Universidade de Lisboa

Ler antes de escrever: Utilização da abordagem pedagógica PRIMM na aprendizagem de estruturas de dados estáticas em Linguagem C

Carlos Manuel Martins Lopes

Mestrado em Ensino de Informática

Relatório da Prática de Ensino Supervisionada orientada pelo

Professor Doutor João Piedade e pelo Professor Doutor Carlos Lourenço

2024

Agradecimentos

Quero agradecer a todos os docentes do Instituto de Educação, com qual tive o privilégio de me cruzar, pelos conhecimentos transmitidos.

Um agradecimento especial ao Professor Doutor João Piedade, orientador da Prática de Ensino Supervisionada, pelo seu apoio, nomeadamente, pelas sugestões e correções no desenvolvimento deste relatório, em que só demonstra a sua dedicação, disponibilidade e acima de tudo o seu excelente profissionalismo.

Destaco igualmente o Professor Carlos Lourenço, pela orientação nos conceitos científicos e todos comentários e sugestões que me ajudaram na melhoria deste relatório.

Agradeço à Professora Cooperante Ana Gato, pelos conselhos e pela orientação no decorrer do processo de observação, planificação e intervenção na sua turma.

Quero agradecer à Direção do Agrupamento de Escolas 4 de Outubro, Professora Cristina Marques e Professora Isabel Araújo, por terem aceitado a minha sugestão para a realização do protocolo com o Instituto de Educação, no qual foi possível realizar a Prática de Ensino Supervisionada na Escola Secundária Dr. António Carvalho Figueiredo.

Aos meus colegas de mestrado, nomeadamente, Amâncio Ferreira, Paulo Freitas, Miguel Santos, Márcia Maças e Gilberto Júnior agradeço, pela amizade, colaboração e apoio mútuo.

À minha família, em especial ao meu pai, minha mãe e minha irmã, pelo apoio e incentivo que sempre me deram para investir meu tempo na educação. Vocês são a minha base e a minha força.

A todos os meus amigos que, de alguma forma, colaboraram para a realização da minha tese de mestrado, meu sincero agradecimento.

Resumo

Este relatório surge no âmbito da Unidade Curricular de Iniciação à Prática Profissional IV, do Mestrado em Ensino de Informática, do Instituto de Educação da Universidade de Lisboa. Relata a intervenção pedagógica que ocorreu na Escola Secundária Dr. António Carvalho Figueiredo, do Agrupamento de Escolas 4 de Outubro, na disciplina de Programação de Sistemas de informação (PSI), no módulo 4, Estruturas de Dados Estáticas, numa turma do 1º ano, do curso Profissional de Técnico de Gestão e Programação e Sistemas de Informáticos (TGPSI). Lecionei 7 aulas no total de 675 minutos sob a supervisão da Professora Cooperante. Os conceitos científicos abordados foram as Estruturas de Dados Estáticas, Vetores (*arrays*). A abordagem pedagógica adotada foi a PRIMM (*Predict, Run, Investigate, Modify e Make*). Como atividades práticas de aprendizagem, os alunos realizaram, individualmente, duas sessões da abordagem pedagógica PRIMM e por último implementaram o cenário de aprendizagem "Sequência Mágica", dedicado ao desenvolvimento de um programa em C. Considerando a abordagem pedagógica adotada, defini o seguinte problema de investigação: "Qual o papel da abordagem pedagógica PRIMM na melhoria da aprendizagem e dos conceitos de programação em linguagem C relativos às estruturas de dados estáticas?" Para dar resposta ao problema de investigação defini as seguintes questões: 1) Qual o efeito da utilização da abordagem PRIMM na aprendizagem dos conceitos relacionados com as estruturas de dados estáticas em linguagem C? 2) Que mais valias são reconhecidas pelos alunos na compreensão dos conceitos decorrentes da utilização da abordagem pedagógica? 3) Quais as principais dificuldades sentidas pelos alunos na utilização da abordagem pedagógica PRIMM? O design metodológico foi de natureza quase experimental, onde coexistiram um grupo experimental e um grupo de controlo. Os resultados permitiram sinalizar que não se registaram diferenças estatisticamente significativas entre os grupos, tendo, no entanto, sido reconhecidas pelos alunos algumas mais valias da abordagem PRIMM na aprendizagem de programação.

Palavras-chave: PRIMM; Ensino Profissional; Estrutura de Dados Estáticas; Aplicação Consola; Linguagem Programação C

Abstract

This report appears within the scope of the Curricular Unit of Initiation to Professional Practice IV, of the Master's Degree in Computer Science Teaching, of the Institute of Education of the University of Lisbon. It describe the pedagogical intervention took place at Escola Secundária de Dr. António Carvalho Figueiredo, of the 4 de Outubro School Group, in the subject of Information Systems Programming, in module 4, Static Data Structures, in a 1st year class, of the Professional Management and Programming Technician and Information Systems course. I taught 7 classes for a total of 675 minutes under the supervision of the cooperating teacher. The scientific concepts covered were Static Data Structures, Vectors (arrays). The pedagogical approach adopted was PRIMM (*Predict, Run, Investigate, Modify and Make*). As practical learning activities, students individually carried out two sessions of the PRIMM pedagogical approach and finally implemented the "Sequência Mágica" learning scenario, dedicated to develop a program in C language. Considering the pedagogical approach adopted, i defined the following research problem: "What role does the PRIMM pedagogical approach play in improving learning and programming concepts in C language related to static data structures". To answer the research problem, I defined the following questions: 1) What is the effect of using the PRIMM approach on the learning concepts related to static data structures in C language? 2) What added value do students recognize in understanding the concepts as a result of the use of this pedagogical approach? 3) What are the main difficulties experienced by students when using the PRIMM pedagogical approach?

The methodological design was of an almost experimental nature, where an experimental group and a control group coexisted The results showed that the research question was answered, and indicate that the PRIMM pedagogical approach did not have a significant effect on student results.

Keywords: PRIMM; Profissional Education; Static Data Structures; Console Aplication; C Programming Language

Índice

Índice de Figuras

Índice de Tabelas

Abreviaturas

AE4O	Agrupamento de Escolas 4 Outubro
ASE	Ação Social Escolar
CFT	Componente de Formação Tecnológica
CT	Conselho de Turma
EQAVET	European Quality Assurance Reference Framework for IPP IV Iniciação à Prática Profissional IV
ESACF	Escola Secundária Dr. António Carvalho Figueiredo
FCT	Formação em Contexto de Trabalho
IDE	Ambiente de desenvolvimento integrado
IE	Instituto de Educação
IPP III	Iniciação à Prática Profissional III
IPP IV	Iniciação à Prática Profissional IV
MEI	Mestrado em Ensino de Informática
PAE	Técnico/a de Ação Educativa
PAF	Técnico/a de Farmácia
PAG	Técnico/a de Apoio à Gestão
PAP	Prova de Aptidão Profissional
PAS	Técnico/a Auxiliar de Saúde
PAV	Técnico/a de Audiovisuais PAV
PCD	Técnico/a de Comunicação e Sistemas Digitais PCD
PCM	Técnico/a Comercial
PDP	Técnico/a de Desporto
PGI	Técnico/a de Gestão e Programação de Sistemas Informáticos

PIA	Intérprete/Ator/Atriz
PIS	Técnico/a de Informática – Sistemas
PMM	Técnico/a de Multimédia PMM
PRIMM	Predict, Run, Investigate, Modify, Make
PrBL	Problem Based Learning
PSI	Programação e Sistemas de Informação
UL	Universidade de Lisboa
TI	Tecnologias da Informação
T1	Turno 1
T2	Turno 2
TGPSI	Técnico de Gestão e Programação de Sistemas Informáticos
Vocational Education and Training	
UFCD	Unidades de Formação de Curta Duração

1. Introdução

Este relatório tem como objetivo apresentar o percurso efetuado na disciplina de Iniciação à Prática Profissional IV (IPP IV), do Mestrado em Ensino de Informática (MEI), do Instituto de Educação (IE) da Universidade de Lisboa (UL).

O percurso efetuado, foi a intervenção pedagógica que decorreu na Escola Secundária Dr. António Carvalho Figueiredo (ESACF), no curso profissional de Técnico de Gestão e Programação de Sistemas Informáticos (TGPSI), na disciplina de Programação e Sistemas de Informação (PSI), da responsabilidade da Professora Cooperante Ana Gato.

A intervenção pedagógica ocorreu no início do quarto módulo, Estruturas de Dados Estáticos, da disciplina de PSI, com a duração de 675 minutos, correspondem à primeira metade do módulo, entre os dias 4 e 18 de março de 2024. A intervenção foi orientada para o desenvolvimento das aprendizagens sobre as estruturas de dados estáticas na linguagem C.

A abordagem pedagógica que utilizei na minha intervenção foi a PRIMM, porque da revisão da literatura realizada, entendi que as potencialidades desta abordagem podem despertar o interesse e a motivação nos alunos a resolver problemas de programação. Esta abordagem coloca o seu foco na leitura do código fornecido, em vez de um aluno iniciante na programação desenvolver de um programa do "zero".

Após a introdução, apresento o contexto escolar: meio envolvente; a escola e a turma, onde ocorreu a intervenção.

No terceiro capítulo, apresento o enquadramento curricular da disciplina de PSI, desde à análise da matriz curricular do Curso Profissional TGPSI, bem como a análise da disciplina de PSI e seus conteúdos programáticos.

No quarto capítulo, apresento a planificação da intervenção, desde a observação das aulas, às competências a desenvolver, aos objetivos de aprendizagem, às metodologias e estratégias utilizadas, aos recursos e atividades realizadas.

No quinto capítulo, apresento a avaliação da intervenção pedagógica, a avaliação diagnóstica, a avaliação formativa através de grelhas de observação das atividades realizadas pelos alunos e a avaliação sumativa.

No sexto capítulo, apresento a dimensão investigativa, onde exponho a problemática e quais as questões de investigação, os instrumentos de recolha de dados e as conclusões do estudo.

No sétimo capítulo, apresento o balanço reflexivo da intervenção.

No oitavo e nono capítulo, apresento as referências e os anexos respetivamente.

2. Caracterização Contexto Escolar

O Agrupamento de Escolas 4 de Outubro (AE4O) procura continuamente consolidar os passos que vem dando, estendendo e aprofundando as suas raízes na região em que se insere. No exercício da sua autonomia, o agrupamento procura garantir e afirmar a sua especificidade e a sua identidade.

O Agrupamento aposta na melhoria contínua da sua oferta educativa e do serviço que presta à comunidade, pretende assim implementar um sistema de garantia da qualidade alinhado com os princípios do Quadro de Referência Europeu de Garantia da Qualidade na Educação e Formação Profissional (EQAVET), com o objetivo de obter o Selo de Qualidade EQAVET (Projeto, 2023).

No que diz respeito à melhoria do serviço que presta à comunidade, o AE4O, segundo Túlio & Cruz, (2023), valoriza a participação dos parceiros a nível da orientação pedagógica (componente curricular, atividades de complemento e enriquecimento curriculares ou extracurriculares, projetos internos e externos, etc.). O AE4O beneficia da melhoria dos processos e do uso parcerias como um instrumento facilitador de oferta de estágios e de entrada no mercado de trabalho, reforçando a imagem do Agrupamento. O **Anexo A**, apresenta a lista de entidades/parceiros de acolhimento de estágios e de entidades empregadoras.

O EQAVET (acrónimo de European Quality Assurance Reference Framework for Vocational Education and Training), em português Quadro de Referência Europeu de Garantia da Qualidade para a Educação e Formação Profissional), é o instrumento de referência para promover e monitorizar o aperfeiçoamento dos sistemas europeus de ensino e formação profissional (EQAVET, 2021).

De acordo, Regulamento Interno, (2014), o AE4O atribui quatro tipos de reconhecimento de mérito:

- **Académico –** Para todos os alunos que na avaliação sumativa final obtenham:

- **Nos 1º, 2º e 3º anos do 1º ciclo do ensino básico**, a menção de Muito Bom em, pelo menos, quatro áreas curriculares e não inferior a Bom nas restantes;
- **No 4º ano do 1º ciclo do ensino básico**, uma média de 4,5 ou superior nas áreas de Matemática e de Português, a menção de Muito Bom em, pelo menos, três áreas curriculares e nenhuma menção inferior a Bom;

- **Nos 2º e 3º ciclos do ensino básico**, média final igual ou superior 4,50 e nenhuma classificação inferior a 3;

- **No ensino secundário**, média final igual ou superior a 17,50 (classificação em todas as disciplinas do plano curricular).

