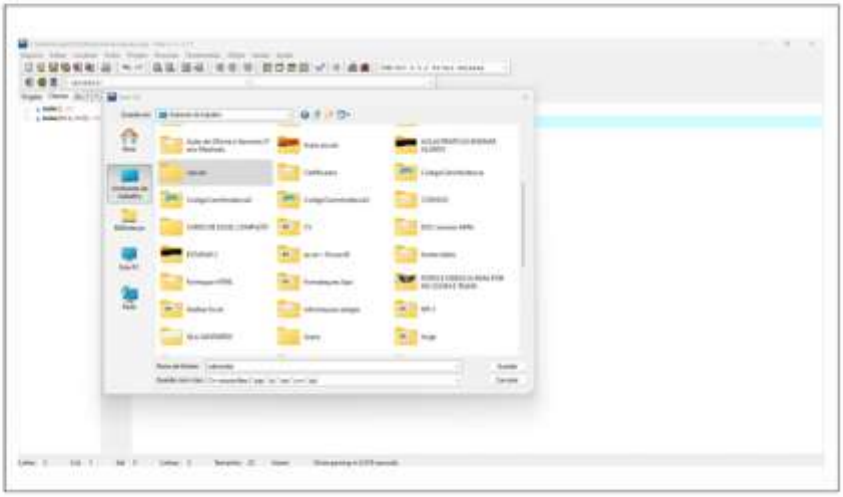
Passo 2 – Vamos a arquivo – Novo – Arquivo Fonte



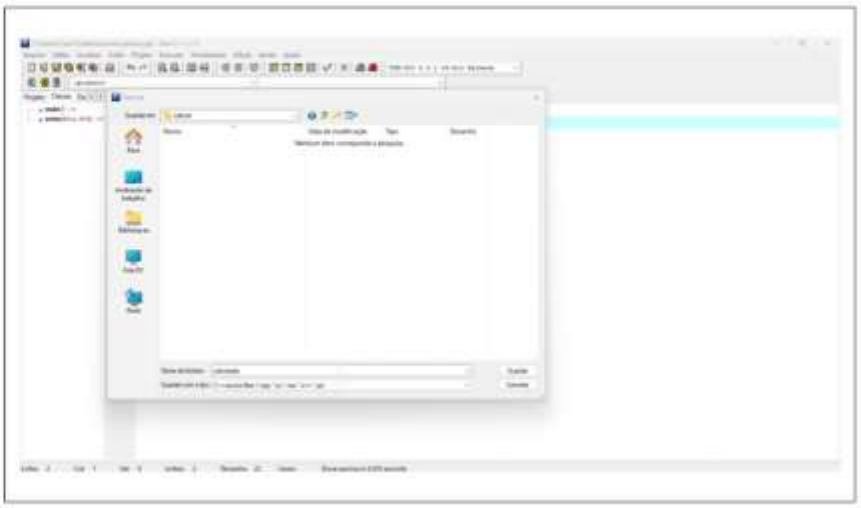
Passo 3 – Vamos agora salvar o ficheiro em salvar como...



Passo 4 – Vamos agora salvar o ficheiro numa pasta chamada "calcmedia"



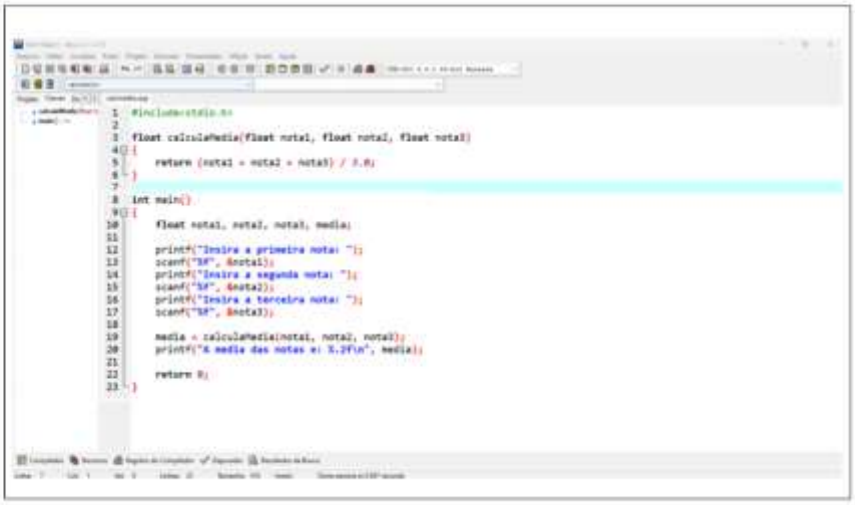
Passo 5 – Vamos guardar o ficheiro com o nome "calcmedia"



Passo 6 – No topo do programa do Dev C++ linha 1 devemos incluir a biblioteca - #include<stdio.h>

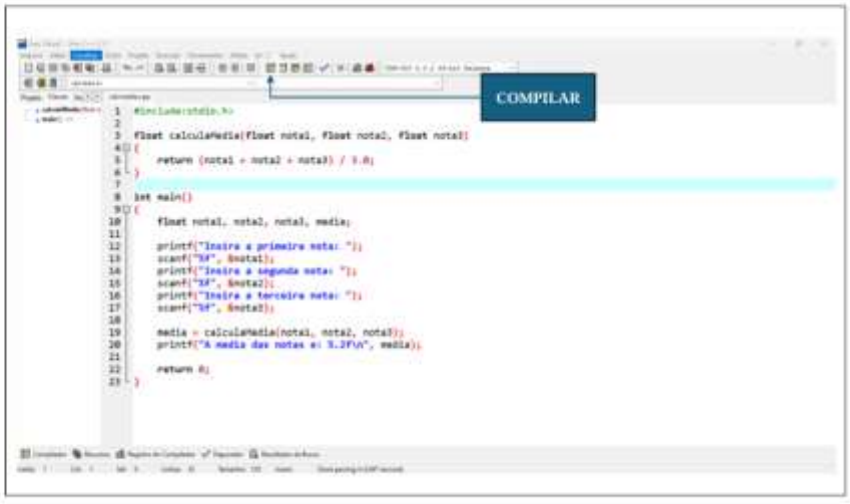
Passo 7 – Vamos declarar a função float calculaMedia(float nota1, float nota2, float nota3) antes da função main()

Passo 8 – Vamos agora escrever o resto do programa utilizando a função main()

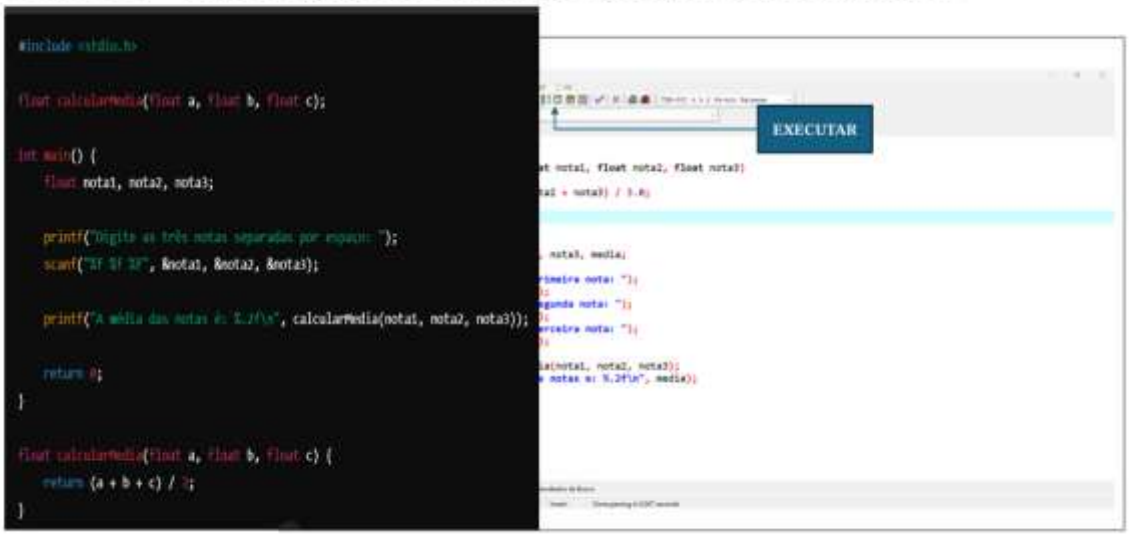


```
#include<stdio.h>

float calculaMedia(float nota1, float nota2, float nota3)
{
    return (nota1 + nota2 + nota3) / 3.0;
}

int main()
{
    float nota1, nota2, nota3, media;

    printf("Insira a primeira nota: ");
    scanf("%f", &nota1);
    printf("Insira a segunda nota: ");
    scanf("%f", &nota2);
    printf("Insira a terceira nota: ");
    scanf("%f", &nota3);

    media = calculaMedia(nota1, nota2, nota3);
    printf("A media das notas e: %.2f\n", media);

    return 0;
}
```

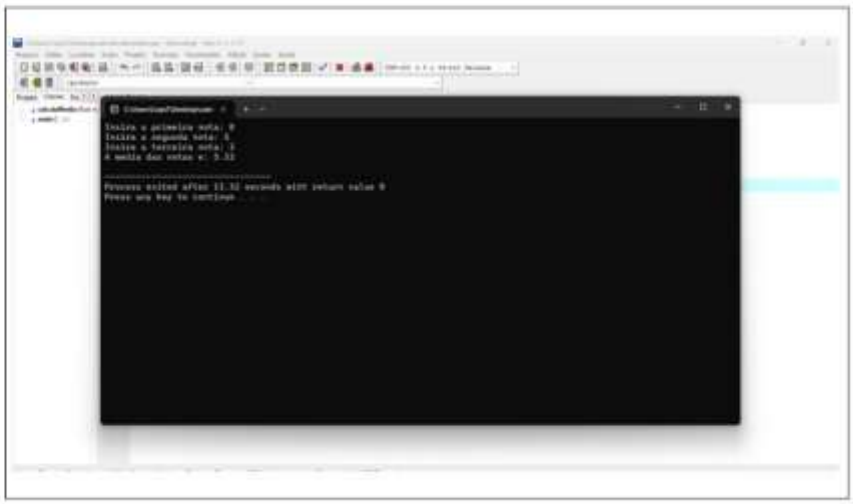
Passo 9 – Vamos agora compilar o programa para verificarmos se existem erros no código