- **Desportivo –** Para os alunos que, ao nível do Desporto Escolar, o coletivo dos professores de Educação Física considerar merecedores;
- **Cidadania –** Para os alunos que o Conselho de Turma (CT), ou outras estruturas, considerar que merecem este reconhecimento pelas atitudes demonstradas ao longo do ano letivo;
- **Representatividade –** Para os alunos que representarem o Agrupamento em concursos regionais, nacionais ou internacionais com desempenhos relevantes, indicados pelos professores que acompanharam esses processos de representação;

2.1. Meio Envolvente

O AE4O situa-se no distrito de Lisboa e no concelho de Loures. Foi criado a 26 de julho de 1886, e possui três zonas distintas: a parte oriental, residencial e de forte presença de indústria e serviços; a parte ocidental com um sobre crescimento urbano mal planeado e alguma indústria; e a parte norte, de que faz parte a sede do concelho onde se situam os serviços (reflexo da função de sede de concelho).

O concelho de Loures, com 167,24 km² de área e 201 590 habitantes (dados de 2021), está subdividido em 18 freguesias, conforme mostra a **Figura 1**, compreende duas cidades: Loures (elevada a cidade em 9 de agosto de 1990) e Sacavém (elevada a cidade a 4 de junho de 1997) e sete vilas: Bobadela, Bucelas, Camarate, Moscavide, Santa Iria de Azóia, Santo António dos Cavaleiros e São João da Talha.

Figura 1

Freguesias do Concelho de Loures

No dia 4 de outubro do ano de 1910, dá-se em Loures um facto histórico de grande importância: depois de ocupados os Paços do Concelho, uma bandeira com as cores republicanas, o verde e o vermelho, é hasteada e declarada a Implantação da República pela Junta Revolucionária. É neste facto histórico que se baseia o nome do Agrupamento.

2.2. O Agrupamento de Escolas 4 de Outubro

O AE4O, criado a 3 de julho de 2012, é constituído por quatro escolas:

- Escola Secundária Dr. António Carvalho Figueiredo (escola sede - 3º ciclo e ensino secundário)
- Escola Básica de Vila de Rei (1ºCiclo e Jardim Infância)
- Escolas Básicas da Bemposta (1º ciclo)
- Escola Básica de Bucelas (pré-escolar, 1º, 2º e 3º ciclos).

2.3. A Escola Secundária Dr. António Carvalho Figueiredo

A ESACF, escola sede do AE4O, localiza-se na Cidade de Loures. Inicialmente batizada como Escola Secundária nº 2 de Loures (Diário da República nº227, I Série, de 1 de outubro de 1983), a ESACF recebe o seu atual nome, em 1989, por proposta da autarquia local, em homenagem a António Carvalho de Figueiredo (1853-1917), distinto médico-cirurgião que se notabilizou como bacteriologista e pioneiro no estudo da doença do sono.

Em 2009/2010, é requalificada e ampliada pela Parque Escolar, passando de uma estrutura arquitetónica em pavilhões para um edifício por blocos interligados, rodeado de um grande espaço dedicado a jardim, conforme mostra a **Figura *2***.

Figura 2

Vista aérea Escola Secundária Dr. António Carvalho Figueiredo

A oferta educativa do Agrupamento é muito diversificada, abrangendo todos os níveis de ensino. No ensino secundário, o Agrupamento oferece todos os cursos científico-humanísticos:

- Ciências e Tecnologias (CT);
- Línguas e Humanidades (LH);
- Ciências Socioeconómicas (CS);
- Artes Visuais (AV).

Para Além do ensino secundário, o Agrupamento oferece Percursos Curriculares Alternativos e Cursos Profissionais, numa preocupação constante em dar respostas adequadas às necessidades, em termos de percurso escolar, manifestadas por alunos e famílias. A nível de outras ofertas formativas, a ESACF oferece Cursos Profissionais em diferentes áreas, nomeadamente:

- Técnico/a Auxiliar de Saúde (PAS);
- Técnico/a de Ação Educativa (PAE);
- Técnico/a Comercial (PCM);
- Técnico/a de Apoio à Gestão (PAG);
- Técnico/a de Gestão e Programação de Sistemas Informáticos (PGI);
- Técnico/a de Desporto (PDP);
- Técnico/a de Farmácia (PAF);
- Técnico/a de Multimédia (PMM);
- Técnico/a de Audiovisuais (PAV);
- Técnico/a de Comunicação e Sistemas Digitais (PCD);
- Técnico/a de Informática - Sistemas (PIS);
- Intérprete/Ator/Atriz (PIA).

Para além da oferta educativa e formativa a ESACF desenvolve um conjunto de atividades de enriquecimento (oferta não curricular), que procura de uma forma integral o sucesso dos alunos, do ponto de vista intelectual, pessoal e social de forma a promover o trabalho de equipa e a procura de soluções para os diferentes problemas.

As atividades de enriquecimento são:

- Desporto escolar;
- Eco-Escolas;
- Clube Ciência Viva;
- Clube Robótica;
- Astronomia;
- Horta Pedagógica;
- Clube de Cerâmica;
- Programa de Educação para a Saúde;
- Matemática;
- Clube de Teatro;
- Sala de Estudo;

Segundo o PEA, a escola tem 30 salas de aulas, 6 laboratórios, 6 salas artes/EV/ET, 5 salas de informática, 4 campos desportivos exteriores, 8 balneários, 1 biblioteca, 1 sala de estudo e serviços reprografia/papelaria, bar/refeitório.

No ano letivo 2022/2023, frequentavam a escola 1482 alunos, repartidos por 464 alunos no 3º ciclo, 728 alunos no secundário e 290 no ensino profissional. Os alunos abrangidos pela ação social totalizam 402, distribuídos 205 pelo escalão A, 167 pelo escalão B e 30 pelo escalão C.

Tendo a escola uma forte componente no ensino profissional, possui uma forte rede de contactos com empresas do concelho onde coloca os alunos a realizar o seu estágio profissional.

A ESACF possui um clube de robótica onde os alunos têm a oportunidade de experimentar a programação de robôs, com o objetivo de motivar os alunos na aprendizagem da programação em particular e da informática no geral.

A ACF participa em diversos concursos nacionais nas áreas tecnológicas, nomeadamente, programação e robótica.

A ACF promove a participação dos alunos em desafios que estimulam o pensamento critico e computacional, tais como o Desafio Bebras e *Hour of Code*.

2.4. A Sala de Aula

A sala de onde observei a turma, é a sala TIC 1 (vide **Figura 3**), a disposição das carteiras é em forma de U, é constituída por 13 computadores para os alunos, um computador para o professor, um videoprojector e um quadro branco. No centro da sala existem 3 filas de carteiras centrais sem computadores.

Figura 3

Sala TIC 1

Nas aulas teóricas, os alunos ocupam as carteiras laterais e as centrais, para ficarem enquadrados com quadro onde a professora projeta os conteúdos. Nas aulas práticas os alunos ocupam as carteiras laterais e as carteiras do fundo da sala, são as que têm um computador disponível.

Apesar de a turma estar dividida em 2 turnos, a sala não possui 1 computador por aluno.

Dos 13 computadores existentes na sala, existem 2 que não estão operacionais, todos os computadores têm instalado o ambiente de desenvolvimento integrado (IDE) DevC, no entanto, à data da intervenção nenhum dos alunos possui o kit tecnológico, apesar de já o terem solicitado, pelo que irão usar os computadores disponíveis na sala.

O kit tecnológico é cedido pela escola, no âmbito do plano transição digital, é constituído por 1 computador portátil e acesso à internet móvel.

Os computadores existentes na sala de aula possuem características necessárias para o ensino da programação, são as seguintes:

- Sistema operativo Windows 10 Pro;
- CPU Intel(R) Core(TM) i5 a 3.2Ghz;
- Memória RAM 4 Gb;
- Disco Rígido 237 Gb
- Ligação à internet

2.5. Caracterização da Turma 1º PGI e Alunos

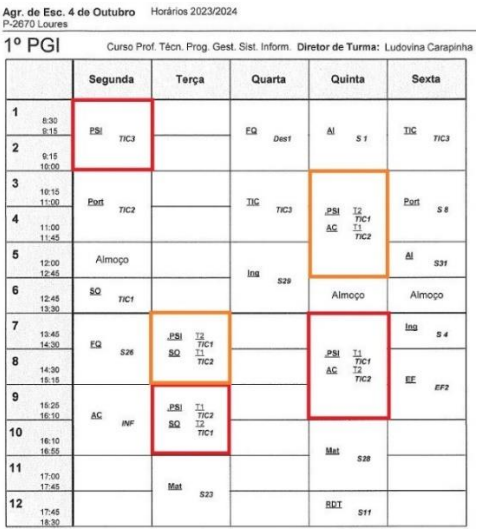
A turma é constituída por 30 alunos, e está dividida em 2 turnos, o turno 1 (T1) e turno 2 (T2), com 15 alunos em cada turno. A disciplina PSI possui uma carga horária de 315 minutos semanais para cada turno, distribuída da seguinte forma:

- Aula teórica de 90 minutos às segundas-feiras de manhã das 8:30h-10:00h com os 2 turnos em simultâneo;
- Aula prática de 90 minutos às terças-feiras das 13:45h-15:15h com T2 e das 15:25h-16:55h com T1;
- Aula prática de 135 minutos às quintas-feiras das 10:15h-12:55h com T2 e das 13:45h-16:10h com T1

As aulas teóricas para o T1 e o T2, bem como as aulas práticas do T2 (selecionados com a cor vermelha) foram lecionadas por mim, enquanto as aulas práticas do T1 (selecionados com a cor laranja) foram lecionadas pela Professora Cooperante (vide **Figura 4**).

Figura 4

Horário dos dois turnos da turma 1º PGI



Agr. de Esc. 4 de Outubro Horários 2023/2024
P-2670 Loures

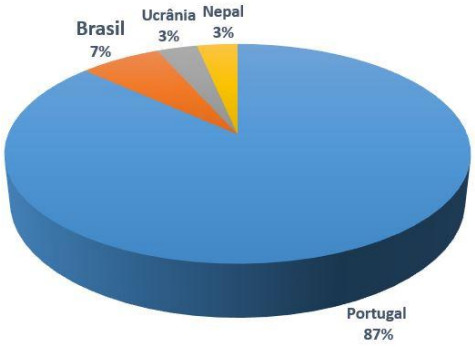
1º PGI Curso Prof. Técn. Prog. Gest. Sist. Inform. **Diretor de Turma:** Ludovina Carapinha

	Segunda	Terça	Quarta	Quinta	Sexta
1 8:30 9:15	PSI TIC3		FQ Des1	AI S 1	TIC TIC3
2 9:15 10:00					
3 10:15 11:00	Port TIC2		TIC TIC3	PSI T2 TIC1 AC T1 TIC2	Port S 8
4 11:00 11:45					
5 12:00 12:45	Almoço		Ing S29		AI S31
6 12:45 13:30	SO TIC1			Almoço	Almoço
7 13:45 14:30	FQ S26	PSI T2 TIC1 SO T1 TIC2		PSI T1 TIC1 AC T2 TIC2	Ing S 4
8 14:30 15:15					EF EF2
9 15:25 16:10	AC INF	PSI T1 TIC2 SO T2 TIC1			
10 16:10 16:55				Mat S28	
11 17:00 17:45		Mat S23			
12 17:45 18:30				RDT S11	

A maioria dos alunos são de nacionalidade portuguesa, à exceção de dois alunos brasileiros, um ucraniano e outro nepalês (vide **Figura 5**), nenhum deles está matriculado à disciplina de Português Língua não Materna e não apresentam dificuldades na compreensão e escrita da língua portuguesa.

Figura 5

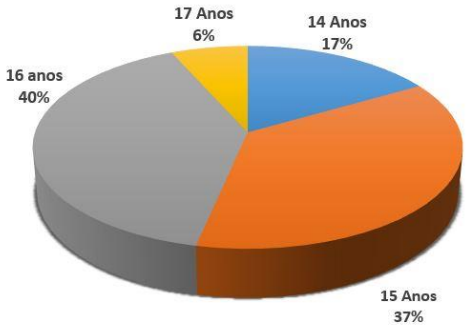
Distribuição dos alunos da turma pela nacionalidade



No que diz respeito às idades, a turma é constituída por alunos idades entre os 14 e 17 anos (vide **Figura 6**).

Figura 6

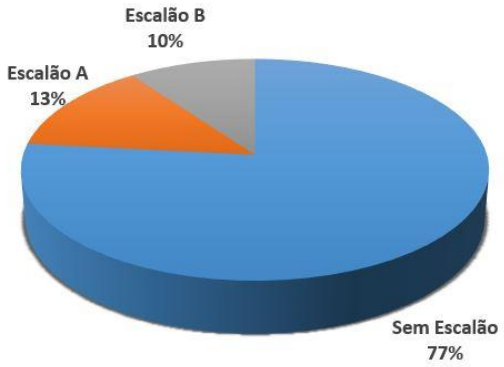
Distribuição dos alunos pela idade no início do curso



Em relação à Ação Social Escolar (ASE), a turma possui 4 alunos no escalão A, 3 no escalão B e os restantes 23 sem escalão (vide **Figura 7**).

Figura 7

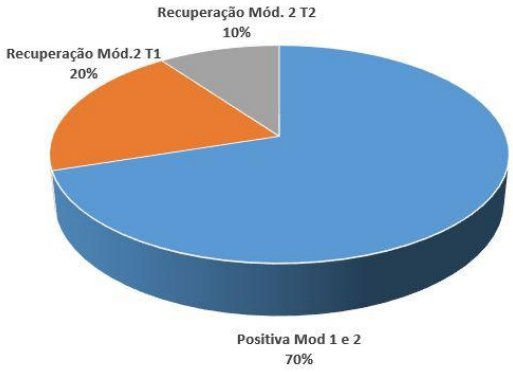
Distribuição da ASE pela turma



Até ao início da intervenção pedagógica, todos os alunos têm positiva ao módulo 1. No que diz respeito ao módulo 2, 21 alunos têm positiva e 9 alunos estão em recuperação. Destes 9 alunos, 6 pertencem ao T1 e 3 ao T2 (vide **Figura *8***).

Figura 8

Situação da turma até à intervenção pedagógica



O levantamento de informações acerca da turma, foi realizado através de uma reunião com a Professora Cooperante, com a diretora de turma e com a consulta dos processos dos alunos.

3. Enquadramento Curricular

A intervenção ocorreu no Curso Profissional de Técnico de Programação de Sistemas de Informáticos (TGPSI), na disciplina de Programação e Sistemas de informação (PSI) no módulo 4, Estruturas de dados Estáticas. Este módulo aborda os conceitos de cadeias de carateres (*Strings*), *Arrays* unidimensionais (vetores) e *Arrays* multidimensionais (matrizes).

3.1. Matriz Curricular do Curso Profissional

O curso TGPSI foi criado pela Portaria n.º 916/2005, de 26 de setembro (Portaria nº916, 2005). Este curso visa a saída profissional de técnico de gestão e programação de sistemas informáticos e integra-se na área de educação e formação de Ciências Informáticas (481). O Perfil de desempenho à saída do curso é o profissional qualificado apto a realizar, de forma autónoma ou integrado numa equipa, atividades de conceção, especificação, projeto, implementação, avaliação, suporte e manutenção de sistemas informáticos e de tecnologias de processamento e transmissão de dados e informações.

Os Cursos Profissionais são um percurso de ensino secundário com dupla certificação, ou seja, em que se desenvolvem competências sociais, científicas e profissionais necessárias ao exercício de uma atividade profissional e simultaneamente se obtém o nível secundário de educação. Estes cursos preparam os jovens para uma mais fácil e qualificada inserção no mercado de trabalho e permitem a realização de estudos ao nível pós-secundário e ensino superior (ANQUEP, 2018).

Os Cursos Profissionais integrados no Catálogo Nacional de Qualificações têm duração de três anos, com uma carga horária que varia entre 3100 e 3440 horas e estão organizados em quatro componentes de formação (Portaria n.º 235-A, 2018):

- **Formação Sociocultural -** estruturada em disciplinas comuns a todos os cursos, visa contribuir para a construção de identidade pessoal, social e cultural dos alunos;

- **Formação Científica -** estruturada em duas ou três disciplinas, visa proporcionar uma formação científica consistente com a qualificação a adquirir;
- **Formação Tecnológica -** organizada em Unidades de Formação de Curta Duração (UFCD), visa a aquisição e desenvolvimento de um conjunto de competências técnicas necessárias ao exercício profissional;
- **Formação em Contexto de Trabalho -** é realizada em empresas ou noutras organizações, em períodos de duração variável ao longo ou no final da formação, e visa a aquisição e o desenvolvimento de competências técnicas, relacionais e organizacionais relevantes para a qualificação profissional.

A **Figura 9**, apresenta a carga horária das quatro componentes do curso profissional TGPSI.

Figura 9

Plano de Estudos Curso profissional segundo Dec. Lei 55/2018

Cursos profissionais

Ensino secundário

Componentes de formação		Carga horária Ciclo de formação (horas) (a)
Sociocultural:		
Português		320
Língua Estrangeira I, II ou III (b)		220
Área de Integração		220
Tecnologias de Informação e Comunicação/Oferta de Escola (c)		100
Educação Física		140
Subtotal		1000
Científica:		
Duas a três disciplinas (d)		500
Tecnológica:		
UFCD (e)		1000 a 1300
Formação em contexto de trabalho		600 a 840
Educação Moral e Religiosa (g)		(g)
Total (h)		3100 a 3440

Cidadania e Desenvolvimento (f)

O Decreto Lei n.º 55, (2018), indica que a carga horária da Componente de Formação Tecnológica (CFT) dos cursos profissionais do ensino secundário varia entre 1000 e 1300 horas e está organizado em UFCD, no entanto, no AE4O, o curso TGPSI ainda mantém a estrutura da componente de formação tecnológica organizada em módulos com uma carga horária de 1100 horas distribuídas pelos 3 anos do curso.

No AE4O, Formação em Contexto de Trabalho (FCT), possui uma carga horária de 600horas, e CFT é constituída pelas seguintes disciplinas:

- Arquitetura de Computadores, com carga horária de 130 horas;
- Sistemas Operativos, com uma carga horária de 140 horas;
- Redes de Computadores, com uma carga horária de 252 horas;
- Programação e Sistemas de Informação, com uma carga horária de 578 horas.

Estes cursos culminam com uma apresentação e defesa, perante um júri, de um projeto, designado por Prova de Aptidão Profissional (PAP), na qual são demonstradas as competências e os conhecimentos que desenvolveram ao longo da formação (ANQUEP, 2018).

No final do curso, os alunos obtêm uma dupla certificação - o ensino secundário e uma certificação profissional - conferindo o nível 4 de qualificação do Quadro Nacional de Qualificações (Portaria n.º 782, 2009).

A conclusão do ensino secundário nos Cursos Profissionais está dependente da aprovação em todas as disciplinas e módulos, na formação em contexto de trabalho e na Prova de Aptidão Profissional. Após a finalização do curso, os alunos deverão desempenhar as seguintes tarefas:

- Instalar, configurar e efetuar a manutenção de computadores isolados ou inseridos numa rede local;
- Instalar, configurar e efetuar a manutenção de periféricos de computadores ou de uma rede local;

- Instalar, configurar e efetuar a manutenção de estruturas e equipamentos de redes locais;
- Instalar, configurar e efetuar a manutenção de sistemas operativos de clientes e de servidores;
- Implementar e efetuar a manutenção de políticas de segurança em sistemas informáticos;
- Instalar, configurar e efetuar a manutenção de aplicações informáticas;
- Efetuar a análise de sistemas de informação;
- Conceber algoritmos através da divisão dos problemas em componentes;
- Desenvolver, distribuir, instalar e efetuar a manutenção de aplicações informáticas, utilizando ambientes e linguagens de programação procedimentais e visuais;
- Conceber, implementar e efetuar a manutenção de bases de dados;
- Manipular dados retirados de bases de dados;
- Instalar, configurar e efetuar a manutenção de servidores para a Internet;
- Planificar, executar e efetuar a manutenção de páginas e sítios na Internet;
- Desenvolver, instalar e efetuar a manutenção de sistemas de informação baseados nas tecnologias web.

3.2. A Disciplina Programação e Sistemas de Informação

A disciplina de PSI, integra a componente de formação técnica dos cursos profissionais, de forma a garantir aos jovens a aprendizagem de técnicas de programação e desenvolvimento de sistemas informáticos, indispensáveis ao sucesso pessoal e profissional nesta área (Direcção-Geral de Formação Vocacional, 2005).

A disciplina de PSI tem diversas finalidades, das quais destaco:

- Desenvolver a capacidade de analisar de forma objetiva as linguagens de programação existentes;
- Fomentar a capacidade de compreender as técnicas básicas de implementação de linguagens de programação, e desenvolver uma

capacidade acrescida de aprender novas linguagens de programação, assim como uma acrescida capacidade de conceção e desenvolvimento de *software* e sistemas de informação;

- Desenvolver a capacidade de análise de problemas reais da área da informática, e ser capaz de desenvolver soluções de *software* que permitam colmatar as necessidades verificadas;
- Promover a autonomia, a criatividade, a responsabilidade, bem como a capacidade para trabalhar em equipa numa perspetiva de abertura à mudança, à diversidade cultural e ao exercício de uma cidadania ativa;
- Fomentar o interesse pela pesquisa, pela descoberta e pela inovação, face aos desafios da sociedade do conhecimento.

3.3. Conteúdos Programáticos da Disciplina

A disciplina PSI é distribuída por 16 módulos obrigatórios (542 horas) e 3 módulos com tema opcional (90 horas), (vide **Figura 10**), a ser selecionado entre 7 temas possíveis, que serão escolhidos de acordo com o Projeto Educativo da Escola.

Figura 10

Elenco modular disciplina PSI antes Dec Lei 55/2018

Número	Designação (obrigatórios)	Duração de referência (horas)
1	Introdução à Programação e Algoritmia	36
2	Mecanismos de Controlo de Execução	36
3	Programação Estruturada	36
4	Estruturas de Dados Estáticas	30
5	Estruturas de Dados Compostas	30
6	Estruturas de Dados Dinâmicas	36
7	Tratamento de Ficheiros	30
8	Conceitos Avançados de Programação	18
9	Introdução à Programação Orientada a Objectos	36
10	Programação Orientada a Objectos	36
11	Programação Orientada a Objectos Avançada	30
12	Introdução aos Sistemas de Informação	21
13	Técnicas de Modelação de Dados	36
14	Linguagem de Manipulação de Dados	36
15	Linguagem de Definição de Dados	21
16	Projecto de Software	74
17 (1)	Tema opcional	30
18 (1)	Tema opcional	30
19 (1)	Tema opcional	30

(1) Os temas destes módulos deverão ser seleccionados de entre os sete **módulos opcionais** apresentados no quadro da página seguinte.

A **Figura 11**, apresenta os módulos opcionais.

Figura 11

Módulos Opcionais

Módulos opcionais

Número	Designação	Duração de referência (horas)
OP1	Tecnologias de Acesso a Bases de Dados	30
OP 2	Técnicas de Detecção e Tratamento de Erros	30
OP 3	Metodologias de Análise e Desenvolvimento de Sistemas	30
OP 4	Conceitos de Organização e Gestão de Empresas	30
OP 5	Ferramentas de Desenvolvimento de Páginas *Web*	30
OP 6	Ferramentas de Animação Gráfica	30
OP 7	Ferramentas de Tratamento de Imagem	30

No conjunto dos 16 módulos, a disciplina aborda em três conceitos distintos. Dos módulos 1 ao 7 são dedicados aos conceitos iniciais da programação, tais como, algoritmia, fluxogramas, variáveis, estruturas de decisão, estruturas de repetição e *arrays*, essencialmente os módulos são dedicados à programação numa linguagem estruturada.

O módulo 8 introduz conceitos de programação avançada, tais como, conceito e janela, interface com o utilizador, programação por eventos, *queues* e multitarefa.

Dos módulos 9 ao 11, são dedicados aos conceitos da programação orientada por objetos, tais como, classes, atributos, métodos, eventos, herança, polimorfismo e tratamento de exceções.

Dos módulos 12 ao 15 são dedicados aos conceitos iniciais de Sistemas de informação, tais como, armazenamento de dados, modelo entidade-relação, base de dados, tabela, índice, chaves primárias, chaves estrangeiras, normalização de base dados, linguagem de definição de dados (CREATE, ALTER TABLE), linguagem de manipulação de dados (SEELCT, INSERT TABLE).

A **Figura 12**, apresenta o elenco modular da disciplina de PSI adotado na ESACF, onde é apresentado para cada módulo, o número de horas, o ano letivo em que é lecionado e quais as linguagens ou ferramentas utilizadas. A carga horária da componente tecnológica está de acordo com o Dec. Lei 55/2108.

Figura 12

Elenco Modular disciplina PSI adotado pelo AE4O

Nº Módulo	Designação do Módulo	Nº horas	Ano Letivo	Linguagens Programação / Tecnologias utilizadas
1	Introdução à Programação e Algoritmia	20	10º	visualG
2	Mecanismos de Controlo de Execução	53	10º	C
3	Programação Estruturada	20	10º	C
4	Estruturas de Dados Estáticas	32	10º	C
5	Estruturas de Dados Compostas	26	10º	C
6	Estruturas de Dados Dinâmicas	11	10º	C
7	Tratamento de Ficheiros	18	10º	C
8	Conceitos Avançados de Programação	18	11º	C#
9	Introdução à Programaşão Orientada a Objectos	36	11º	C#
10	Programação Orientada a Objectos	36	11º	C#
11	Programação Orientada a Objectos Avançada	30	11º	C#
12	Introdução aos Sistemas de Informação	21	11º	Teoria
13	Técnicas de Modelação de Dados	36	11º	Teoria
14	Linguagem de Manipulação de Dados	36	11º	Teoria SQL
15	Linguagem de Definição de Dados	21	11º	SQL / XAMPP
16	Projecto de Software	74	12º	C# / SQL / XAMPP
17	Tecnologias de Acesso a Bases de Dados	30	12º	PHP
18	Ferramentas de desenvolvimento de Páginas Web	30	12º	CSS
19	Ferramentas de Tratamento de Imagem	30	12º	Photoshop

A minha observação decorreu nos módulos 1, 2 e 3, são eles Introdução à Programação e Algoritmia, Mecanismos de Controlo e Execução e Programação estruturada, respetivamente. A intervenção ocorreu no início do módulo 4, Estruturas de dados Estáticas.