Passo 10 – Vamos agora executar o programa e ver o resultado



Passo 10 – Vamos agora executar e ver a nossa consola

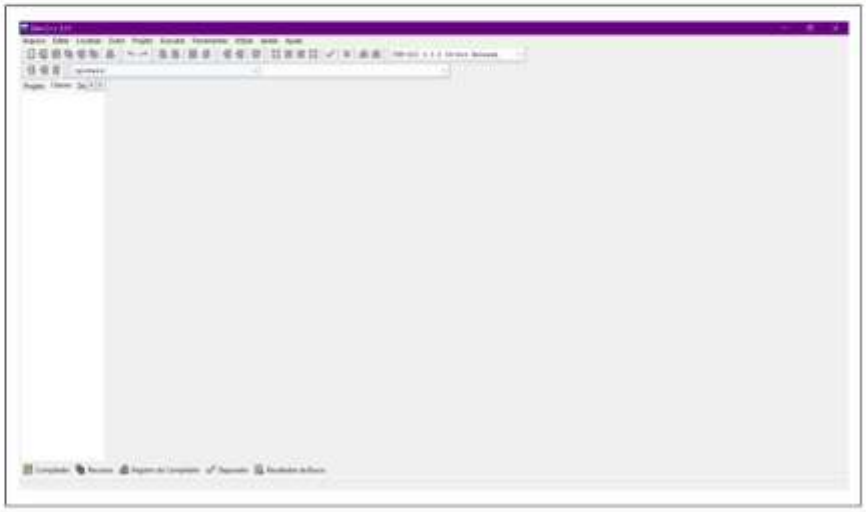


FIM DO EXERCICIO 2 CONDUZIDO STEP BY STEP

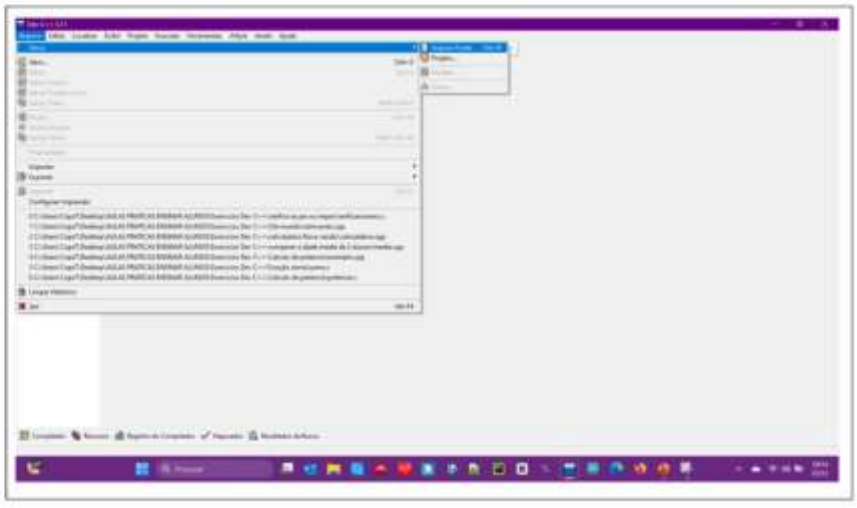
EXERCICIO 3 CONDUZIDO STEP BY STEP

- Desenvolva um programa que calcule a média de três ponderações diferentes sendo que o trabalho vale (25%), a ficha de trabalho (15%) e um teste (60%).
- Crie uma função com o nome calculaMedia() e que retorne o valor da média. O programa deve exibir a média no ecrã.

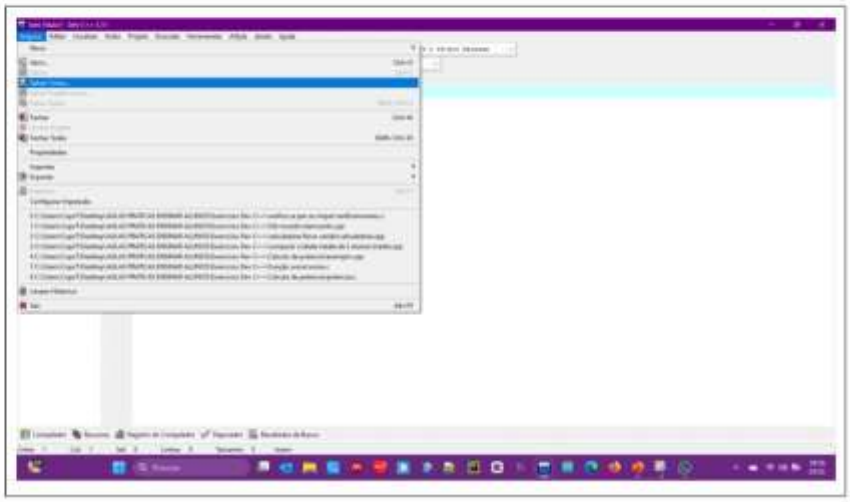
Passo 1 – Abrir o program Dev C++



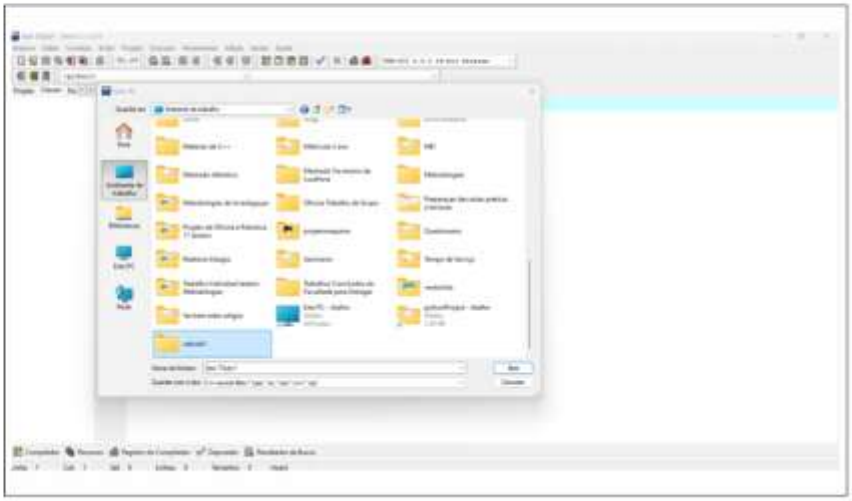
Passo 2 – Vamos a arquivo – Novo – Arquivo Fonte



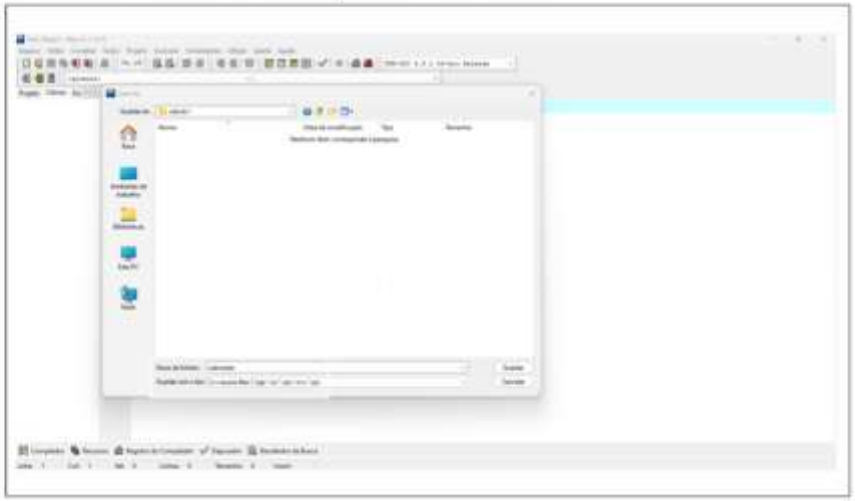
Passo 3 – Vamos agora salvar o ficheiro e vamos a salvar como…



Passo 4 – Vamos agora salvar o ficheiro com o nome "calcmedia" numa pasta no ambiente de trabalho dentro da pasta "calculo1"

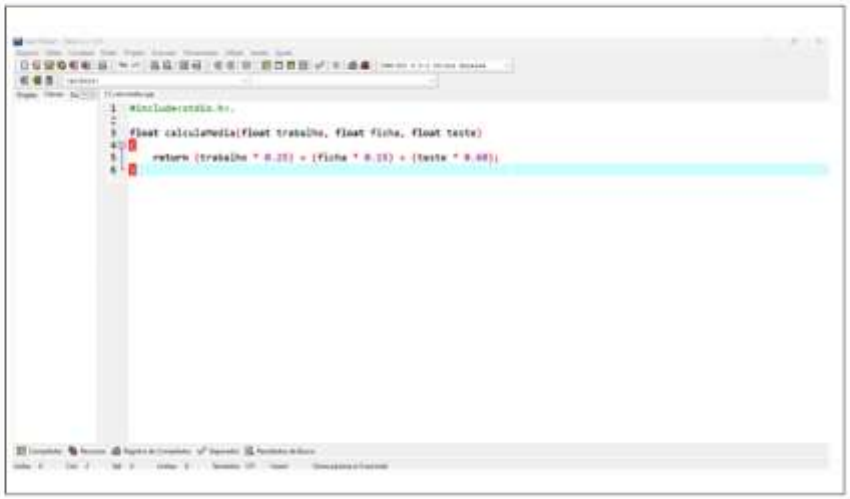


Passo 5 – Vamos agora salvar o ficheiro com o nome "calcmedia" numa pasta no ambiente de trabalho dentro da pasta "calculo1"



Passo 6 – No topo do programa do Dev C++ linha 1 devemos incluir a biblioteca - #include<stdio.h>

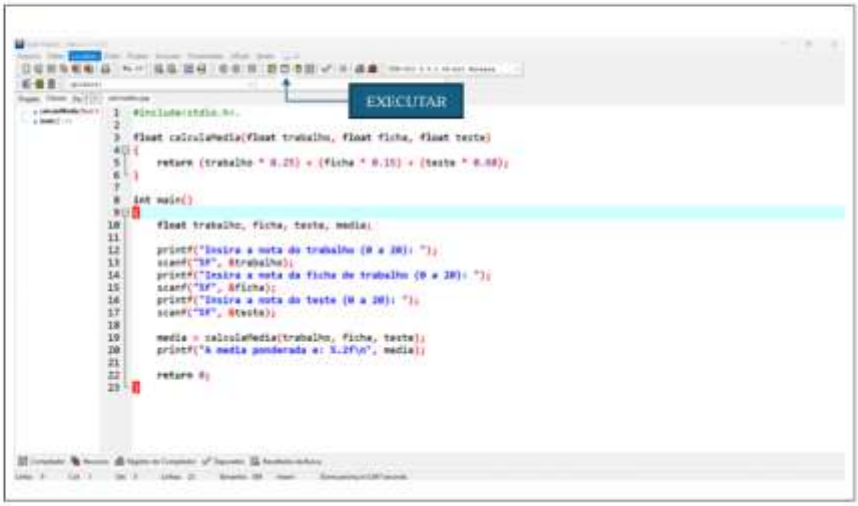
Passo 7 – Vamos declarar a função float calculaMedia(float trabalho, float ficha, float teste) antes da função main()



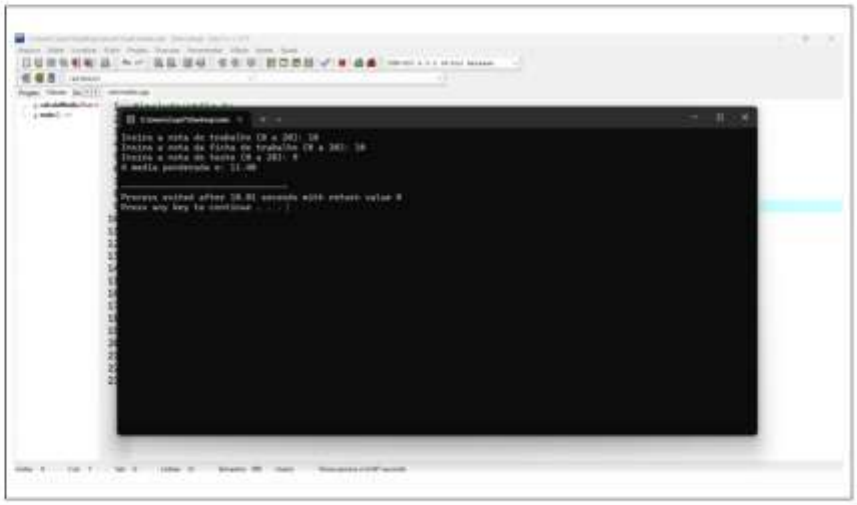
Passo 8 – Vamos agora escrever o resto do programa utilizando a função main()

Passo 9 – Vamos agora compilar o programa para verificarmos se existem erros no código

Passo 10 – Vamos agora executar o programa e ver o resultado



Passo 11 – Vamos agora executar e ver a nossa consola



FIM DA AULA 2 e 3

ESCOLA SECUNDÁRIA DE SANTO ANDRÉ
CURSO PROFISSIONAL TÉCNICO DE INFORMÁTICA - SISTEMAS

AULA 4 e 5

Conteúdos Específicos
1 – Exploração das Funções com Recurso à Inteligência Artificial • Utilizar ferramentas de IA para criar uma simulação de uma calculadora em C. 2 - Desconstrução da Solução • Compreender os elementos do programa selecionado pelos alunos.

ESCOLA SECUNDÁRIA DE SANTO ANDRÉ
CURSO PROFISSIONAL TÉCNICO DE INFORMÁTICA - SISTEMAS

Sumário da aula 4 e 5

• Desenvolvimento de um programa em C com o objetivo de criar uma calculadora em C utilizando as 4 principais operações básicas: A Soma, Subtração, Multiplicação e Divisão.
• Criação de funções individuais para cada operação.
• Implementação de um menu interativo com o objetivo de utilizar as estruturas de controlo (switch-case).

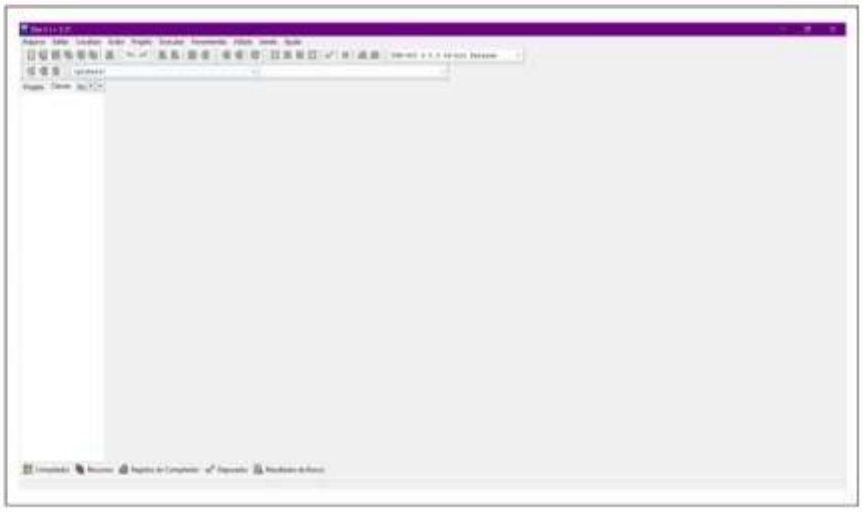
INICIO DO EXERCICIO – CRIAÇÃO DE UMA CALCULADORA

1. Criar uma calculadora com as operações básicas:
   - Soma.
   - Subtração.
   - Multiplicação.
   - Divisão.
2. Criar um menu para o utilizador poder escolher a operação desejada.

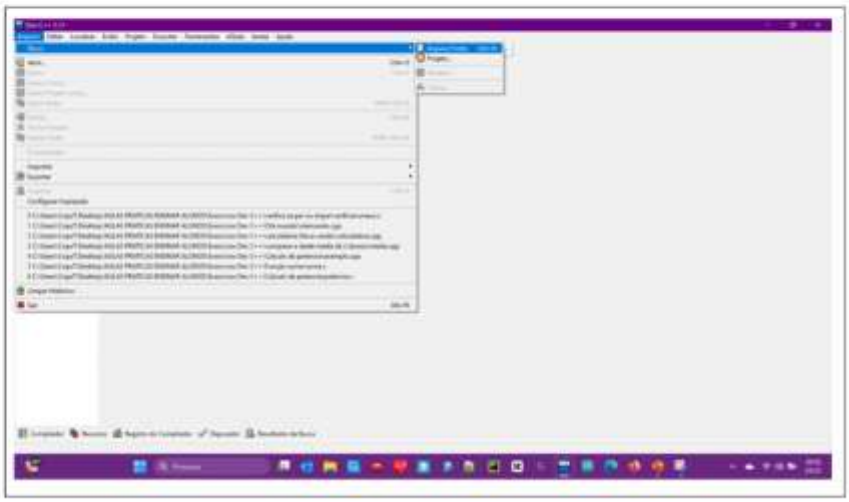
Regras:

- Utilizar funções individuais para cada uma das operações.
- Utilizar o Switch-case.

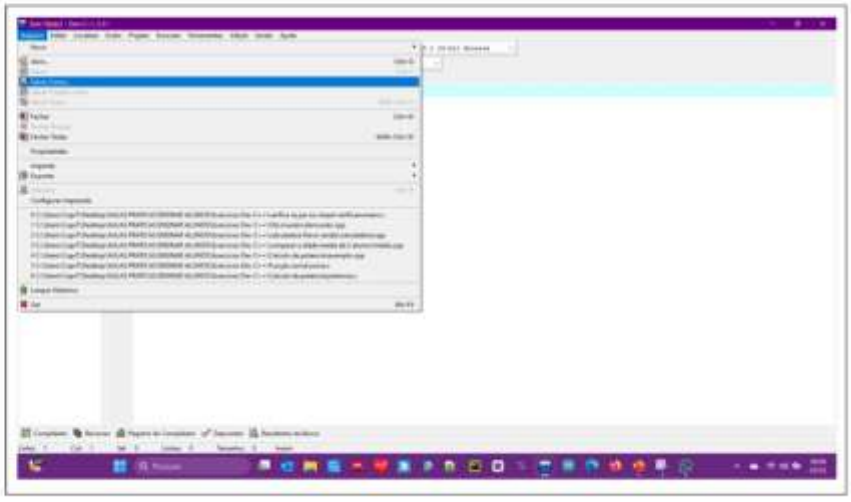
Passo 1 – Abrir o program Dev C++



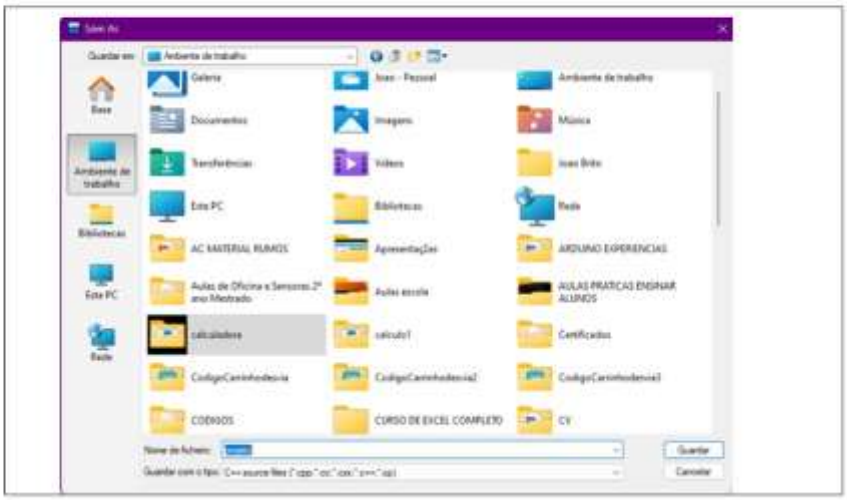
Passo 2 – Vamos a arquivo – Novo – Arquivo Fonte



Passo 3 – Vamos agora salvar o ficheiro e vamos a salvar como…



Passo 4 – Vamos agora salvar o ficheiro com o nome "projeto" numa pasta no ambiente de trabalho chamado "calculadora"



Passo 5 – No topo do programa do Dev C++ linha 1 devemos incluir a biblioteca - **#include<stdio.h>**

Vamos incluir no programa as 4 funções

4. Agora vamos definir as funções para as nossas 4 operações:

- Operação da soma:

```
1  #include <stdio.h>
2
3  int soma(int a, int b) {
4      return a + b;
5  }
```

- Operação da subtração:

```
7  int subtracao(int a, int b) {
8      return a - b;
9  }
```

- Operação da multiplicação:

```
10
11 int multiplicacao(int a, int b) {
12     return a * b;
13 }
```

- Operação da divisão:

```
16 float divisao(int a, int b)
17 {
18         return (a/b);
19 }
```

Passo 6 – Vamos declarar as 4 funções para as 4 operações: Soma, Subtração, Divisão e multiplicação

Calculadora.cpp

```
1  #include <stdio.h>
2  #include <locale.h>
3
4  int Soma(int a, int b)
5  {
6          return a + b;
7  }
8  int Subtracao(int a, int b)
9  {
10         return a - b;
11 }
12 int Multiplicacao(int a, int b)
13 {
14         return a * b;
15 }
16 float divisao(int a, int b)
17 {
18         return (a/b);
19 }
```

DECLARAÇÃO DA FUNÇÃO MAIN() e DO SWITCH

5. Agora vamos declarar a função main():

```
24  int main()
25  {
26      int opcao, a, b;
27  }
```

6. Agora vamos criar o menu com as opções de seleção para o utilizador:

7. Vamos agora escrever uma estrutura de controlo Switch-case tendo em conta que são 4 casos uma vez que são 4 operações.

```
switch (opcao)
{
        case 1:
            //Soma
        printf("Digite dois numeros:");
        scanf("%d %d", &a, &b);
        printf("Resultado da soma:%d\n", Soma(a, b));
        break;

case 2:
    // Subtração
    printf("Digite dois numeros: ");
    scanf("%d %d", &a, &b);
    printf("Resultado da subtracao: %d\n", subtracao(a, b));
    break;
```

EXERCICIO STEP BY STEP

```
case 3:
    // Multiplicação
    printf("Digite dois numeros: ");
    scanf("%d %d", &a, &b);
    printf("Resultado da multiplicacao: %d\n", multiplicacao(a, b));
    break;


// divisão
case 4:
printf("Digite dois numeros:");
scanf("%f %f", &a, &b);
if (b!=0)
printf("Resultado da divisao:%.2f\n", divisao(a, b));
else
printf("Erro: não é possivel dividir por zero");
break;
```

8. Vamos ainda incluir uma mensagem de erro para advertir o utilizador caso ele coloque outro número:

```
default:
    // Opção inválida
    printf("Opcao invalida!\n");
```

9. Encerramos o nosso programa:

```
return 0;
```

10. Por último iremos compilar o nosso código para verificar se tem erros.

Passo 8 – Vamos agora escrever o resto do programa declarando a função main() incluindo o switch e o return

[*] Calculadora.cpp

```c
#include <stdio.h>
#include <locale.h>

int Soma(int a, int b)
{
        return a + b;
}
int Subtracao(int a, int b)
{
        return a - b;
}
int Multiplicacao(int a, int b)
{
        return a * b;
}
float divisao(int a, int b)
{
        return (a/b);
}
```

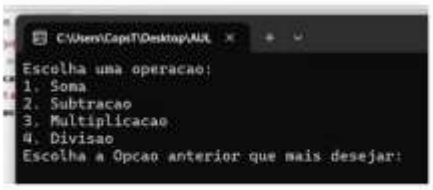
```c
int main()
{
        setlocale(LC_ALL, "pt_PT.UTF-8");

        int opcao, a, b;

        printf("Escolha uma operacao:\n");
        printf("1. Soma\n 2. Subtracao\n 3. Multiplicacao\n 4. Divisao\n");
        printf("Opcao:");
        scanf("%d", &opcao);

        switch (opcao)
        {
                case 1:
                printf("Digite dois numeros:");
                scanf("%d %d", &a, &b);
                printf("Resultado da soma:%d\n", Soma(a, b));
                break;

                case 2:
                printf("Digite dois numeros:");
                scanf("%d %d", &a, &b);
                printf("Resultado da subtracao:%d\n", Subtracao(a, b));
                break;
```