3.3.1. Módulo 1 – Introdução à Programação

Este módulo é o primeiro da disciplina, tem duração de 36 horas, e tem como função principal dar ao aluno um conhecimento do funcionamento lógico de um programa. A algoritmia é a base essencial para a programação, este módulo permite o estímulo do raciocínio lógico e prepara os alunos para a resolução de problemas de programação mais ou menos complexos. Serão, também, abordados conceitos de

algoritmo, de sequência lógica, pseudocódigo, fluxogramas e os diferentes operadores e tipos de dados utilizados num programa. Este módulo valoriza também o pensamento sistemático e estruturado de resolver os problemas. Introduz também o conceito de Entrada e Saída de dados e interação com o utilizador (Direcção-Geral de Formação Vocacional, 2005).

Neste módulo, os alunos ainda não têm contacto com nenhuma linguagem de programação. O contacto mais próximo com linguagem de programação é na utilização da aplicação VisuAlg.

O VisuAlg, é uma aplicação Windows que permite editar, interpretar e executar algoritmos com uma linguagem próxima do português estruturado. É uma ferramenta ideal para aprender técnicas de elaboração de algoritmos, utilizando uma linguagem simples semelhante ao "Portugol". A interface gráfica do Visual G, é semelhante a editores de código usados na maioria das linguagens de programação.

Os principais conceitos abordados no módulo 1, são os seguintes:

- **Lógica Programação -** Técnica de encadear pensamentos para atingir determinado objetivo;
- **Sequência Lógica** – Passos executados até atingir um objetivo ou solução de um problema;
- **Instrução** – é um comando no qual o programador indica ao processador do computador para realizar uma determinada tarefa.
- **Algoritmo** – Sequencia finita de instruções descritas de forma lógica, ordenada e clara para resolver um problema.
- **Linguagem Natural** – Linguagem projetada falada ou escrita, (português ou Inglês), é passível de várias interpretações;
- **Fluxograma** – representação de um fluxo de ações de um programa através e símbolos, em que cada símbolo represente diferentes tipos de ações e o seu encadeamento na sequência do programa;
- **Pseudocódigo** – Representação textual de um algoritmo por palavras da linguagem natural, não segue nenhuma sintaxe de linguagem de programação;

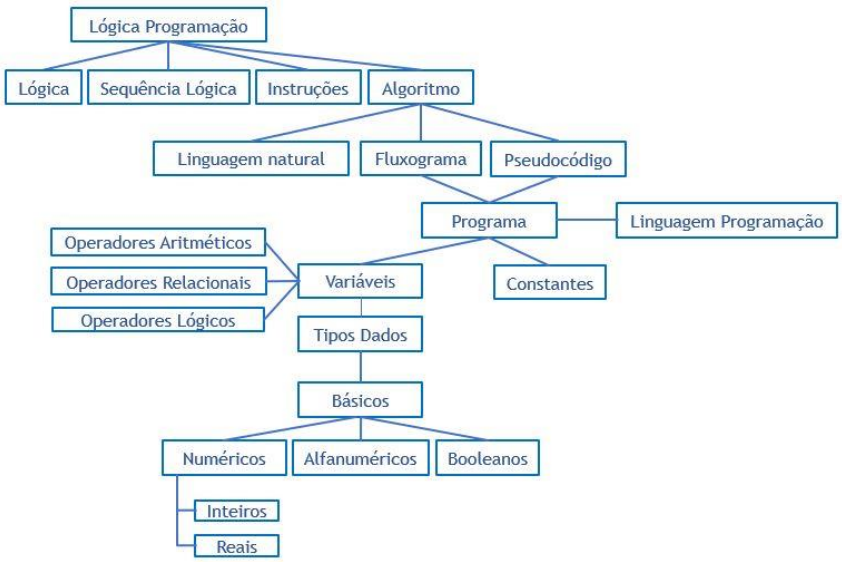
- **Linguagem de Programação** – São um conjunto de instruções que seguindo regras sintáticas, permitem criar um programa, orientando um computador a resolver um determinado problema;
- **Programa** – Um programa é a implementação de um algoritmo numa determinada linguagem de programação. São um conjunto de instruções ou código fonte que é escrito para ser executado por um computador. Um programa informático é criado com o objetivo de resolver um problema, realizar tarefas especificas, sob a forma de automatizar processos ou processar dados fornecidos pelo utilizador;
- **Variáveis** – são estruturas usadas na programação para armazenar valores. Esses valores podem ser modificados durante a execução de um programa;
- **Constantes** – são caso particular das variáveis, em que é uma estrutura que armazena um valor, mas que não pode ser modificado durante a execução de um programa;
- **Operadores Aritméticos** – usados na realização de cálculos aritméticos simples usando as quatro operações básicas da matemática (soma, subtração, multiplicação e divisão) mais a operação de exponenciação e do módulo;
- **Operadores Relacionais** – usados para estabelecer uma relação entre duas variáveis (operandos) que sejam do mesmo tipo. É comparada a variável à esquerda e a variável à direita do operador e é retornado um valor Booleano/Lógico.
- **Operadores Booleanos/Lógicos** – Recebem valores Booleanos como entrada e retornam também valores Booleanos;
- **Tipos de dados** – São categorias que definem quais os tipos de valores que são armazenados nas variáveis. Cada linguagem de programação possui os seus tipos de dados, no entanto a maioria das linguagens possui os seguintes tipos básicos:
    - **Numérico Inteiro** – Representa números inteiros sem parte fracionária;

- **Numérico Real** - Representa números com parte fracionada;
- **Alfanumérico** – represente uma sequência de caracteres;
- **Booleano** – Representa valor lógico (verdadeiro ou falso).

A **Figura 13**, apresenta o mapa de conceitos do módulo 1 relacionados entre si.

Figura 13

Mapa de Conceitos do Módulo 1



3.3.2. Módulo 2 – Mecanismos de Controlo e Execução

O segundo módulo, tem duração de 36 horas, serão abordadas as diferentes estruturas de controlo existentes numa linguagem de programação. Serão também abordados os mecanismos de repetição, sendo dado especial ênfase na sua utilização no mundo da programação bem como as várias combinações existentes.

Serão propostos aos alunos diversos métodos para alcançar os mesmos objetivos, promovendo a discussão das vantagens e desvantagens das várias soluções. Este tipo de exercícios visa desenvolver o espírito crítico e os mecanismos de autonomia de pensamento do aluno (Direcção-Geral de Formação Vocacional, 2005).

Os principais conceitos abordados no módulo 2, são os mecanismos de controlo de execução, dos quais destaco:

- **Input** – são instruções que permitem ler, dados fornecidos pelo utilizador, dados de ficheiros ou outros dispositivos (sensores);
- **Output** – são instruções que permitem enviar dados ou mensagens para o ecrã, mas também gravar informações em ficheiros ou outros dispositivos;
- **Afetação** – são instruções usadas para atribuir valores a uma variável;
- **Condição** – é uma expressão que determina se uma instrução ou bloco de código, quando avaliado, resulta num valor booleano;
- **Decisão** – são instruções que servem para controlar o fluxo de um programa. Permitem que um programa tome decisões lógicas e execute diferentes blocos de código dependendo se uma condição é verdadeira ou falsa;
- **Repetição** – são instruções também conhecidas como *loops*, são utilizadas na programação para executar um bloco de código repetidamente enquanto uma dada condição for verdadeira.

A **Figura 14**, apresenta o mapa de conceitos do módulo 2.

Figura 14

Mapa de Conceitos do Módulo 2

Segundo o programa da disciplina, a bibliografia sugerida para o módulo 2, sugere três linguagens de programação: Pascal, C e Java. O AE4O adota a linguagem C nos módulos 1 a 7 da disciplina de PSI.

Segundo Damas, (2007), a linguagem C foi criada em 1972 nos *Bell Telephone Laboratories* por Dennis Ritchie com finalidade de permitir a escrita de um sistema operativo, Unix, utilizando uma linguagem de relativo alto nível, evitando recurso ao *Assembly*. Devido às suas capacidades e através da divulgação do sistema operativo Unix pelas universidades dos Estados Unidos, a linguagem C rapidamente disseminou-se e tornou-se conhecida por todos os tipos de programadores.

Esta disseminação levou a que diferentes organizações desenvolvessem e utilizassem diferentes versões da linguagem C, criando desta forma problemas de portabilidade, entre outros. Perante este cenário o American National Standards Institute (ANSI), formou em 1983uma comitê para a definição de um padrão da linguagem C.

O nome da linguagem C, resulta da evolução de uma outra linguagem de programação desenvolvida por Ken Thompson também nos Laboratório Bell, chamada B. Desta forma a linguagem C é a evolução natural da linguagem B.

3.3.3. Módulo 3 – Programação Estruturada

O módulo 3, tem duração de 36 horas, destina-se a dar aos alunos uma visão global da estruturação de programas. Pretende-se que os alunos compreendam que a utilização de subprogramas permite a aplicação dos princípios da programação estruturada assim como a reutilização de código escrito.

Neste módulo os alunos devem conhecer as regras de declaração e utilização de subprogramas assim como controlar o ciclo de vida das variáveis. Os alunos devem tomar consciência da independência dos subprogramas relativamente aos programas através do uso da parametrização. Estes conceitos visam encaminhar os alunos para soluções mais eficientes e racionais e promover a divisão de problemas em componentes simples como meio de solução de problemas complexos (Direcção-Geral de Formação Vocacional, 2005).

O principal tema abordado no módulo 3, é a programação estruturada, dentro da programação estruturada, destaco os seguintes conceitos:

- **Subprograma** - conhecido como procedimento ou função, é um bloco de código dentro de um programa maior que executa uma tarefa específica e pode ser chamado (invocado) a partir de outras partes do programa. Facilita a reutilização de código e sua manutenção. Existem dois tipos principais de subprogramas: procedimento e função;
- **Procedimento** – um subprograma que pode, ou não, receber parâmetros de entrada e não retorna nenhum valor;
- **Função** – um subprograma que pode, ou não, receber parâmetros de entrada e retorna um valor;
- **Parâmetro** – Valor ou variável que pode ser recebido por um subprograma. Permite a flexibilidade e reutilização de funções ou procedimentos porque permitem invocar funções/procedimentos com valores diferentes a cada invocação. Existem duas formas de passar um parâmetro, por valor ou por referência;
- **Passagem de Parâmetros Por Valor** – É passada para a função ou procedimento uma cópia do valor da variável. Caso haja alguma

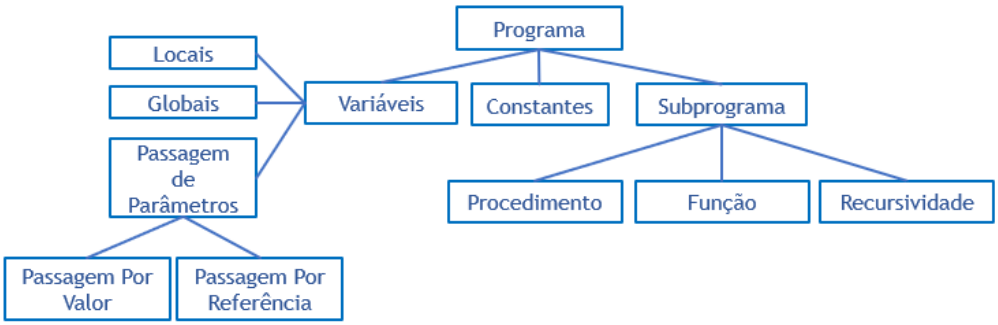
modificação do valor da variável dentro da função, não afeta valor da variável fora da função;

- **Passagem de Parâmetros Por Referência** - É passada para a função ou procedimento o endereço de memória da variável, permite que uma função possa modificar o valor da variável dentro da função;
- **Variáveis Locais** – são variáveis declaradas dentro de uma função ou procedimento. Só podem ser acedidas dentro do bloco de código da função/procedimento, são criadas quando a execução do programa entra na função/procedimento, e são destruídas quando a execução sai da função/procedimento;
- **Variáveis Globais** - são variáveis declaradas no corpo principal do programa. Podem ser acedidas em qualquer parte do código (dentro ou fora da função/procedimento), e mantêm o seu valor durante a execução do programa;
- **Recursividade** – Técnica usada quando uma função se invoca a si própria. É importante ter em atenção quando se recorre a esta técnica para evitar um *loop* infinito.

A **Figura 15**, apresenta o mapa de conceitos do módulo 3.

Figura 15

Mapa de Conceitos do Módulo 3



No final do módulo 3, os alunos deverão ter assimilado os conceitos científicos acima apresentados.

3.3.4. Módulo 4 – Estrutura de Dados Estáticas

O módulo 4, tem duração de 30 horas, e o objetivo deste módulo é o de introduzir o conceito de estrutura de dados como o mecanismo que permite o armazenamento de dados. Serão introduzidos os conceitos básicos, bem como os algoritmos de criação e manipulação dos mesmos. Estes conceitos permitiram ao aluno complementar os seus conhecimentos e resolver progressivamente problemas mais complexos (Direcção-Geral de Formação Vocacional, 2005).

A minha intervenção ocorreu no início do módulo 4, Estruturas de Dados Estáticas, onde destaco os seguintes conceitos que foram abordados na minha intervenção:

- **Vetor ou *Array* –** Um vetor, também conhecido por *array* é um conjunto de elementos consecutivos, do mesmo tipo, que podem ser acedidos individualmente a partir de um único nome.

Exemplo: **notas**

12	10	8	11	15	16

Cada posição do vetor corresponde a uma nota, desta forma evita-se a criação de uma variável para cada aluno, colocando todas as informações das notas numa variável só;

- **Declaração de um vetor**– Na linguagem C, um vetor é declarado da mesma forma que uma variável simples, e obedece à seguinte sintaxe:

```
tipo nome_variável[N° de elementos]
```

Exemplo: `int notas[5];`

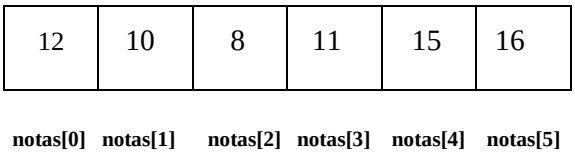
Neste exemplo todos os elementos são identificados pelo mesmo nome (`notas`), para que se possa identificar um elemento

individualmente (**valor indexado**) é necessário um número (**índice**) que indique qual a sua posição no vetor;

- **Tipo** – Corresponde ao tipo de dados de cada um dos elementos o vetor;
- **Nome_variável** – Indica o nome pelo qual esse vetor vai ser conhecido;
- **Nº de elementos –** Valor constante que indica quantos elementos tem o vetor;

- **Índice** – Número que indica qual a posição onde está o elemento. Na linguagem C, os índices de um vetor com **`n`** elementos variam sempre entre o índice **`0`** e **`n-1`**. O índice do primeiro elemento de qualquer vetor em C é sempre 0 (zero). Cada uma das posições do vetor pode ser acedida através do respetivo índice colocado entre **parenteses retos [ ]**.

Exemplo: **notas**



- **Valor indexado** – É o conteúdo armazenado numa posição específica do vetor determinada pelo índice. Tomando de exemplo o vetor acima apresentado:

```
Notas[0] = 12
Notas[1] = 10
…
Notas[5] = 16
```

- **Inicialização de um Vetor** –Na linguagem C, é possível declarar e ao mesmo inicializar automaticamente um vetor, usando a seguinte sintaxe

  tipo nome_variável[n] = {Valor$_1$, valor$_2$, ..., valor$_N$}

  O seguinte exemplo, declara e inicia um vetor com vogais do alfabeto:

  ```
  char vogais[5] = {"a", "e", "i", "o", "u"};
  ```

  Evita-se, assim, de escrever as várias linhas de código:

  ```
  char vogais[5];
  vogais[0] = "a";
  vogais[1] = "e";
  vogais[2] = "i";
  vogais[3] = "o";
  vogais[4] = "u";
  ```

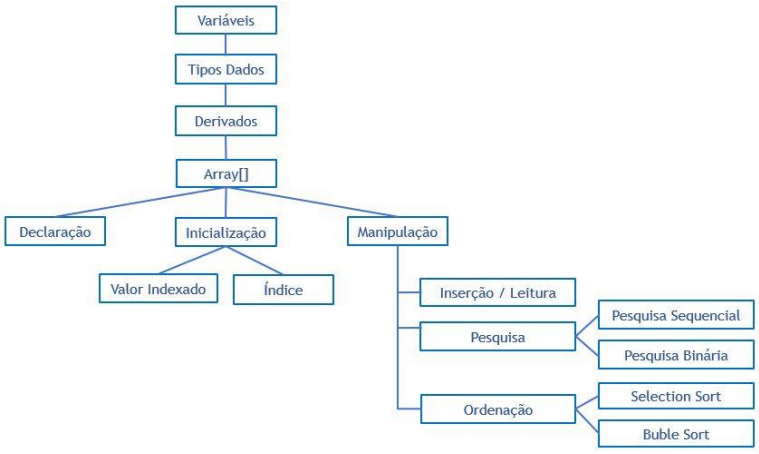
- **Pesquisa** - A pesquisa de um elemento num *array*, é uma tarefa frequente na programação, e há várias formas de a fazer, dependendo das necessidades. Apresento duas abordagens para fazer a pesquisa:
  - **Pesquisa Linear** – Consiste em percorrer o vetor até encontrar o elemento desejado, método simples, contudo, pode ser ineficiente para vetores de grandes dimensões;
  - **Pesquisa Binária** – Consiste em dividir o vetor pela metade em cada iteração, mas só é possível aplicar esta abordagem em vetores ordenados.
- **Ordenação** – Existem vários algoritmos de ordenação de vetores, cada um com as suas próprias características e eficiência em situações diferentes, dos quais destaco:

- **Ordenação por Bolha (Bubble Sort)** – Consiste na comparação e troca de elementos adjacentes, não é eficiente para vetores de grandes dimensões;
- **Ordenação por seleção (Selection Sort)** – Seleciona o menor elemento e coloca-o na primeira posição do vetor. Menos eficiente que algoritmos mais avançados. Na minha intervenção, por questões de limitação de tempo, apenas irei lecionar este algoritmo de ordenação;
- **Ordenação Rápida (Quick Sort)** – Escolhe um elemento do vetor como pivô, e reorganiza o vetor para que os elementos menores fiquem à esquerda e maiores à direita. Eficiente para grandes conjuntos de dados;
- **Ordenação por junção (Merge Sort)** – Consiste em dividir o vetor em duas partes, ordena cada uma delas e junta as duas partes. É eficiente para grandes conjuntos de dados Esau Taiwo et al., (2020).

A **Figura 16**, apresenta o mapa de conceitos do módulo 4.

Figura 16

Mapa Conceitos Módulo 4



Os conceitos acima mencionados foram descritos com recurso ao livro (Damas, 2007).

3.4. Análise Critica ao Programa da Disciplina

O programa da disciplina de PSI compreende um total de 19 módulos e uma carga horária total de 632 horas, sendo que 16 módulos constituem a base do programa, enquanto os restantes três são opcionais, cujo tema deve ser selecionado entre sete módulos alternativos e consoante as necessidades dos alunos e do mercado de trabalho (Direcção-Geral de Formação Vocacional, 2005).

Após a leitura do programa da disciplina de PSI destaco alguns aspetos positivos no mesmo, tal como é o caso da coerência entre as competências a desenvolver na disciplina e os seus objetivos, devidamente alinhadas também com as orientações acerca do processo de ensino e aprendizagem mais recentes (Perfil dos alunos à saída da Escolaridade Obrigatória, 2016).

Destaco também, como aspeto positivo do programa, as técnicas e os métodos selecionados para a avaliação dos alunos desta disciplina. Mais concretamente, é sugerido que os professores efetuem, no início do ano letivo, um teste diagnóstico para identificar as competências e os conhecimentos dos alunos, no sentido de delinear um plano de ação adequado para o nível de competências e de aprendizagem da turma no geral. Para além da avaliação diagnóstica, é igualmente sugerido, a observação direta sobre o trabalho desenvolvido pelo aluno durante as aulas, e momentos de avaliação que permitam avaliar os conhecimentos e competências adquiridos.

Outro aspeto positivo é, a compreensão dos módulos deve ser reforçada durante o processo de aprendizagem dos alunos, pois ao partir da base dos seus conhecimentos e competências os professores acabam por ter uma melhor noção dos conteúdos adquiridos pelos alunos. Logo, esta técnica permite uma melhor perceção dos conhecimentos atuais dos alunos, bem como delinear as planificações dos professores para as aulas do ano letivo que se inicia, promovendo sempre a aquisição de novo conteúdo e competências.

Por último, outro aspeto positivo, no decurso do ano letivo em si, os métodos e técnicas de avaliação são também pertinentes, no sentido em que se pretende obter um feedback constante ao longo do processo de aprendizagem. Regra geral, os métodos de avaliação baseiam-se na observação direta do trabalho desenvolvido em

contexto de sala de aula, assim como na vertente formativa, relacionada com a realização de testes de avaliação. Desta forma, os professores conseguem percecionar o nível de aquisição e de desenvolvimento das novas competências por parte dos seus alunos, registando a evolução de cada um ao longo do ano letivo.

No que diz respeito a aspetos negativos, destaco, que a bibliografia sugerida, apesar de extensa, nos vários módulos foi publicada há cerca de vinte anos ou mais, o que denota a ineficácia do programa em atualizar-se consoante as evoluções e modificações na nova era digital e de informação.

Outro aspeto negativo, e diretamente associado ao primeiro, está relacionado com o próprio processo de aprendizagem dos alunos, que acaba por ser algo limitado, visto que não engloba inovações tecnológicas utilizadas nos dias de hoje. Logo, sugere-se a atualização do programa da disciplina de PSI, integrando as várias inovações tecnológicas mais recentes, no sentido de dotar os alunos de conhecimentos atuais e passíveis de serem aplicados na prática, bem como no mercado de trabalho.

Como referi acima, o programa da disciplina sugere a utilização de três linguagens de programação no ensino dos módulos 1 ao 8, são elas: Pascal, C e Java. Dado que o AE4O adota a linguagem C, e que em minha opinião a bibliografia da disciplina é datada, fui pesquisar na internet acerca das características e potencialidades da linguagem C.

Segundo o autor, Damas, (2007), a linguagem C é extremamente potente e flexível, possuindo as seguintes características:

- **Simples -** A sua sintaxe é extremamente simples, com um número reduzido de palavras-chave, de tipos básicos e operadores, reduzindo desta forma quantidade de tempo e esforço necessários à aprendizagem da linguagem;
- **Rapidez** – Consegue obter performances semelhantes às obtidas pelo *Assembly*. Através de instruções de alto nível;
- **Portável** – Existe um padrão (ANSI) que define as características de qualquer compilador. Deste modo código escrito numa máquina pode

ser transportado para outra máquina e compilado sem qualquer alteração;

- **Popular** – É internacionalmente conhecida e utilizada, está muito bem documentada em livros, revistas da especialidade. Existem compiladores para todo o tipo de arquiteturas de computadores;
- **Modular** – Permite o desenvolvimento modular de aplicações, facilitando a separação de projetos em módulos distintos e independentes, recorrendo à utilização de funções específicas dentro de cada módulo;
- **Alto Nível -** É considerada uma linguagem de terceira geração, ou seja, linguagem de alto nível quando comparada com a linguagem *Assembly*. Permite ainda o acesso à maior parte das funcionalidades de *Assembly*, utilizando expressões e instruções de alto nível;
- **Bibliotecas Poderosas** – Pelo facto de o C possuir um número reduzido de palavras-chave, as suas capacidades não são limitadas. A maior parte das funcionalidades de linguagem C, é adicionada pela utilização de funções que existem em bibliotecas adicionais e realizam todo o tipo de tarefas, desde a escrita de um caractere no ecrã, ou até ao processamento de *strings*, entre outras;
- **Evolução** – É uma linguagem estável. A evolução das linguagens fez com que também evoluísse no sentido das linguagens Orientadas a Objetos, dando origem a uma nova linguagem: C++, a qual mantém a sintaxe da linguagem C e permite um conjunto adicional de características, tais como, Encapsulamento, Herança, Polimorfismo e Abstração. A linguagem Java, também se baseia na linguagem C e C++.

3.5. Avaliação da Disciplina

No que diz respeito à avaliação da disciplina, segundo o programa (Direcção-Geral de Formação Vocacional, 2005), sugere que no início do ano letivo seja realizada

uma avaliação diagnóstica que permita identificar grupos diferenciados e estabelecer um plano de ação para cada grupo de alunos, tendo em vista a aquisição, por parte de todos eles, competências essenciais definidas no programa.