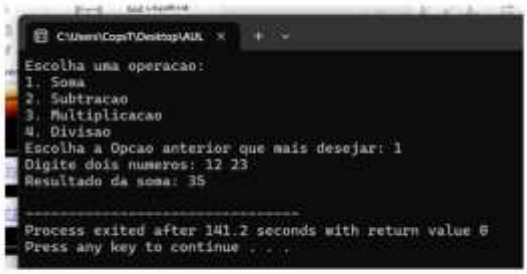
Passo 8 – Vamos agora escrever o resto do programa declarando a função main() incluindo o switch e o return – Continuação

```c
                case 3:
                printf("Digite dois numeros:");
                scanf("%d %d", &a, &b);
                printf("Resultado da multiplicacao:%d\n", Multiplicacao(a, b));
                break;
                // divisão
                case 4:
                printf("Digite dois numeros:");
                scanf("%f %f", &a, &b);
                if (b!=0)
                printf("Resultado da divisao:%.2f\n", divisao(a, b));
                else
                printf("Erro: não é possivel dividir por zero");
                break;

                default:
                        printf("Opcao invalida!\n");

        }
return 0;

}
```

EXERCICIO STEP BY STEP

11. Vamos agora salvar o nosso programa numa pasta chamada "Exercício step by step" e chamamos ao nosso ficheiro "calculadora".
12. Vamos executar o nosso código para que ele abra a seguinte janela:





FIM DA AULA 4 e 5

ESCOLA SECUNDÁRIA DE SANTO ANDRÉ
CURSO PROFISSIONAL TÉCNICO DE INFORMÁTICA - SISTEMAS

AULA 6 e 7

Conteúdos Específicos
• Prática computacional no ambiente de programação Dev C++. • Resolução de uma ficha formativa para aplicação de conceitos abordados nas aulas anteriores relativos a funções.

ESCOLA SECUNDÁRIA DE SANTO ANDRÉ
CURSO PROFISSIONAL TÉCNICO DE INFORMÁTICA - SISTEMAS

Sumário da aula 6 e 7

• Resolução de uma ficha formativa no ambiente de programação **Dev-C++** para aplicação prática dos conceitos abordados nas aulas anteriores.

CORREÇÃO DA FICHA FORMATIVA

Questão 1

a) A programação modular é uma forma de dividir o programa em partes menores chamados módulos ou subprogramas. Cada módulo é responsável por realizar uma tarefa especifica dentro do programa de forma a desenvolver, testar e manter o código de forma independente. É através da programação modular que é possível organizar o código, torná-lo mais legível de forma a facilitar a leitura e também reutilizá-lo sempre que necessário.

b) Legibilidade, Reutilização de código e facilidade na organização/manutenção do código.

- Legibilidade - O código fica mais organizado e fácil de compreender porque cada módulo possui uma função bem definida.
- Reutilização do código - Os módulos quando são bem definidos e construídos podem ser utilizados noutros programas ou em diferentes partes do mesmo programa.
- Facilidade na manutenção – Os módulos podem ser mais facilmente atualizados ou substituídos sem modificar todo o código já escrito.

CORREÇÃO DA FICHA FORMATIVA

Questão 2

a) Uma função em linguagem C refere-se a um bloco de código independente dentro de um programa que executa uma tarefa especifica. A função retorna sempre um valor (return).

b) Uma função com retorno é uma função que executa uma tarefa que retorna sempre um valor ao programa. O tipo de retorno poderá ser um número inteiro ou um número flutuante entre outros. São utilizadas quando necessitamos de obter um resultado. Exemplo: As funções sem retorno são funções que executam tarefas mas que não retornam valores, como por exemplo a função void.

```
4  int soma(int a, int b) {
5      return a + b;
6  }
```

```
3  void exibirMensagem()
4  {
5      printf("Ola, mundo!\n");
6  }
```

CORREÇÃO DA FICHA FORMATIVA

Questão 2

c) Estrutura básica de uma função em linguagem C

```
<tipo de dados primitivo e nome da função> (lista de argumentos)
{
        return <valor>;
}
```

d) Os argumentos de uma função são os valores atribuídos à função quando ela é chamada. Neste exemplo os argumentos da função são os valores atribuídos a "a" e "b".

```
[*] Sem Título1
1  // Função que calcula a soma
2  int soma(int a, int b) {  //
3      return a + b;         //
4  }
```

CORREÇÃO DA FICHA FORMATIVA

Questão 2

e) A estrutura de um programa em C é constituída pelas diretivas ao pré-processador, as declarações globais, a função main() e a declaração de funções:

CORREÇÃO DA FICHA FORMATIVA

Questão 3

Função sem retorno que verifica se um determinado número à escolha do utilizador é par ou ímpar:

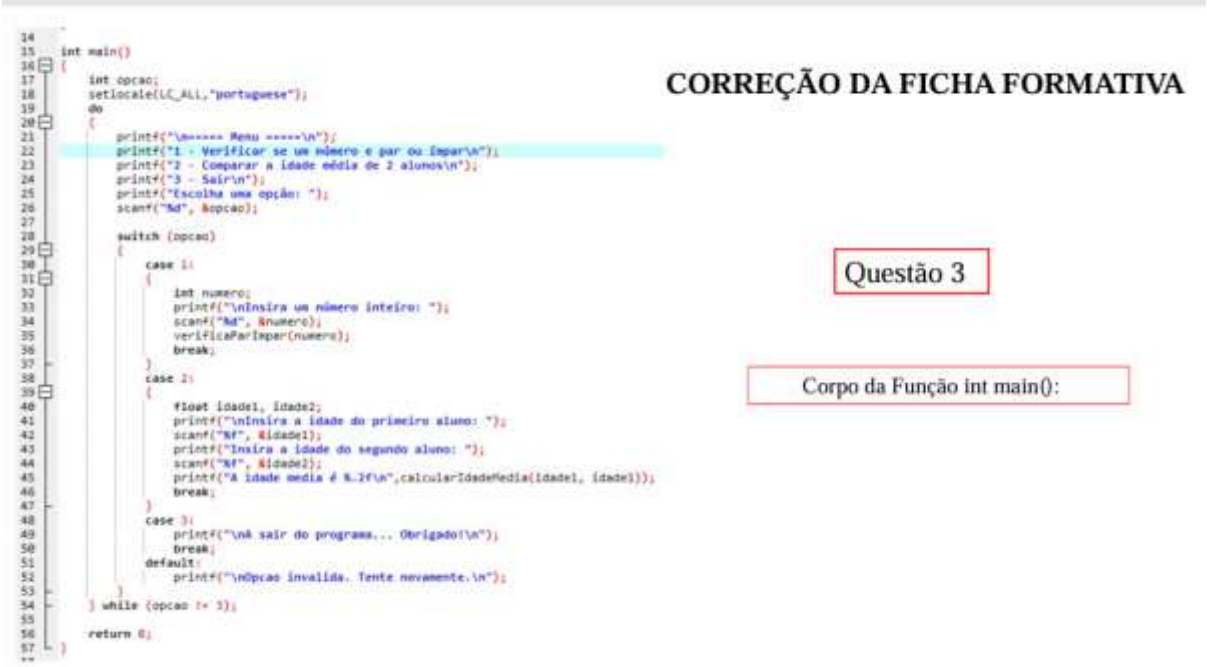
```
3   void verificaParImpar(int numero)
4   {
5       if (numero % 2 == 0)
6           printf("O número %d é par.\n", numero);
7       else
8           printf("O número %d é impar.\n", numero);
9   }
```

CORREÇÃO DA FICHA FORMATIVA

Questão 3

Função com retorno que compare a idade média de 2 alunos:

```
10  float calcularIdadeMedia(float idade1, float idade2)
11  {
12      return (idade1 + idade2) / 2.0;
13  }
```

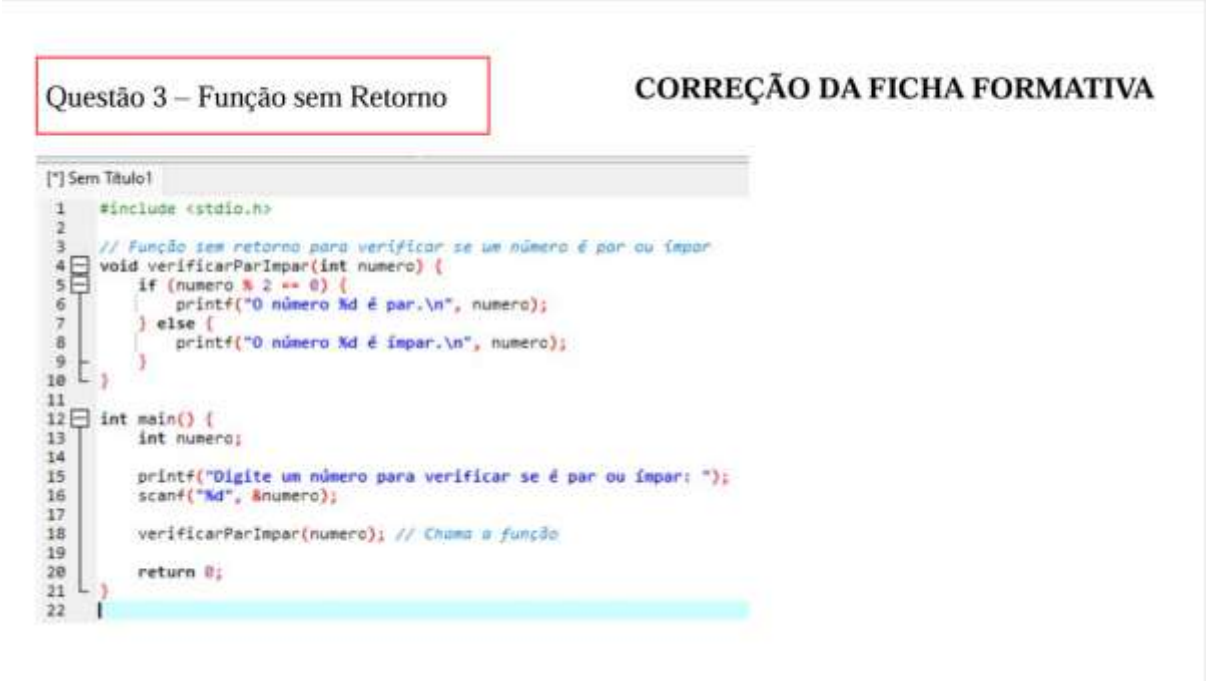


```c
int main()
{
    int opcao;
    setlocale(LC_ALL,"portuguese");
    do
    {
        printf("\n====== Menu =====\n");
        printf("1 - Verificar se um número é par ou ímpar\n");
        printf("2 - Comparar a idade média de 2 alunos\n");
        printf("3 - Sair\n");
        printf("Escolha uma opção: ");
        scanf("%d", &opcao);

        switch (opcao)
        {
            case 1:
            {
                int numero;
                printf("\nInsira um número inteiro: ");
                scanf("%d", &numero);
                verificaParImpar(numero);
                break;
            }
            case 2:
            {
                float idade1, idade2;
                printf("\nInsira a idade do primeiro aluno: ");
                scanf("%f", &idade1);
                printf("Insira a idade do segundo aluno: ");
                scanf("%f", &idade2);
                printf("A idade media é %.2f\n",calcularIdadeMedia(idade1, idade1));
                break;
            }
            case 3:
                printf("\nA sair do programa... Obrigado!\n");
                break;
            default:
                printf("\nOpcao invalida. Tente novamente.\n");
        }
    } while (opcao != 3);

    return 0;
}
```