Deverá ser privilegiada a observação direta do trabalho desenvolvido pelo aluno durante as aulas, utilizando para isso instrumentos de avaliação diversificados que permitam registar o seu desempenho nas situações que lhe são proporcionadas e a progressão na aprendizagem ao longo do ano letivo, nomeadamente quanto ao interesse e à participação no trabalho, à capacidade de desenvolver trabalho em grupo, à capacidade de explorar, investigar e mobilizar conceitos em diferentes situações, bem como relativamente à qualidade do trabalho realizado e à forma como o aluno o gere, organiza e autoavalia.

A par da avaliação contínua, permitindo o registo da evolução do aluno aula a aula e a recuperação, em tempo útil, de qualquer dificuldade, deverão ser previstos momentos de avaliação, procedendo-se à aplicação de provas de carácter prático ou teórico-prático que permitam avaliar os conhecimentos e competências adquiridos.

Os critérios gerais de avaliação do AE4O para a disciplina de PSI, (vide **Anexo B**), são constituídos por duas competências:

- **Conhecimentos e capacidades conceptuais e factuais**
- **Conhecimentos e capacidades processuais e comunicacionais**

A competência dos Conhecimentos e capacidades conceptuais e factuais resultará em 65% da avaliação e os restantes 35% resultarão da avaliação da competência dos conhecimentos e capacidades processuais e comunicacionais.

Os instrumentos de recolha de informação para as duas competências, acima citadas, são: Teste; Trabalho de pesquisa; Procedimento prático; Trabalho de projeto; Relatório de projeto; Apresentação oral e Observação direta.

A Professora Cooperante utiliza na disciplina PSI os seguintes instrumentos: o teste; fichas de avaliação formativa; trabalhos realizados individualmente e em grupo; observação diária dos alunos; e a autoavaliação.

3.6. Planificação do Professor Cooperante

A planificação anual do Professor Cooperante, como mostra o **Anexo C**, apresenta os objetivos, as estratégias, os recursos e o número de aulas atribuídas aos módulos 1, 2, 3 e 4.

3.7. Dificuldades do Ensino da Temática

A programação de computadores é uma das disciplinas da ciência da computação que normalmente apresenta altos índices de desistência e retenção de alunos. A literatura aponta vários fatores que contribuem para desistências e retenções. A dificuldade de compreensão de conceitos abstratos e o método de ensino atual baseado em aulas tradicionais com baixa interação entre os alunos geram baixa motivação e consequentemente o desinteresse em aprender programação de computadores (Piteira & Haddad, 2011).

A programação pode ser um processo doloroso para os programadores principiantes. Devem possuir conhecimentos declarativos e procedimentais, memorização, compreensão, resolução de problemas, capacidade de abstração e raciocínio lógico, entre outros (Piteira & Costa, 2013).

Tendo por base, o módulo das Estruturas de Dados Estáticas da disciplina de PSI, constata-se que, de acordo com o programa (Direcção-Geral de Formação Vocacional, 2005), este é lecionado com base na programação em Linguagem C e corresponde a dois conceitos específicos: *Strings* e *Arrays*. De facto, neste módulo específico o objetivo é dotar os alunos de conhecimento sobre o conceito de Estruturas de Dados Estáticas enquanto mecanismo de armazenamento de dados, pressupondo que os módulos anteriores foram devidamente aprendidos e interiorizados pelos alunos.

Além disso, o próprio programa da disciplina de PSI recomenda, nas orientações de competências a desenvolver pelos alunos, o domínio da programação orientada para os objetos. Ora, como existe uma orientação clara para a utilização da

linguagem C, que é bastante complexa por si só, constata-se que não existe uma congruência com o paradigma de programação.

Para além desta dificuldade, é possível que o desempenho dos alunos nos módulos anteriores não tenha sido o melhor, o que irá impactar negativamente a aprendizagem de módulos mais avançados, dado que estes últimos pressupõem o domínio dos módulos anteriores.

Deste modo, é possível que os alunos se sintam desinteressados no processo de aprendizagem, pois não dominam os conceitos prévios, o que dificulta a aquisição de novos conhecimentos de programação, especialmente da linguagem C.

O programa da disciplina de PSI também esclarece que se deve privilegiar a participação dos alunos em projetos vários, baseados na resolução de problemas e na simulação da realidade. Contudo, o programa não define orientações concretas no concernente aos recursos e metodologias a seguir nas várias atividades (Direcção-Geral de Formação Vocacional, 2005).

Por conseguinte, é crucial promover outros métodos e recursos no ensino de PSI entre os alunos do 10º ano, no sentido de facilitar a compreensão dos conteúdos lecionados, bem como a aplicação do seu conhecimento em atividades práticas.

Neste sentido, destaco o argumento apresentado por Piedade et al., (2019), que evidencia que se o objetivo é desenvolver as competências dos alunos a nível da programação e pensamento computacional, então é fundamental considerar que o primeiro contacto dos alunos com a programação deve demonstrar uma preocupação com toda a problemática de ensino em si, mas também promover o seu enorme potencial de inter e transdisciplinaridade, especialmente com aplicação prática no âmbito da resolução de problemas.

Porém, é importante acrescentar que o insucesso dos alunos no desenvolvimento das suas competências em programação também se pode dever ao processo de ensino em si, sendo da responsabilidade dos professores promover uma boa metodologia de ensino para os seus alunos. Mais precisamente, alguns estudos referem a inadequação dos métodos pedagógicos para o ensino da disciplina de PSI,

pois não são congruentes com os estilos de aprendizagem dos alunos, não promovendo o seu acompanhamento dos conteúdos lecionados (Gomes et al., 2008).

3.8. Casos de Sucesso no Ensino da Programação

Para (Pi, 2021), existem diversas abordagens que os professores podem usar na sala de aula no ensino da programação, das quais destaco, o Pair Programming e o PRIMM (*Predict, Run, Investigate, Modify* e Make).

O Pair programming é uma técnica em que dois alunos compartilham um computador e trabalham colaborativamente para desenvolver um software. (Hanks et al., 2011) (Hanks, Fitzgerald, McCauley, Murphy, & Zander, 2011). Durante o processo de programação em pares, um aluno denominado *driver* é responsável por digitar o código e controlar o teclado e o rato. O segundo aluno denominado "*navigator*" ou "*observer*" é responsável por desenvolver algoritmos e monitorizar o funcionamento do *driver* para detetar erros e discutir soluções alternativas. O papel dos alunos normalmente muda após um determinado período para evitar a fadiga do papel (Williams & Kessler, 2002).

A abordagem pedagógica PRIMM (*Predict, Run, Investigate, Modify* e Make) foi desenvolvida por Sue Sentance (Sentance & Waite, 2017) e aplicada ao ensino de programação.

Essencialmente, esta abordagem tem por objetivo ajudar todos os professores na estruturação das aulas de programação em 5 etapas distintas, designadamente:

1) ***Predict*:** sendo que os alunos analisam um programa inicial, tentando prever o que irá ser executado. Esta fase centra-se na função do código;

2) ***Run*:** os alunos executam o programa fornecido para testar a previsão efetuada anteriormente, discutindo o resultado com os seus colegas;

3) ***Investigate*:** o professor propõe a realização de várias tarefas e atividades para que os alunos explorem detalhadamente a estrutura do código;

4) ***Modify*****:** os alunos editam o programa para alterar as suas funcionalidades, objetivando-se a implementação de exercícios cada vez mais desafiantes;

5) ***Make*****:** os alunos elaboram um novo programa utilizando uma estrutura igual ou semelhante à do programa utilizado inicialmente, apesar de solucionar um problema distinto (Sentance & Waite, 2017).

É importante destacar que a abordagem PRIMM foi definida com base em diversas investigações na área da programação, partindo de três aspetos-chave. O primeiro está associado à aprendizagem dos alunos, sendo que estes devem aprender primeiro a ler e não a escrever, similarmente ao que ocorre no processo de alfabetização. De acordo com a literatura, aprender a ler em primeiro lugar é extremamente benéfico para programadores numa fase inicial (Xie et al., 2019).

O segundo aspeto-chave está relacionado com a discussão entre alunos sobre os programas desenvolvidos, englobando três níveis específicos:

1) Encontrar a linguagem/terminologia correta para articular o seu entendimento;

2) Verbalizar os seus pensamentos e conhecimento;

3) Participar na criação do entendimento através do diálogo com os seus colegas (Sentance & Waite, 2017).

Por fim, o terceiro aspeto-chave pressupõe que os alunos comecem com um código que não o seu. Ou seja, os alunos devem começar o seu processo de aprendizagem com um programa de iniciação escrito por outra pessoa, garantindo que não existem erros de sintaxe. Este contexto reduz a tensão emocional dos alunos, visto que os programas não têm qualquer tipo de erro, o que acaba por protegê-los da frustração, da ansiedade e da desilusão no que diz respeito à falha dos programas (Sentance & Waite, 2017).

Sentance & Waite, (2017), levaram a cabo um estudo para implementar e avaliar esta abordagem em sala de aula. O estudo é composto por três fases:

- Fase 1 – treino do professor
- Fase 2 – estudo piloto na escola
- Fase 3 – estudo em larga escala na escola

Na Fase 1, o PRIMM foi implementado com 15 professores não especialistas que frequentavam um curso de 8 semanas sobre o ensino da ciência da computação do 6º ao 8º ano. O curso também incluiu métodos desconectados e abordagens de aprendizagem ativa para ensinar ciência da computação, bem como aprender a programar usando PRIMM. Foi elaborado um instrumento de pesquisa para avaliar a perceção dos professores sobre sua aprendizagem e as abordagens pedagógicas utilizadas ao final do curso.

Na fase 2, foram recrutados 7 professores experientes para utilizar a abordagem PRIMM nas suas salas de aula. Este estudo piloto envolveu três etapas: formação dos professores no método PRIMM; intervenção na escola durante um período de 4 a 6 semanas e entrevista. Os professores participaram numa sessão onde se familiarizaram com o método PRIMM e adaptaram os materiais para que fossem adequados às suas próprias aulas. Perguntas de múltipla escolha foram usadas como pré-teste e pós-teste para os alunos e os professores preencheram um diário enquanto usavam as aulas do PRIMM. Os professores foram entrevistados no final do período de 4 a 6 semanas.

Na fase 3, ainda por implementar, será realizado um estudo em mais escolas ao longo de todo um ano letivo. Esta fase está nos estágios iniciais de desenvolvimento.

Os resultados obtidos na fase 1, 15 professores responderam ao questionário de final de curso. O curso modelou uma boa pedagogia durante todas as atividades, além de ensinar conceitos de ciência da computação e programação Python para principiantes. O feedback de todos os aspetos do curso foi altamente positivo. No que diz respeito ao PRIMM, perguntou-se aos professores se consideravam útil aprender a programar utilizando o PRIMM. 47% relataram que foi muito útil, outros 47% relataram que foi útil e apenas 1 entrevistado (7%) relatou que esta abordagem foi ligeiramente útil. No que diz respeito ao seu próprio ensino, 80% dos professores afirmaram que planeavam utilizar o método PRIMM, 13% afirmaram que possivelmente o fariam e apenas 7% (o mesmo inquirido) afirmaram que não o fariam.

Embora seja uma amostra pequena, este feedback é encorajador à medida que avançamos para a Fase 2.

Resumindo, a abordagem PRIMM é uma tentativa não só de propor, mas também de avaliar rigorosamente, uma abordagem ao ensino de programação que possa ser diretamente implementada na sala de aula secundária por professores, experientes ou não (Sentance & Waite, 2017).

Para além destas duas abordagens, *Pair Progamming* e PRIMM, a abordagem Aprendizagem Baseada em Problemas (PrBL), é uma metodologia que difere do modelo tradicional de ensino, pelo fato de utilizar problemas de vida real para iniciar o processo de aprendizagem, visando estimular o desenvolvimento de habilidades para solucionar problemas. Esta metodologia preza pelo uso de problemas baseados no mundo real, que são trabalhados em grupos, no qual o problema é usado para iniciar, direcionar, motivar e focar a aprendizagem, ao contrário dos métodos tradicionais que colocam o problema ao final da apresentação do conteúdo (Stinson & Milter, 1996).

Considerando as potencialidades da abordagem PRIMM apontadas na literatura, considerei a sua utilização para suporte à minha prática pedagógica. Acresce ainda o facto de que a metodologia procura colocar os alunos a resolver problemas de programação, o que já é uma prática seguida pela Professora Cooperante, o que é diferente para os alunos é a forma de abordagem aos problemas.

4. Planificação da Intervenção Pedagógica

4.1. Planificação

A planificação de uma intervenção pedagógica é uma tarefa muito importante e constante na vida de um professor, já que esta consiste na observação de aulas, na criação de um cenário de aprendizagem, na definição das competências a serem desenvolvidas, dos objetivos da aprendizagem a serem atingidos, dos conteúdos a lecionar, das metodologias/estratégias utilizadas, dos recursos necessários e por fim as atividades a serem desenvolvidas pelos alunos, que irei apresentar de seguida.

4.2. Aulas Observadas

A observação desempenha um papel fundamental na melhoria da qualidade do ensino e da aprendizagem, constituindo uma fonte de inspiração e motivação e um forte catalisador de mudança na escola (Reis, 2011).

O estabelecimento de um calendário de observações deverá ser acompanhado da negociação, entre o mentor ou supervisor e o professor, de um conjunto de regras para a sua concretização, nomeadamente no que respeita à frequência, aos participantes, à duração, às finalidades, às dimensões a observar, ao tipo de registo (manuscrito ou gravado em vídeo), ao tipo de observação (participante ou não participante) e ao tipo de feedback (Reis, 2011).

Para registar as minhas observações, utilizei, segundo Reis, (2011), o instrumento de registo de fim aberto, (vide **Figura 17**), que se adequa numa fase inicial/exploratória em que se desconhecem as competências do professor. Este documento permite obter uma ideia do que acontece numa sala de aula através do registo dos principais acontecimentos observados. O registo dos acontecimentos deverá ser efetuado de cinco em cinco minutos, permitindo obter um "retrato" pormenorizado da aula observada.

Figura 17

Grelha de observação de fim aberto

Nome do professor: _____

Data: ___/___/_____ Ano e turma: _____ Disciplina: _____

Tempo	Notas
8h30m	
8h35m	
8h40m	
8h45m	
8h50m	
8h55m	
(...)	

Rubrica do observador: Rubrica do professor:

Antes de iniciar o processo de observação de aulas, apresentei à Professora Cooperante uma tabela com as datas sugeridas em que iriam ocorrer as observações, e a professora concordou com as datas.

No âmbito do meu projeto de intervenção, as observações das aulas foram importantes para ter um conhecimento da turma, e detetar possíveis alunos que tinham ritmos diferentes de aprendizagem. No decorrer da observação constatei que um quarto da turma se mostrava motivado na área da programação, e a restante turma não estava motivada.

Como já referi acima, a turma do 1º PGI tem dois turnos, T1 e T2. Inicialmente eu e Professora Cooperante decidimos que a intervenção iria ocorrer no T1, deste modo, realizei diversas observações neste turno. No entanto, com o decorrer da

observação das aulas no T1, com a planificação da intervenção que apresentei à professora Cooperante, e como o T2 tem as aulas práticas primeiro que o T1, decidimos que seria melhor efetuar a intervenção no T2. Neste sentido realizei, igualmente, diversas observações no T2.

As idades dos alunos da turma variam entre os 14 e os 17 anos, sendo a média de idades de 17 anos. A maioria dos alunos são de nacionalidade portuguesa, à exceção de um aluno ucraniano e outro brasileiro, ambos os alunos não apresentam dificuldades na compreensão e escrita da língua portuguesa. Nenhum dos alunos da turma tem retenção.

Após algumas sessões de observação, faz sentido como refere Reis, (2011), utilizar diversas grelhas de observação que se foquem em diversas áreas especificas (entusiasmo; estratégias de ensino; clareza; organização e gestão; interacção; ambiente de sala de aula). Neste sentido, utilizei o quadro 17 – Grelha de observação focada: estratégias de ensino, (vide **Anexo D**), para registar a organização e gestão do professor.

Para recolher informação sobre os alunos criei uma grelha de observação dos alunos, (vide **Anexo E**).

Durante as minhas observações, registei as seguintes anotações:

- **Organização da sala de aula:** as mesas e cadeiras da sala de aula estão dispostas sob a forma de "U"; os alunos estão sentados a 1,5 metros de distância entre si; os alunos estão agrupados aos pares; a sala possui diversos recursos tais como (Computadores para alunos e computador do professor com acesso à internet, projetor e quadro branco); normalmente não existe barulho na sala, no entanto nas aulas práticas, em alguns momentos os alunos fazem um mais ruído, nesse momento a Professora Cooperante chama a atenção aos alunos quando se distraem e começam a conversar entre si e a fazer muito ruído; Existe luz suficiente na sala de aula; e os alunos têm lugares fixos na sala de aula. Em relação à disposição da sala, em minha opinião, não é a melhor, porque os alunos que estão sentados ao computador no topo sala têm de fazer uma rotação de 180º para conseguirem ver o

quadro/tela, onde a Professora Cooperante escreve ou projeta. Segundo meu ponto de vista, seria melhor que os alunos estivessem sentados de forma a terem de fazer uma rotação de 90º para conseguir ver o quadro.

- **Gestão da sala de aula:** A Professora Cooperante é que define as regras da sala de aula e o que se vai fazer na aula; não há flexibilidade no plano de aula, ou seja, os alunos têm de seguir o planeamento da Professora Cooperante; nenhum aluno possui o Kit digital, apesar de já o terem solicitado à escola; não há computadores para todos os alunos poderem trabalhar de forma individual; nas aulas práticas iniciais do módulo 1, os alunos com ritmo aprendizagem mais rápido trabalhavam de forma individual, e os restantes em grupos de dois alunos; antes de terminar as aulas práticas do módulo 1, a Professora Cooperante agrupou todos os alunos em grupos de dois (aluno com um ritmo de aprendizagem mais rápido com outro aluno com ritmo de aprendizagem mais lento).
- **Interação na sala de aula:** Normalmente é a Professora Cooperante que fala quando dá a matéria usando o método expositivo; nas aulas teóricas e práticas a Professora questiona com frequência os alunos sobre o que fazem determinados blocos de código projetados no quadro; este método promove interação entre a professora e os alunos.
- **Discurso do professor:** A Professora Cooperante discursa com voz bem colocada; nas aulas práticas entoa um pouco mais a voz, para realçar e chamar a atenção aos alunos para detalhes na resolução; felicita/motiva com frequência aos alunos sempre que estes vão progredindo na resolução dos exercícios práticos.
- **Discurso dos alunos:** Os alunos com ritmo aprendizagem mais rápido normalmente têm um discurso mais confiante e participam na aula com maior frequência; os alunos com ritmo aprendizagem mais lento, participam menos na aula.

- **Relação entre os alunos:** Nem todos os alunos interagem uns com os outros, observei que os alunos que estão mais motivados para a aprendizagem da programação, tendem a interagir entre eles, e o mesmo acontece com grupo alunos que estão menos motivados na aprendizagem da programação; os alunos movimentam-se na sala de aula aquando a realização de exercícios práticos, normalmente quando têm alguma dúvida questionam o colega se sabe, ou se já executou determinada tarefa, existem outros alunos que quando têm alguma dúvida, colocam a mão no ar e chamam a Professora Cooperante;
- **Clima de sala de aula:** na sala de aula a Professora Cooperante está interessada e entusiasmada em expor a matéria; a professora sabe os nomes dos alunos e de vez em quando usa o humor na sala de aula, e não inferioriza nem envergonha nenhum aluno; a Professora Cooperante estimula a participação dos alunos e incentiva-os na execução das tarefas/exercícios.
- **Atividades educativas:** As atividades propostas pela Professora Cooperante, são adequadas aos objetivos propostos, e são maioritariamente atividades práticas, tais como o cálculo de área e perímetro de formas geométricas (quadrado, triângulo); cálculo da média aritmética e ponderada; conversões de unidades de medida (horas, minutos, segundos); comparações entre dois valores; apresentar a tabuada de um número inteiro. Ou seja, as atividades propostas têm por objetivo consolidar os conceitos lecionados. A Professora Cooperante informa os alunos qual o tema, a duração, critérios de avaliação das atividades;

4.2.1. Cenário de Aprendizagem

O conceito de cenário de aprendizagem tem sido utilizado em diferentes áreas (e.g. economia, marketing, medicina, desenvolvimento de software, game design, e em muitas outras) como estratégia para pensar sobre o futuro, antecipando problemas

e prevendo soluções para esses mesmos problemas. Neste contexto, os cenários podem ser encarados como ferramentas extremamente úteis para estimular novas estratégias criativas de pensar sobre problemas que levem as pessoas a sair da sua zona de conforto ou de formas pré-estabelecidas de pensar (Piedade et al., 2018).

A idealização de cenários de aprendizagem é algo que o professor faz regularmente, de forma mais ou menos sistematizada, sempre que planifica as suas atividades de ensino e procura antecipar ou desenhar diferentes experiências e problemas que pretende fazer acontecer na sua sala de aula no trabalho com os seus alunos (Piedade et al., 2018).

O cenário de aprendizagem que apresento, (vide **Anexo F**), enquadra-se na disciplina de PSI do curso TGPSI, do ensino profissional, no início do módulo 4. Foi idealizado em conjunto com a Professora Cooperante, pelo que o presento de seguida.

Pretende-se criar o programa Sequência Mágica, na linguagem de programação C. Este programa consiste em gerar duas sequências de números inteiros. A primeira sequência é composta por cinco números inteiros entre 1 e 50, a segunda sequência e composta por dois números inteiros entre 1 e 9, gerados aleatoriamente pelo programa. Ambas as sequencias são armazenadas em dois vetores, respetivamente, pela ordem em que são gerados. Depois gerados os números e os vetores preenchidos, o programa deverá apresentá-los ordenados de forma crescente. Durante os 14 tempos de 45 minutos, os alunos deverão conseguir criar o programa Sequência Mágica.

Este cenário de aprendizagem descreve um desafio que consiste na criação de um programa na linguagem programação C, em que os alunos irão colocar em prática os conhecimentos adquiridos nos módulos 1, 2, 3 e 4 mais concretamente: recordar noção e algoritmo, variáveis, estruturas de decisão, repetição, compreender definição de Array, efetuar inserções e ordenações de elementos num Array.

Este cenário de aprendizagem será realizado em três fases. Numa primeira fase o professor expôs os conceitos teóricos, seguidos de exemplos práticos. Numa segunda fase o professor criou vários exercícios em que a sua resolução resultou em diversos blocos de código que contribuíram para a criação do desafio da Sequência Mágica. Na

terceira e última fase, os alunos terão de juntar blocos de código de forma a obterem o desafio concluído. O **Anexo G**, apresenta com mais detalhes o cenário acima referido.

4.2.2. Objetivos da Aprendizagem

Os objetivos da aprendizagem estão enunciados no cenário de aprendizagem, no entanto como objetivo geral, os alunos deverão conseguir criar um programa em C, como objetivos específicos, os alunos no final dos 14 tempos deverão conseguir:

- Criar e iniciar os vetores do tipo Inteiro;
- Gerar um número aleatório entre 1 e 50;
- Pesquisar um número num vetor/*array;*
- Preencher vetor/*array;*
- Ordenar vetor de números inteiro por ordem crescente.

4.2.3. Metodologias e Estratégias

A disciplina de PSI tem um carácter predominantemente prático e experimental. Torna-se, por isso, necessário implementar metodologias através de actividades que incidam sobre a aplicação prática e contextualizada dos conteúdos, a experimentação, a pesquisa e a resolução de problemas. Neste sentido, as aulas deverão privilegiar a participação dos alunos em projectos e na resolução de problemas e de exercícios que simulem a realidade. O professor deverá adoptar estratégias que motivem o aluno a envolver-se na sua própria aprendizagem e lhe permitam desenvolver a sua autonomia e iniciativa (Direcção-Geral de Formação Vocacional, 2005).

A Professora Cooperante adota a metodologia de aprendizagem baseada em problemas (PrBL), que vai de encontro ao que é sugerido no programa da disciplina de PSI, nomeadamente, com a realização de muitos exercícios práticos que consolidam os conceitos teóricos. Disponibiliza aos alunos várias fichas de trabalho por cada módulo, com diversos problemas para resolver. Na plataforma Teams da ESACF, são

colocados os recursos dos módulos da disciplina de PSI, bem como, todas as fichas de trabalho que os alunos necessitam.

Contudo, durante a observação das aulas, deparei-me que os alunos relevam dificuldades em assimilar alguns conceitos de programação, por exemplo, na estrutura de repetição *for* em C, os alunos na declaração do ciclo confundiam a parte do incremento do ciclo, com o que se pretendia realizar dentro de cada ciclo.

Como refere, Santos, (2022), para me ajudar a identificar a metodologia de ensino/aprendizagem mais adequada para a intervenção na disciplina de PSI, iniciei o planeamento da pesquisa bibliográfica, identificando as fontes para a pesquisa (motores de busca na internet, Bases de Dados online, revistas de investigação, livros, dissertações e teses).

A PRIMM é uma tentativa de não apenas propor, mas avaliar rigorosamente uma abordagem de ensino da programação que pode ser diretamente implementada nas turmas do ensino secundário quer por professores experientes ou não (Sentance & Waite, 2017). Esta abordagem pode ajudar os professores a estruturar aulas de programação. Como já referi acima, PRIMM significa *Predict*, *Run*, *Investigate*, *Modify* e *Make*, representando diferentes etapas de uma aula, ou de uma série de aulas. PRIMM promove debate entre os alunos sobre como programas funcionam e o uso de programas iniciais para incentivar a leitura do código antes de escrever.