CORREÇÃO DA FICHA FORMATIVA

Questão 3

Corpo da Função int main():



Questão 3 – Função sem Retorno

CORREÇÃO DA FICHA FORMATIVA

[*] Sem Título1

```c
#include <stdio.h>

// Função sem retorno para verificar se um número é par ou ímpar
void verificarParImpar(int numero) {
    if (numero % 2 == 0) {
        printf("O número %d é par.\n", numero);
    } else {
        printf("O número %d é ímpar.\n", numero);
    }
}

int main() {
    int numero;

    printf("Digite um número para verificar se é par ou ímpar: ");
    scanf("%d", &numero);

    verificarParImpar(numero); // Chama a função

    return 0;
}
```

Questão 3 – Função com Retorno

CORREÇÃO DA FICHA FORMATIVA

[*] Sem Título1

```c
#include <stdio.h>

// Função com retorno para calcular a idade média entre dois alunos
float calcularIdadeMedia(int idade1, int idade2) {
    return (idade1 + idade2) / 2.0;
}

int main() {
    int idade1, idade2;
    float media;

    printf("Digite a idade do primeiro aluno: ");
    scanf("%d", &idade1);

    printf("Digite a idade do segundo aluno: ");
    scanf("%d", &idade2);

    // Chama a função calcularIdadeMedia e armazena o retorno
    media = calcularIdadeMedia(idade1, idade2);

    printf("A idade média dos dois alunos é: %.2f\n", media);

    return 0;
}
```

FIM DA AULA 6 e 7

ESCOLA SECUNDÁRIA DE SANTO ANDRÉ
CURSO PROFISSIONAL TÉCNICO DE INFORMÁTICA - SISTEMAS

Sumário da aula 8

•Aplicação de uma ficha de avaliação para consolidar os conteúdos abordados.
•Resumo dos conceitos principais.
.

AVALIAÇÃO

- Ficha de avaliação de aprendizagens.
- Duração (50 minutos).

CORREÇÃO DA FICHA DE AVALIAÇÃO

GRUPO 1

1 Qual das seguintes opções define programação modular? 10 pont

- ◯ Uma técnica para escrever programas sem a utilização de funções.
- ◯ Uma forma de criar programas sem erros.
- ◯ Uma técnica usada para programar apenas em linguagens orientadas a objetos.
- ◯ Uma técnica que divide um programa em partes menores, chamados módulos ou subprogramas.

Resposta:

CORREÇÃO DA FICHA DE AVALIAÇÃO

GRUPO 1

2 Qual das seguintes opções ***NÃO*** é uma vantagem da programação modular?

- ◯ Facilitar a manutenção do código.
- ◯ Aumentar a redundância do código.
- ◯ Permitir o trabalho de equipa.
- ◯ Reutilizar código noutros outros projetos.

Resposta:

CORREÇÃO DA FICHA DE AVALIAÇÃO

GRUPO 1

3. Qual das seguintes instruções é considerada uma ***função de "Output"***?

- ◯ **printf().**
- ◯ **main().**
- ◯ **start().**
- ◯ **scanf().**

Resposta:

CORREÇÃO DA FICHA DE AVALIAÇÃO

GRUPO 1

4. Qual a etapa em que a solução é traduzida para ***linguagem máquina***?

- ◯ Edição.
- ◯ Execução.
- ◯ Compilação.
- ◯ Pré-processamento.

Resposta:

CORREÇÃO DA FICHA DE AVALIAÇÃO

GRUPO 1

5. O que acontece durante o ***pré-processamento***? 10 pontos

- ◯ Escrevemos o código num editor de texto.
- ◯ O código é preparado para compilação, analisando diretivas como **#include**.
- ◯ O programa é testado no ambiente de execução.
- ◯ As funções são otimizadas.

Resposta:

CORREÇÃO DA FICHA DE AVALIAÇÃO

GRUPO 1

6. Qual das seguintes afirmações sobre as ***funções é verdadeira***?

- ◯ Um bloco de código que realiza uma tarefa específica.
- ◯ Uma **void** deve sempre retornar um valor.
- ◯ Todas as funções de um programa são executadas automaticamente.
- ◯ Uma função não pode chamar outra.

Resposta:

CORREÇÃO DA FICHA DE AVALIAÇÃO

GRUPO 1

7. Qual é a principal diferença entre uma ***função*** **void** e uma ***função*** com **return**?

- ◯ A **void** retorna valores, mas a com **return** não.
- ◯ A com **return** devolve valores, mas a **void** não.
- ◯ A **void** é sempre obrigatória em qualquer programa.
- ◯ A com **return** não pode ter argumentos.

Resposta:

CORREÇÃO DA FICHA DE AVALIAÇÃO

GRUPO 1

8. Qual das seguintes afirmações referente a variáveis locais é verdadeira?

- ◯ São declaradas dentro de uma função e só podem ser usadas pela mesma.
- ◯ São declaradas fora de qualquer função.
- ◯ Podem ser usadas em qualquer parte do programa.
- ◯ São inicializadas automaticamente com o valor 0.

Resposta:

CORREÇÃO DA FICHA DE AVALIAÇÃO

GRUPO 1

9. O que são ***argumentos*** de uma função?

- ◯ Instruções dentro do corpo da função.
- ◯ Variáveis locais usadas pelo programa.
- ◯ Variáveis que recebem valores e que são enviados para a função.
- ◯ Diretivas ao pré-processador incluídas no código.

Resposta:

CORREÇÃO DA FICHA DE AVALIAÇÃO

GRUPO 1

10. Qual das seguintes opções é uma **vantagem** da programação modular?

- ◯ Permite executar várias funções simultaneamente.
- ◯ Organiza o código e evitam a redundância.
- ◯ Complica a solução.
- ◯ Torna desnecessário a utilização da função main().

Resposta:

CORREÇÃO DA FICHA DE AVALIAÇÃO

GRUPO 1

11. Qual das seguintes opções relativamente a tipo de dados primitivos ***é verdadeira***?

- ( ) O ***int*** contêm números reais.
- ( ) O ***float*** só contêm números inteiro.
- ( ) O ***double*** admite valores reais com precisão.
- ( ) O ***char*** admite dados lógicos.

Resposta:

CORREÇÃO DA FICHA DE AVALIAÇÃO

GRUPO 1

12. Qual das seguintes afirmações relativamente a ***funções é verdadeira***?

- ( ) Devolve mais do que um resultado em simultâneo.
- ( ) É sempre mais complexa.
- ( ) Permite realizar uma tarefa especifica com retorno de um valor.
- ( ) Não necessita de cabeçalho.

Resposta:

CORREÇÃO DA FICHA DE AVALIAÇÃO

GRUPO 1

13. Qual das seguintes afirmações relativamente à etapa da compilação é **verdadeira?**

- ◯ Corrige automaticamente todos os erros do programa.
- ◯ Apenas traduz o código fonte para o código máquina.
- ◯ Identifica erros de sintaxe e de semântica no código fonte.
- ◯ Dá instruções ao pré-processador relativamente ao **#include**.

Resposta:

CORREÇÃO DA FICHA DE AVALIAÇÃO

GRUPO 1

14. Qual das seguintes afirmações **relativamente a subprogramas é verdadeira?**

- ◯ Não obedece às regras da programação estruturada.
- ◯ Pode ou não retornar valores.
- ◯ É um bloco de código que apenas serve para fazer inputs e outputs.
- ◯ Nunca precisa de ser chamado para ser executado.

Resposta:

CORREÇÃO DA FICHA DE AVALIAÇÃO

GRUPO 1

15. Qual das seguintes opções contém instruções associadas à biblioteca **stdio.h**?

- ( ) while.
- ( ) scanf() e printf().
- ( ) sqrt().
- ( ) round().

Resposta:

CORREÇÃO DA FICHA DE AVALIAÇÃO

GRUPO 2

1. Qual o nome da função criada pelo programador no presente código?

Resposta:

```c
# include <stdio.h>

int soma(int a, int b)
{
        return a + b;
}

int main()
{
        printf("%d\n", soma(3,5));
        return 0;
}
```

CORREÇÃO DA FICHA DE AVALIAÇÃO

GRUPO 2

2. Qual o papel da instrução **return a + b**?

_____

_____

_____

Resposta:

```c
# include <stdio.h>

int soma(int a, int b)
{
        return a + b;
}

int main()
{
        printf("%d\n", soma(3,5));
        return 0;
}
```

CORREÇÃO DA FICHA DE AVALIAÇÃO

GRUPO 2

3. Na função main() qual o valor devolvido pela função soma?

_____

_____

_____

Resposta:

```c
# include <stdio.h>

int soma(int a, int b)
{
        return a + b;
}

int main()
{
        printf("%d\n", soma(3,5));
        return 0;
}
```

CORREÇÃO DA FICHA DE AVALIAÇÃO

GRUPO 2

4. Quais são as variáveis declaradas no código acima?

_____

_____

Resposta:

```c
#include <stdio.h>

float CalculaMedia (float nota1, float nota2, float nota3)
{
	return (nota1 + nota2 + nota3) /3.0;
}

int main()
{
	float nota1, nota2, nota3, media;

	printf("Insira a nota 1:");
	scanf("%f", &nota1);

	printf("Insira a nota 2:");
	scanf("%f", &nota2);

	printf("Insira a nota 3:");
	scanf("%f", &nota3);

	media = CalculaMedia(nota1, nota2, nota3);
	printf("A media das 3 notas é %.2f", media);

	return 0;
}
```

CORREÇÃO DA FICHA DE AVALIAÇÃO

GRUPO 2

5. Porque motivo neste código se declaram variáveis com o tipo primitivo **float**?

_____

_____

_____

Resposta:


```c
#include <stdio.h>

float CalculaMedia (float nota1, float nota2, float nota3)
{
	return (nota1 + nota2 + nota3) /3.0;
}

int main()
{
	float nota1, nota2, nota3, media;

	printf("Insira a nota 1:");
	scanf("%f", &nota1);

	printf("Insira a nota 2:");
	scanf("%f", &nota2);

	printf("Insira a nota 3:");
	scanf("%f", &nota3);

	media = CalculaMedia(nota1, nota2, nota3);
	printf("A media das 3 notas é %.2f", media);

	return 0;
}
```

CORREÇÃO DA FICHA DE AVALIAÇÃO

GRUPO 2

6. Quantos valores recebe a *função*? Justifique a sua resposta.

Resposta:

```c
#include <stdio.h>

float CalculaMedia (float nota1, float nota2, float nota3)
{
        return (nota1 + nota2 + nota3) /3.0;
}

int main()
{
        float nota1, nota2, nota3, media;

        printf("Insira a nota 1:");
        scanf("%f", &nota1);

        printf("Insira a nota 2:");
        scanf("%f", &nota2);

        printf("Insira a nota 3:");
        scanf("%f", &nota3);

        media = CalculaMedia(nota1, nota2, nota3);
        printf("A media das 3 notas é %.2f", media);

        return 0;
}
```

RESUMO

- **Programação Modular:**
  - Divisão de programas em módulos ou subprogramas para melhorar a organização e reutilização de código.
- **Vantagens:**
  - Legibilidade.
  - Reutilização de código.
  - Facilidade na depuração e manutenção.
  - Trabalho em equipa eficiente.

- **Conceito de Função:**
  - Bloco de código reutilizável que realiza uma tarefa específica.
  - Central na programação modular, permite agrupar instruções e dividir o programa em partes menores.

RESUMO

Estrutura de uma Função:

- **Cabeçalho:** Define o tipo de retorno, nome da função e argumentos.
- **Corpo:** Contém as instruções que a função executa.

- **Exemplo:**

```c
int soma(int a, int b) {
    return a + b;
}
```

- **Tipos de Funções:**
  - **Com retorno**: Retornam um valor para o programa principal.
  - **Void:** Não retornam valor, mas realizam ações (ex.: exibem mensagens).
- **Argumentos:**
  - **Variáveis** locais da função que recebem valores na chamada.

RESUMO

- **Ciclo de Desenvolvimento de um Programa:**
  - **Etapas:**
    - Edição: Escrever o código.
    - Pré-processamento: Preparar o código para compilação.
    - Compilação: Transformar código-fonte em linguagem de máquina.
    - Execução: Testar e validar o programa.

- **A Função main():**
  - Ponto de entrada de qualquer programa em C.

- Estrutura básica:

```c
int main() {
    printf("Olá, Mundo!\n");
    return 0;
}
```

RESUMO

- **Vantagens das Funções:**
  - Reaproveitamento e organização do código.
  - Evita repetição de blocos.
  - Facilita a leitura e separação de tarefas.

ESCOLA SECUNDÁRIA DE SANTO ANDRÉ
CURSO PROFISSIONAL TÉCNICO DE INFORMÁTICA - SISTEMAS

Fontes Bibliográficas

- Damas, L. (2019). *Linguagem C* (24ª ed.). Edições FCA.
- Deitel, H. M., & Deitel, P. J. (2015). *C: How to program* (8ª ed.). Pearson Education.
- Kernighan, B. W., & Ritchie, D. M. (1988). *The C programming language* (2ª ed.). Prentice Hall.
- King, K. N. (2008). *C programming: A modern approach* (2ª ed.). W. W. Norton & Company.
- Liberty, J. (2014). *Programming in C: A complete introduction to the C programming language*. Sams Publishing.
- Ponti Jr., M. P. (2011). *Uma breve introdução à criação de bibliotecas e makefiles em C/C++*. Universidade de São Paulo, Instituto de Ciências Matemáticas e de Computação.
- Sentance, S., Barendsen, E., Howard, N. R., & Schulte, C. (2023). *Computer science education: Perspectives on teaching and learning in school*. Bloomsbury Academic.

Fichas aplicadas durante as aulas

Escola Secundária de Santo André

CURSO PROFISSIONAL DE TÉCNICO DE SISTEMAS - INFORMÁTICA

Programação/UFCD 0784 **Ficha de Trabalho – 1.ºL**

20/11/2024 O Professor Estagiário: João Brito

Introdução

O tema que estamos a trabalhar tem como objetivo identificar e criar funções na linguagem C. Pretende-se que os alunos compreendam a noção de subprogramas bem como que com a sua utilização é possível aplicar os princípios da programação estruturada e reescrever códigos de uma forma mais simples, ou seja através da divisão de problemas em componentes simples como meio de solução para resolver problemas mais complexos.

Objetivos Gerais

- Adquirir a noção de subprograma;
- Conhecer as regras de declaração de subprogramas;
- Conhecer as regras de execução de subprogramas;
- Utilizar corretamente parâmetros;
- Distinguir os diferentes tipos de subprogramas;
- Conhecer os mecanismos de utilização de bibliotecas de subprogramas.

Objetivos Específicos

- Conceber programas através da divisão de problemas
- Estimular o raciocínio lógico e o pensamento computacional
- Saber escolher e adequar as soluções tecnológicas aos problemas a resolver
- Estimular a reflexão, a observação e autonomia

1

Tarefas

- Ler, interpretar e responder a um conjunto de questões teóricas acerca de programação modular e funções em linguagem C.
- Leitura, interpretação e análise de um problema.
- Codificação da solução na linguagem de programação C, através da utilização do software de desenvolvimento Dev C++, devidamente indentada e comentada.

Enunciado da Tarefa

1. **Conceitos Básicos:**
   a) Diga o que entende por programação modular.
   b) Liste três vantagens de utilizar a programação modular.
2. **Funções:**
   a) Defina o conceito de função em linguagem C.
   b) Explique a diferença entre: função com retorno e função sem retorno.
   c) Escreva a estrutura básica de uma função em linguagem C.
   d) Diga o que entende por argumentos de função. Pode recorrer a um exemplo concreto durante a sua explicação.
   e) Indique qual a estrutura de um programa em linguagem C.
3. **Exercícios práticos recorrendo ao programa do Dev C++:**
   Crie um programa em linguagem C que permita ao utilizador optar por:
   - Função sem retorno que verifica se um determinado número à escolha do utilizador é par ou ímpar.
   - Função com retorno que calcule a idade media entre 2 alunos.

   Nota: O programa **termina** quando o utilizador digitar o número **3.**
   Sugestão: Pode criar um pequeno menu para o utilizador escolher a opção que pretende.

Entrega do trabalho realizado

O trabalho final (documento word com as respostas às questões teóricas, código em) deve ser compactado (utilizar Winzip ou Winrar), o nome do ficheiro deve indicar a data, os

2

nomes dos elementos do grupo, a respetiva turma e disciplina (exemplo: 24-11-2024 jose_teixeira_e_bruno_caetano, 1.ºL, Programacao.zip).

O trabalho deverá ser publicado na tarefa criada na disciplina de Programação, na plataforma *Microsoft Teams* por todos os elementos do grupo até ao final da aula.

Duração da Tarefa

Uma sessão de 50 minutos + os primeiros 20 minutos de uma segunda sessão. Os últimos 20 minutos da segunda sessão serão para a correção da ficha de trabalho.

A tarefa deverá ser resolvida em grupos de dois ou três alunos, de acordo com a orientação do professor.

Avaliação da Tarefa

Esta tarefa tem carácter formativo

3

Escola Secundária de Santo André

CURSO PROFISSIONAL DE TÉCNICO DE SISTEMAS - INFORMÁTICA

Programação/UFCD 0784 **Correção – 1.ºL**

20/11/2024 O Professor Estagiário: João Brito

Introdução

O tema que estamos a trabalhar tem como objetivo identificar e criar funções na linguagem C. Pretende-se que os alunos compreendam a noção de subprogramas bem como que com a sua utilização é possível aplicar os princípios da programação estruturada e reescrever códigos de uma forma mais simples, ou seja através da divisão de problemas em componentes simples como meio de solução para resolver problemas mais complexos.

Objetivos Gerais

- Adquirir a noção de subprograma;
- Conhecer as regras de declaração de subprogramas;
- Conhecer as regras de execução de subprogramas;
- Utilizar corretamente parâmetros;
- Distinguir os diferentes tipos de subprogramas;
- Conhecer os mecanismos de utilização de bibliotecas de subprogramas.

Objetivos Específicos

- Conceber programas através da divisão de problemas
- Estimular o raciocínio lógico e o pensamento computacional
- Saber escolher e adequar as soluções tecnológicas aos problemas a resolver
- Estimular a reflexão, a observação e autonomia

1

Tarefas

- Ler, interpretar e responder a um conjunto de questões teóricas acerca de programação modular e funções em linguagem C.
- Leitura, interpretação e análise de um problema.
- Codificação da solução na linguagem de programação C, através da utilização do software de desenvolvimento Dev C++, devidamente indentada e comentada.

Enunciado da Tarefa

1. **Conceitos Básicos:**
   a) Diga o que entende por programação modular.
   b) Liste três vantagens de utilizar a programação modular.
2. **Funções:**
   a) Defina o conceito de função em linguagem C.
   b) Explique a diferença entre: função com retorno e função sem retorno.
   c) Escreva a estrutura básica de uma função em linguagem C.
   d) Diga o que entende por argumentos de função. Pode recorrer a um exemplo concreto durante a sua explicação.
   e) Indique qual a estrutura de um programa em linguagem C.
3. **Exercícios práticos recorrendo ao programa do Dev C++:**
   Crie um programa em linguagem C que permita ao utilizador optar por:
   - Função sem retorno que verifica se um determinado número à escolha do utilizador é par ou ímpar.
   - Função com retorno que calcule a idade media entre 2 alunos.

   Nota: O programa **termina** quando o utilizador digitar o número **3.**
   Sugestão: Pode criar um pequeno menu para o utilizador escolher a opção que pretende.

Entrega do trabalho realizado

O trabalho final (documento word com as respostas às questões teóricas, código em) deve ser compactado (utilizar Winzip ou Winrar), o nome do ficheiro deve indicar a data, os

2

nomes dos elementos do grupo, a respetiva turma e disciplina (exemplo: 24-11-2024 jose_teixeira_e_bruno_caetano, 1.ºL, Programacao.zip).

O trabalho deverá ser publicado na tarefa criada na disciplina de Programação, na plataforma *Microsoft Teams* por todos os elementos do grupo até ao final da aula.

Duração da Tarefa

Uma sessão de 50 minutos + os primeiros 20 minutos de uma segunda sessão. Os últimos 20 minutos da segunda sessão serão para a correção da ficha de trabalho.

A tarefa deverá ser resolvida em grupos de dois ou três alunos, de acordo com a orientação do professor.

Avaliação da Tarefa

Esta tarefa tem carácter formativo

Correção da Ficha

O que entende por programação modular?

A programação modular é uma técnica de desenvolvimento de software que divide um programa em pequenas partes ou módulos independentes e reutilizáveis, cada um responsável por uma funcionalidade específica.

Três vantagens de utilizar a programação modular:

- Facilita a manutenção do código.
- Promove a reutilização de código em diferentes partes do programa ou em projetos futuros.
- Simplifica o processo de depuração, uma vez que os módulos podem ser testados individualmente.

Defina o conceito de função em linguagem C:

Uma função em C é um bloco de código reutilizável que executa uma tarefa específica e pode ser chamada em qualquer parte do programa.

Diferença entre função com retorno e função sem retorno:

- Função **com retorno**: Retorna um valor para o programa que a chamou. Exemplo: int soma(int a, int b) { return a + b; }.
- Função **sem retorno**: Não retorna nenhum valor, apenas executa ações. Exemplo: void imprimirMensagem() { printf("Olá Mundo!"); }.

3

Estrutura básica de uma função em C:

c

Copiar código

```
tipo_de_retorno nome_da_funcao(tipo_de_parametro parametro) {
   // Corpo da função
}
```

O que entende por argumentos de função?

Os argumentos são valores passados para uma função quando ela é chamada, para que possam ser utilizados dentro dela.

Exemplo:

```
int soma(int a, int b) {
   return a + b;
}
```

Estrutura básica de um programa em C:

```
#include <stdio.h>

int main() {
   // Código principal
   return 0;
}
```

Resolução dos exercícios práticos:

```c
#include <stdio.h>

void verificarParImpar(int numero); // Função sem retorno
float calcularIdadeMedia(int idade1, int idade2); // Função com retorno

int main() {
    int opcao;

    do {
        // Menu de opções
        printf("\n=== Menu de Opções ===\n");
        printf("1. Verificar se um número é par ou ímpar (Função sem retorno)\n");
        printf("2. Calcular a idade média entre dois alunos (Função com retorno)\n");
        printf("3. Sair\n");
        printf("Escolha uma opção: ");
        scanf("%d", &opcao);
```

4

```c
        switch(opcao) {
            case 1: {
                int numero;
                printf("Digite um número: ");
                scanf("%d", &numero);
                verificarParImpar(numero); // Chama a função sem retorno
                break;
            }
            case 2: {
                int idade1, idade2;
                printf("Digite a idade do primeiro aluno: ");
                scanf("%d", &idade1);
                printf("Digite a idade do segundo aluno: ");
                scanf("%d", &idade2);
                float media = calcularIdadeMedia(idade1, idade2); // Chama a função com retorno
                printf("A idade média é: %.2f\n", media);
                break;
            }
            case 3:
                printf("Programa encerrado.\n");
                break;
            default:
                printf("Opção inválida! Tente novamente.\n");
        }
    } while(opcao != 3); // Sai do programa quando o utilizador digita 3

    return 0;
}

// Função sem retorno para verificar se um número é par ou ímpar
void verificarParImpar(int numero) {
    if (numero % 2 == 0) {
        printf("O número %d é par.\n", numero);
    } else {
        printf("O número %d é ímpar.\n", numero);
    }
}

// Função com retorno para calcular a idade média entre dois alunos
float calcularIdadeMedia(int idade1, int idade2) {
    return (idade1 + idade2) / 2.0;
}
```

5

	Escola Secundária de Santo André - sede **Grupo de Recrutamento 550 – Informática** Ficha de Avaliação– **Módulo 3 – Funções** **Duração: 50 minutos** **Os professores: Margarida Batista e João Brito** Ano letivo de 2024/ 2025 – DATA:xx/xx/xx	Curso: **TIS** Disciplina: **CBP** Turma **1.ºL**

Classificação: Valores

Nome do aluno: _____ Número: _____

A presente ficha de avaliação tem como objetivo aferir os conhecimentos teórico-práticos adquiridos relativamente a módulos ou subprogramas.

Duração da prova: 50 minutos **8** páginas

O teste é constituído por dois grupos:

Grupo 1: Questões de escolha múltipla que avaliam os conceitos teóricos.
Neste grupo para cada questão existe apenas uma resposta.

Grupo 2: Questões de interpretação e análise de código.

PÁGINA 1 DE 8

	Escola Secundária de Santo André - sede **Grupo de Recrutamento 550 – Informática** Ficha de Avaliação– **Módulo 3 – Funções** **Duração: 50 minutos** **Os professores: Margarida Batista e João Brito** Ano letivo de 2024/ 2025 – DATA:xx/xx/xx	Curso: **TIS** Disciplina: **CBP** Turma **1.ºL**

5. O que acontece durante o ***pré-processamento***? 10 pontos

- ◯ Escrevemos o código num editor de texto.
- ◯ O código é preparado para compilação, analisando diretivas como **#include**.
- ◯ O programa é testado no ambiente de execução.
- ◯ As funções são otimizadas.

6. Qual das seguintes afirmações sobre as ***funções é verdadeira***? 10 pontos

- ◯ Um bloco de código que realiza uma tarefa específica.
- ◯ Uma **void** deve sempre retornar um valor.
- ◯ Todas as funções de um programa são executadas automaticamente.
- ◯ Uma função não pode chamar outra.

7. Qual é a principal diferença entre uma ***função* void** e uma ***função*** com **return**? 10 pontos

- ◯ A **void** retorna valores, mas a com **return** não.
- ◯ A com **return** devolve valores, mas a **void** não.
- ◯ A **void** é sempre obrigatória em qualquer programa.
- ◯ A com **return** não pode ter argumentos.

PÁGINA 3 DE 8

	Escola Secundária de Santo André - sede **Grupo de Recrutamento 550 – Informática** Ficha de Avaliação– **Módulo 3 – Funções** **Duração: 50 minutos** **Os professores: Margarida Batista e João Brito** Ano letivo de 2024/ 2025 – DATA:xx/xx/xx	Curso: **TIS** Disciplina: **CBP** Turma **1.ºL**

8. Qual das seguintes afirmações referente a variáveis locais é verdadeira? 10 pontos

- ◯ São declaradas dentro de uma função e só podem ser usadas pela mesma.
- ◯ São declaradas fora de qualquer função.
- ◯ Podem ser usadas em qualquer parte do programa.
- ◯ São inicializadas automaticamente com o valor 0.

9. O que são ***argumentos*** de uma função? 10 pontos

- ◯ Instruções dentro do corpo da função.
- ◯ Variáveis locais usadas pelo programa.
- ◯ Variáveis que recebem valores e que são enviados para a função.
- ◯ Diretivas ao pré-processador incluídas no código.

10. Qual das seguintes opções é uma **vantagem** da programação modular? 10 pontos

- ◯ Permite executar várias funções simultaneamente.
- ◯ Organiza o código e evitam a redundância.
- ◯ Complica a solução.
- ◯ Torna desnecessário a utilização da função main().

11. Qual das seguintes opções relativamente a tipo de dados primitivos ***é verdadeira?*** 10 pontos

- ◯ O ***int*** contêm números reais.
- ◯ O ***float*** só contêm números inteiro.
- ◯ O ***double*** admite valores reais com precisão.
- ◯ O ***char*** admite dados lógicos.

PÁGINA 4 DE 8

	Escola Secundária de Santo André - sede **Grupo de Recrutamento 550 – Informática** Ficha de Avaliação– **Módulo 3 – Funções** **Duração: 50 minutos** **Os professores: Margarida Batista e João Brito** Ano letivo de 2024/ 2025 – DATA:xx/xx/xx	Curso: **TIS** Disciplina: **CBP** Turma **1.ºL**

12. Qual das seguintes afirmações relativamente a ***funções é verdadeira?*** 10 pontos

- ◯ Devolve mais do que um resultado em simultâneo.
- ◯ É sempre mais complexa.
- ◯ Permite realizar uma tarefa especifica com retorno de um valor.
- ◯ Não necessita de cabeçalho.

13. Qual das seguintes afirmações relativamente à etapa da compilação é **verdadeira?**

- ◯ Corrige automaticamente todos os erros do programa.
- ◯ Apenas traduz o código fonte para o código máquina.
- ◯ Identifica erros de sintaxe e de semântica no código fonte.
- ◯ Dá instruções ao pré-processador relativamente ao **#include**.

14. Qual das seguintes afirmações **relativamente a subprogramas é verdadeira?** 10 pontos

- ◯ Não obedece às regras da programação estruturada.
- ◯ Pode ou não retornar valores.
- ◯ É um bloco de código que apenas serve para fazer inputs e outputs.
- ◯ Nunca precisa de ser chamado para ser executado.

15. Qual das seguintes opções contém instruções associadas à biblioteca **stdio.h**? 10 pontos

- ◯ while.
- ◯ scanf() e printf().
- ◯ sqrt().
- ◯ round().

PÁGINA 5 DE 8

	Escola Secundária de Santo André - sede **Grupo de Recrutamento 550 – Informática** Ficha de Avaliação– **Módulo 3 – Funções** **Duração: 50 minutos** **Os professores: Margarida Batista e João Brito** Ano letivo de 2024/ 2025 – DATA:xx/xx/xx	Curso: **TIS** Disciplina: **CBP** Turma **1.ºL**

Grupo 2: Questões de Interpretação de código

Responda às questões seguintes tendo em atenção ao código apresentado nas imagens:

```c
# include <stdio.h>

int soma(int a, int b)
{
        return a + b;
}

int main()
{
        printf("%d\n", soma(3,5));
        return 0;
}
```

1. Qual o nome da função criada pelo programador no presente código? 10 pontos

2. Qual o papel da instrução **return a + b**? 10 pontos

3. Na função main() qual o valor devolvido pela função soma? 10 pontos

PÁGINA 6 DE 8

	Escola Secundária de Santo André - sede **Grupo de Recrutamento 550 – Informática** Ficha de Avaliação– **Módulo 3 – Funções** **Duração: 50 minutos** **Os professores: Margarida Batista e João Brito** Ano letivo de 2024/ 2025 – DATA:xx/xx/xx	Curso: **TIS** Disciplina: **CBP** Turma **1.ºL**

Analise a seguinte imagem e responda às seguintes questões:

```c
# include <stdio.h>

float CalculaMedia (float nota1, float nota2, float nota3)
{

        return (nota1 + nota2 + nota3) /3.0;

}

int main()
{

        float nota1, nota2, nota3, media;

        printf("Insira a nota 1:");
        scanf("%f", &nota1);

        printf("Insira a nota 2:");
        scanf("%f", &nota2);

        printf("Insira a nota 3:");
        scanf("%f", &nota3);

        media = CalculaMedia(nota1, nota2, nota3);
        printf("A media das 3 notas é %.2f", media);

        return 0;

}
```

4. Quais são as variáveis declaradas no código acima? 6 pontos

5. Porque motivo neste código se declaram variáveis com o tipo primitivo **float**? 6 pontos

6. Quantos valores recebe a ***função***? Justifique a sua resposta. 8 pontos

Fim da Avaliação.

PÁGINA 7 DE 8

	Escola Secundária de Santo André - sede **Grupo de Recrutamento 550 – Informática** Ficha de Avaliação– **Módulo 3 – Funções** **Duração: 50 minutos** **Os professores: Margarida Batista e João Brito** Ano letivo de 2024/ 2025 – DATA:xx/xx/xx	Curso: **TIS** Disciplina: **CBP** Turma **1.ºL**

Cotações

Grupo 1	**150 pontos**
1	10 pontos
2	10 pontos
3	10 pontos
4	10 pontos
5	10 pontos
6	10 pontos
7	10 pontos
8	10 pontos
9	10 pontos
10	10 pontos
11	10 pontos
12	10 pontos
13	10 pontos
14	10 pontos
15	10 pontos
Grupo 2	**50 pontos**
1	10 pontos
2	10 pontos
3	10 pontos
4	6 pontos
5	6 pontos
6	8 pontos
Total	**200 pontos**

FIM!

PÁGINA 8 DE 8

	Escola Secundária de Santo André - sede **Grupo de Recrutamento 550 – Informática** Ficha de Avaliação– **Módulo 3 – Funções** **Duração: 50 minutos** **Os professores: Margarida Batista e João Brito** Ano letivo de 2024/ 2025 – DATA:xx/xx/xx	Curso: **TIS** Disciplina: **CBP** Turma **1.ºL**

Classificação: Valores:

Nome do aluno: _____João Brito_____ Número: _____

A presente ficha de avaliação tem como objetivo aferir os conhecimentos teórico-práticos adquiridos relativamente a módulos ou subprogramas.

Duração da prova: 50 minutos **8** páginas

O teste é constituído por dois grupos:

Grupo 1: Questões de escolha múltipla que avaliam os conceitos teóricos.
Neste grupo para cada questão existe apenas uma resposta.

Grupo 2: Questões de interpretação e análise de código.

PÁGINA 1 DE 8

	Escola Secundária de Santo André - sede **Grupo de Recrutamento 550 – Informática** Ficha de Avaliação– **Módulo 3 – Funções** **Duração: 50 minutos** **Os professores: Margarida Batista e João Brito** Ano letivo de 2024/ 2025 – DATA:xx/xx/xx	Curso: **TIS** Disciplina: **CBP** Turma **1.ºL**

Grupo 1: Questões de escolha múltipla

Escolha a resposta correta.

1 Qual das seguintes opções define programação modular? 10 pontos

- ( ) Uma técnica para escrever programas sem a utilização de funções.
- ( ) Uma forma de criar programas sem erros.
- ( ) Uma técnica usada para programar apenas em linguagens orientadas a objetos.
- (X) Uma técnica que divide um programa em partes menores, chamados módulos ou subprogramas.

2 Qual das seguintes opções ***NÃO*** é uma vantagem da programação modular? 10 pontos

- ( ) Facilitar a manutenção do código.
- (X) Aumentar a redundância do código.
- ( ) Permitir o trabalho de equipa.
- ( ) Reutilizar código noutros outros projetos.

3. Qual das seguintes instruções é considerada uma ***função de "Output"***? 10 pontos

- (X) **printf().**
- ( ) **main().**
- ( ) **start().**
- ( ) **scanf().**

4. Qual a etapa em que a solução é traduzida para ***linguagem máquina***? *10 pontos*

- ( ) Edição.
- ( ) Execução.
- (X) Compilação.
- ( ) Pré-processamento.

PÁGINA 2 DE 8

	Escola Secundária de Santo André - sede **Grupo de Recrutamento 550 – Informática** Ficha de Avaliação– **Módulo 3 – Funções** **Duração: 50 minutos** **Os professores: Margarida Batista e João Brito** Ano letivo de 2024/ 2025 – DATA:xx/xx/xx	Curso: **TIS** Disciplina: **CBP** Turma **1.ºL**

5. O que acontece durante o ***pré-processamento***? 10 pontos

- ☐ Escrevemos o código num editor de texto.
- ☒ O código é preparado para compilação, analisando diretivas como **#include**.
- ☐ O programa é testado no ambiente de execução.
- ☐ As funções são otimizadas.

6. Qual das seguintes afirmações sobre as ***funções é verdadeira***? 10 pontos

- ☒ Um bloco de código que realiza uma tarefa específica.
- ☐ Uma **void** deve sempre retornar um valor.
- ☐ Todas as funções de um programa são executadas automaticamente.
- ☐ Uma função não pode chamar outra.

7. Qual é a principal diferença entre uma ***função* void** e uma ***função*** com **return**? 10 pontos

- ☐ A **void** retorna valores, mas a com **return** não.
- ☒ A com **return** devolve valores, mas a **void** não.
- ☐ A **void** é sempre obrigatória em qualquer programa.
- ☐ A com **return** não pode ter argumentos.

PÁGINA 3 DE 8

	Escola Secundária de Santo André - sede **Grupo de Recrutamento 550 – Informática** Ficha de Avaliação– **Módulo 3 – Funções** **Duração: 50 minutos** **Os professores: Margarida Batista e João Brito** Ano letivo de 2024/ 2025 – DATA:xx/xx/xx	Curso: **TIS** Disciplina: **CBP** Turma **1.ºL**

8. Qual das seguintes afirmações referente a variáveis locais é verdadeira? 10 pontos

- [x] São declaradas dentro de uma função e só podem ser usadas pela mesma.
- [ ] São declaradas fora de qualquer função.
- [ ] Podem ser usadas em qualquer parte do programa.
- [ ] São inicializadas automaticamente com o valor 0.

9. O que são ***argumentos*** de uma função? 10 pontos

- [ ] Instruções dentro do corpo da função.
- [ ] Variáveis locais usadas pelo programa.
- [x] Variáveis que recebem valores e que são enviados para a função.
- [ ] Diretivas ao pré-processador incluídas no código.

10. Qual das seguintes opções é uma **vantagem** da programação modular? 10 pontos

- [ ] Permite executar várias funções simultaneamente.
- [x] Organiza o código e evita a redundância.
- [ ] Complica a solução.
- [ ] Torna desnecessário a utilização da função main().

11. Qual das seguintes opções relativamente a tipo de dados primitivos ***é verdadeira?*** 10 pontos

- [ ] O ***int*** contêm números reais.
- [ ] O ***float*** só contêm números inteiro.
- [x] O ***double*** admite valores reais com precisão.
- [ ] O ***char*** admite dados lógicos.

PÁGINA 4 DE 8

	Escola Secundária de Santo André - sede **Grupo de Recrutamento 550 – Informática** Ficha de Avaliação– **Módulo 3 – Funções** **Duração: 50 minutos** **Os professores: Margarida Batista e João Brito** Ano letivo de 2024/ 2025 – DATA:xx/xx/xx	Curso: **TIS** Disciplina: **CBP** Turma **1.ºL**

12. Qual das seguintes afirmações relativamente a ***funções é verdadeira?*** 10 pontos

- ( ) Devolve mais do que um resultado em simultâneo.
- ( ) É sempre mais complexa.
- (X) Permite realizar uma tarefa especifica com retorno de um valor.
- ( ) Não necessita de cabeçalho.

13. Qual das seguintes afirmações relativamente à etapa da compilação é **verdadeira?**

- ( ) Corrige automaticamente todos os erros do programa.
- ( ) Apenas traduz o código fonte para o código máquina.
- (X) Identifica erros de sintaxe e de semântica no código fonte.
- ( ) Dá instruções ao pré-processador relativamente ao **#include**.

14. Qual das seguintes afirmações **relativamente a subprogramas é verdadeira?** 10 pontos

- ( ) Não obedece às regras da programação estruturada.
- (X) Pode ou não retornar valores.
- ( ) É um bloco de código que apenas serve para fazer inputs e outputs.
- ( ) Nunca precisa de ser chamado para ser executado.

15. Qual das seguintes opções contém instruções associadas à biblioteca **stdio.h**? 10 pontos

- ( ) while.
- (X) scanf() e printf().
- ( ) sqrt().
- ( ) round().

PÁGINA 5 DE 8

	Escola Secundária de Santo André - sede **Grupo de Recrutamento 550 – Informática** Ficha de Avaliação– **Módulo 3 – Funções** **Duração: 50 minutos** **Os professores: Margarida Batista e João Brito** Ano letivo de 2024/ 2025 – DATA:xx/xx/xx	Curso: **TIS** Disciplina: **CBP** Turma **1.ºL**

Grupo 2: Questões de Interpretação de código

Responda às questões seguintes tendo em atenção ao código apresentado nas imagens:

```c
# include <stdio.h>

int soma(int a, int b)
{
        return a + b;
}

int main()
{
        printf("%d\n", soma(3,5));
        return 0;
}
```

1. Qual o nome da função criada pelo programador no presente código? 10 pontos

O nome da função criada pelo programador chama-se soma() e recebe dois valores porque calcula a+b.

2. Qual o papel da instrução **return a + b**? 10 pontos

A instrução return a + b, devolve o resultado de uma soma dos dois argumentos (a e b) à função main()

3. Na função main() qual o valor devolvido pela função soma? 10 pontos

Na função main() o valor devolvido pela função soma() é o resultado da soma de (a+b) = 8

PÁGINA 6 DE 8

	Escola Secundária de Santo André - sede **Grupo de Recrutamento 550 – Informática** Ficha de Avaliação– **Módulo 3 – Funções** **Duração: 50 minutos** **Os professores: Margarida Batista e João Brito** Ano letivo de 2024/ 2025 – DATA:xx/xx/xx	Curso: **TIS** Disciplina: **CBP** Turma **1.ºL**

Analise a seguinte imagem e responda às seguintes questões:

```c
# include <stdio.h>

float CalculaMedia (float nota1, float nota2, float nota3)
{

        return (nota1 + nota2 + nota3) /3.0;

}

int main()
{

        float nota1, nota2, nota3, media;

        printf("Insira a nota 1:");
        scanf("%f", &nota1);

        printf("Insira a nota 2:");
        scanf("%f", &nota2);

        printf("Insira a nota 3:");
        scanf("%f", &nota3);

        media = CalculaMedia(nota1, nota2, nota3);
        printf("A media das 3 notas é %.2f", media);

        return 0;

}
```

4. Quais são as variáveis declaradas no código acima? 6 pontos

No código acima as variáveis declaradas são: media, nota1, nota2 e nota3

5. Porque motivo neste código se declaram variáveis com o tipo primitivo **float**? 6 pontos

Neste código se utilizássemos um tipo primitivo de dados inteiro (int) as notas seriam armazenadas como números inteiros o que perderia a parte decimal. Utilizar dados do tipo primitivo float aumenta a precisão e uma vez que o código envolve divisões, utilizar float evita problemas de truncamento (corte da parte decimal).

6. Quantos valores recebe a ***função***? Justifique a sua resposta. 8 pontos

A função CalculaMedia recebe 3 valores, float nota1, float nota2 e float nota3 o que significa que a função é chamada com três argumentos pelo utilizador.

Fim da Avaliação.

PÁGINA 7 DE 8

	Escola Secundária de Santo André - sede **Grupo de Recrutamento 550 – Informática** Ficha de Avaliação– **Módulo 3 – Funções** **Duração: 50 minutos** **Os professores: Margarida Batista e João Brito** Ano letivo de 2024/ 2025 – DATA:xx/xx/xx	Curso: **TIS** Disciplina: **CBP** Turma **1.ºL**

Cotações

Grupo 1	**150 pontos**
1	10 pontos
2	10 pontos
3	10 pontos
4	10 pontos
5	10 pontos
6	10 pontos
7	10 pontos
8	10 pontos
9	10 pontos
10	10 pontos
11	10 pontos
12	10 pontos
13	10 pontos
14	10 pontos
15	10 pontos
Grupo 2	**50 pontos**
1	10 pontos
2	10 pontos
3	10 pontos
4	6 pontos
5	6 pontos
6	8 pontos
Total	**200 pontos**

FIM!

Atividade aplicada durante as aulas

Semana da Net-Segura Atividade: Deep Learning - Como as Máquinas Aprendem?

Numa era em que a informação é amplamente disseminada online, é essencial que os alunos desenvolvam competências ao nível da seleção e recolha de conteúdos de qualidade, recorrendo a fontes crediveis garantindo assim a utilização da internet de forma segura. A Inteligência Artificial é uma subárea do *Deep Learning* uma vez que utiliza grandes quantidades de redes neurais a partir da qual consegue procurar e analisar grandes quantidades de dados.

A atividade tem como objetivo principal proporcionar aos alunos uma experiência prática e reflexiva sobre a utilização responsável da internet para a criação de um trabalho académico utilizando como apoio uma ferramenta de inteligência artificial aplicando as normas de citação académica. A realização deste trabalho tem como objetivo demonstrar o potencial da IA relativamente à capacidade de síntese e análise de informação, mas também incentivar a leitura critica dos resultados obtidos.

Com esta atividade, é esperado que os alunos adquiram competências essenciais ao nivel da pesquisa académica, mas ao mesmo tempo compreendam a importância da segurança digital e da credibilidade das fontes e utilização ética das ferramentas de inteligência Artificial. Integrar estas ferramentas de inteligência artificial durante a aprendizagem contribui para uma visão mais critica e informada sobre o impacto das mesmas para o campo da investigação e do conhecimento dos alunos.

1.ª parte do trabalho a desenvolver:

1. Pesquisa de um artigo sobre o tema do *Deep Learning*, recorrendo a fontes crediveis, tais como universidades de renome (exemplo: Universidade Católica, Universidade de Cambridge entre outras).
2. O artigo pesquisado não deverá ultrapassar um máximo de 15 páginas.
3. Após a seleção do artigo o aluno deverá realizar o download do mesmo para o ambiente de trabalho do computador.
4. Deverá abrir a página da internet e abrir a ferramenta de inteligência artificial "ChatGPT" e realizar o upload do artigo.
5. Após a realização do upload do artigo o aluno deverá pedir um resumo do artigo que submeteu e de seguida trabalhá-lo.

2.ª parte do trabalho a desenvolver:

1. O trabalho deverá ser realizado em Word.
2. O trabalho deverá seguir a seguinte estrutura:
    a. Capa (Nome, número, turma, nome da escola).
    b. Resumo.
    c. Introdução.
    d. Desenvolvimento.
    e. Conclusão.
    f. Referências (Cumprir as normas da APA 7ª edição).
3. Não deverá ultrapassar as duas páginas sem contabilizar a capa e a página das referências.
4. Durante a construção do texto deverão citar o autor do artigo cumprindo as normas APA.