O debate em sala de aula é um aspeto importante no ensino de muitos assuntos, mas não é frequentemente mencionado em relação ao ensino de programação. Muitas atividades do PRIMM são realizadas em pares, e já sabemos que a programação em pares é uma forma eficaz de aprendizagem e envolve os alunos praticando para articular o que fazer quando escrevem um programa. O debate traz os seguintes benefícios:

- Falar sobre um programa e como ele funciona ajuda os alunos encontrar a terminologia correta a ser usada para articular os seus entendimentos. Ter uma linguagem comum para falar na fase de escrita de um programa é importante;

- Na verdade, verbalizar em voz alta as etapas de um programa que são difíceis de entender podem ajudar os alunos a se concentrarem em elementos menores ou atómicos de cada vez;
- Através do debate com outras pessoas, podemos perguntar e responder a questões e aprender com os outros;

Escrever código num editor em branco, pode deixar os alunos principiantes pressionados. A sintaxe precisa estar correta, ou bastante mensagens de erro podem intimidar. Neste sentido, a primeira etapa da abordagem PRIMM, *Predict*, envolve a leitura e previsão do que um pequeno programa faz, libertando o aluno do stress de criar um programa de raiz. A etapa, *Run*, o envolve a execução de um programa fornecido para verificar se a previsão estava correta ou não. A etapa, *Investigate*, surge no sentido de alunos investigarem o programa quando a previsão não coincide com a execução do programa. Gradualmente, na etapa *Modify*, à medida que os alunos vão tendo alguma compreensão de como o código funciona, eles podem modificar o código e apropriar-se da nova funcionalidade. Por último, etapa *Make*, os alunos criam seu próprio programa tendo por base o programa modificado, nesta etapa alunos já têm sentimento de pertença sobre o programa (Sentance & Waite, 2017).

Tendo por base estes aspetos da abordagem PRIMM acima referidos, após a leitura da dissertação do colega Miguel Santos e tendo em conta as dificuldades sentidas pelos alunos, irei utilizar a abordagem pedagógica PRIMM, para implementar o cenário de aprendizagem acima referido.

4.2.4. Papel do Professor

Numa primeira fase, apresentei os conceitos teóricos relativos ao módulo 4, Estruturas de Dados Estáticas, e realização de alguns exercícios práticos de consolidação.

Na segunda fase, apliquei a abordagem PRIMM, onde pretendi apresentar aos alunos um pequeno programa em C a funcionar, e a partir deste os alunos realizaram

diversas atividades inseridas dentro das etapas da abordagem PRIMM. Desta forma, os alunos interagiram entre si no desenvolvimento de programas, onde assumi um papel de orientador.

Na terceira fase, apresentei o desafio da Sequência Mágica à turma e acompanhei os alunos na realização do desafio, assumindo igual forma um papel de orientador.

4.2.5. Papel dos Alunos (Competências a Desenvolver)

Com a criação das duas sessões PRIMM e do cenário de aprendizagem, Sequência Mágica, pretendi que os alunos tivessem um papel ativo no processo de aprendizagem, realizando brainstorming nos exercícios acima propostos e interagindo trabalhando em equipa.

Basicamente, os alunos tiveram de realizar diversos exercícios em cada sessão PRRIM que foram: ler e analisar um programa na linguagem C facultado pelo Professor Estagiário e prever qual o resultado da sua execução; executar o programa num editor da linguagem C; após a execução do programa, e caso o output não estivesse de acordo com o previsto, alunos deveriam investigar o porquê; modificar o programa alterando a sua funcionalidade; desenvolver um novo programa tendo por base o programa inicialmente facultado, mas resolvendo um problema diferente.

Por último, os alunos realizaram o cenário de aprendizagem, Sequência Mágica, que essencialmente, consistia em reaproveitar os exercícios realizados nas duas sessões PRIMM para resolver um problema diferente.

Com o desenvolvimento das duas sessões PRIMM e o cenário de aprendizagem , pretendi que os alunos adquirissem as seguintes competências:

- Linguagens e textos;
- Compreender as estruturas de dados estáticas na Linguagem C;
- Raciocínio e Resolução de Problemas;
- Pensamento Critico e Pensamento Computacional;

- Relacionamento Interpessoal;
- Desenvolvimento Pessoal e Autonomia;
- Saber científico, técnico e tecnológico,

4.2.6. Ferramentas e Recursos

Para desenvolver com os alunos as duas sessões PRIMM e o cenário de aprendizagem que idealizei, foram necessárias as seguintes ferramentas e recursos:

- Computadores: professor e todos os alunos;
- Acesso à internet;
- DevC;
- videoprojector;
- Tela;
- Grelhas de Observação suportadas pelo Microsoft Excel;
- RED suportado pelo Microsoft Forms para as sessões PRIMM.

4.2.7. Pessoas e Lugares

Ambas as sessões PRIMM e realização do cenário do cenário de aprendizagm, decorreram na sala de aula, com o acompanhamento da Professora Cooperante e do Professor Estagiário que idealizou este projeto de intervenção.

4.3. Calendarização da Intervenção

A planificação de intervenção foi delineada para ter uma ficha de avaliação diagnóstica, seis aulas e uma ficha de avaliação final. De acordo, com o cronograma anual da Professora Cooperante, (vide **Anexo H**), a intervenção pedagógica estava planeada iniciar a 19 de fevereiro de 2024 e terminar a 29 de fevereiro, constituída por seis aulas que cuja duração variava entre 90 e 135 minutos, totalizando 630 minutos. No entanto, não foi possível realizar a intervenção na data prevista, porque a Professora Cooperante terminou mais tarde do que previsto o módulo 3, bem como foi

necessário mais uma aula. Desta forma, a intervenção acabou por ser constituída por sete aulas cuja duração variou entre 90 e 135 minutos, que tiveram início a 4 de março e terminaram a 18 de março, o que perfez um total de 690 minutos.

4.3.1. Aula #0 - 29/02/2024 (45 minutos)

Esta aula, que designei de aula zero, foi lecionada pela Professora Cooperante, no entanto utilizei os primeiros 35 minutos para que os alunos dos dois turnos (T1 e T2) pudessem responder ao teste de avaliação diagnóstica (vide **Anexo P**), disponibilizado na plataforma Microsoft Forms no seguinte link: (https://forms.office.com/e/FfzwWNcTJY).

4.3.2. Aula #1 - 04/03/2024 (90 minutos)

Na primeira aula, a maioria dos alunos foram assíduos e pontuais, exceto cinco alunos chegaram trinta minutos atrasados e apenas um aluno faltou. Nesta aula estiveram presentes os dois turnos (T1 e T2), por se tratar de uma teórica semanal.

Comecei por apresentar, com recurso a uma apresentação PowerPoint (vide **Anexo Q**), o propósito da minha intervenção à turma, ou seja, de forma resumida disse o que iríamos fazer nas próximas seis aulas os temas que iríamos abordar, o desafio que iríamos desenvolver e como seria a avaliação.

Seguidamente, fiz uma breve revisão dos conceitos lecionados nos módulos dois e três, dos quais destaco algumas instruções, palavras reservadas em C e os tipos de dados simples, com objetivo de rever as dificuldades no teste diagnóstico e contextualizar a introdução aos tipos de dados estruturados.

Continuei com a exposição teórica, onde apresentei os quatro tipos de dados estruturados (vetor, registo, conjunto e ficheiro). Expliquei que a minha intervenção só iria incidir sobre o tipo de dados estruturado vetor.

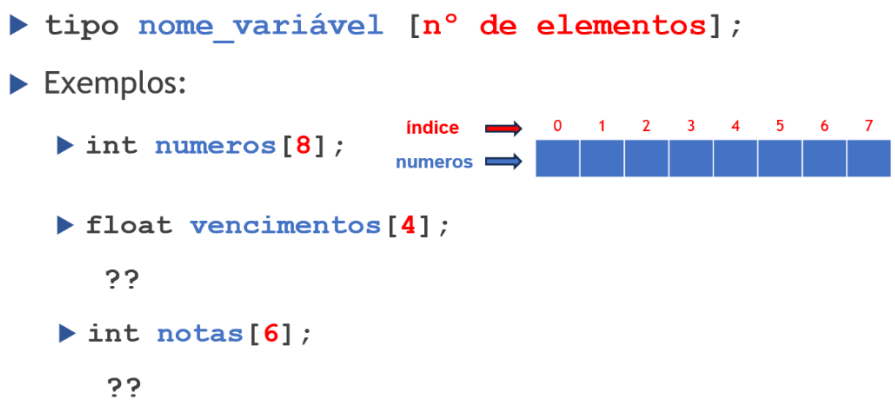
De seguida comecei por apresentar o que é, como se declara e a forma como acedemos a um vetor, bem como a diferença entre índice e valor indexado. Apesar de

apresentação em PowerPoint ter sido disponibilizada na plataforma Teams, solicitei aos alunos que transcrevessem para o seu caderno os conceitos científicos associados aos vetores, permitindo que alguns alunos fizessem anotações nos seus cadernos, promovendo dessa forma uma melhor memorização e a compreensão dos conceitos sobre vetores.

Após os alunos transcreverem os conceitos, questionei se têm dúvidas, ao qual respondem que não. Apesar de os alunos responderem que não tinham dúvidas, como complemento da exposição teórica e para promover interação entre mim e os alunos, coloquei questões à turma sobre a declaração de vetores, pedi-lhes que respondessem nos seus cadernos a declaração dos seguintes vetores: *float vencimentos[4]* e *int notas[6]* apresentados na **Figura 18**, para que percebam a diferença entre o número de elementos e o índice de um vetor.

Figura 18

Exemplo de declaração de um vetor



Na segunda parte da aula, prossegui com a exposição teórica sobre como, se acede a elemento do vetor, se atribui um valor a um elemento do vetor, é feita leitura de um elemento do vetor e a inicialização automática do vetor. Solicitei aos alunos para transcreverem os conceitos e questiono se têm dúvidas. À semelhança da primeira parte da aula, interagi com os alunos para perceber se eles entenderam os conceitos de

índice e valor indexado. Nesse sentido pedi-lhes que respondam nos seus cadernos qual seria o seria o output das linhas de código apresentadas na **Figura 19**.

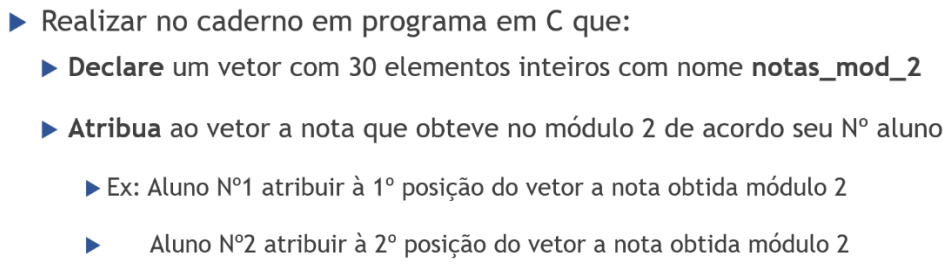
Figura 19

Distinção entre índice e valor indexado

Com o aproximar do fim da aula e como última tarefa, solicitei aos alunos que realizassem no seu caderno um exercício prático que aborda os todos conceitos lecionados na aula, ver **Figura 20**.

Figura 20

Exercício prático que resume os conceitos científicos abordados na aula 1.



Após disponibilizar algum tempo aos alunos para realizarem o exercício, fiz a resolução no quadro, um exemplo de declaração do vetor com a notas do aluno número um e número trinta. O objetivo deste exercício era aferir se os alunos perceberam a distinção entre índice e valor indexado.

O que tinha planeado abordar na primeira aula foi integralmente conseguido (vide **Anexo I**).

4.3.3. Aula #2 - 05/03/2024 (90 minutos)

Na segunda aula, a maioria dos alunos foram assíduos e pontuais, exceto três alunos chegaram cerca de dois minutos depois da tolerância. Nesta aula esteve presente apenas o T2.

Comecei a aula, com recurso a uma apresentação PowerPoint (vide **Anexo R**), com o resumo da aula anterior, onde questionei os alunos sobre os conceitos fundamentais abordados. Todas as questões aos alunos foram respondidas.

De seguida apresentei o sumário e prossegui com os objetivos delineados para a segunda aula (vide **anexo J**). Na primeira parte da aula, foi apresentada a abordagem pedagógica PRIMM, que iria ser utilizada na realização dos exercícios práticos sobre os conceitos teóricos lecionados na aula #1.

Prossegui com a apresentação e explanação aos alunos das cinco fases da abordagem pedagógica PRIMM. De seguida expliquei aos alunos que iremos realizar a primeira sessão prática da abordagem PRIMM, e que todos passos necessários à realização do exercício estão disponibilizados na plataforma TEAMS.

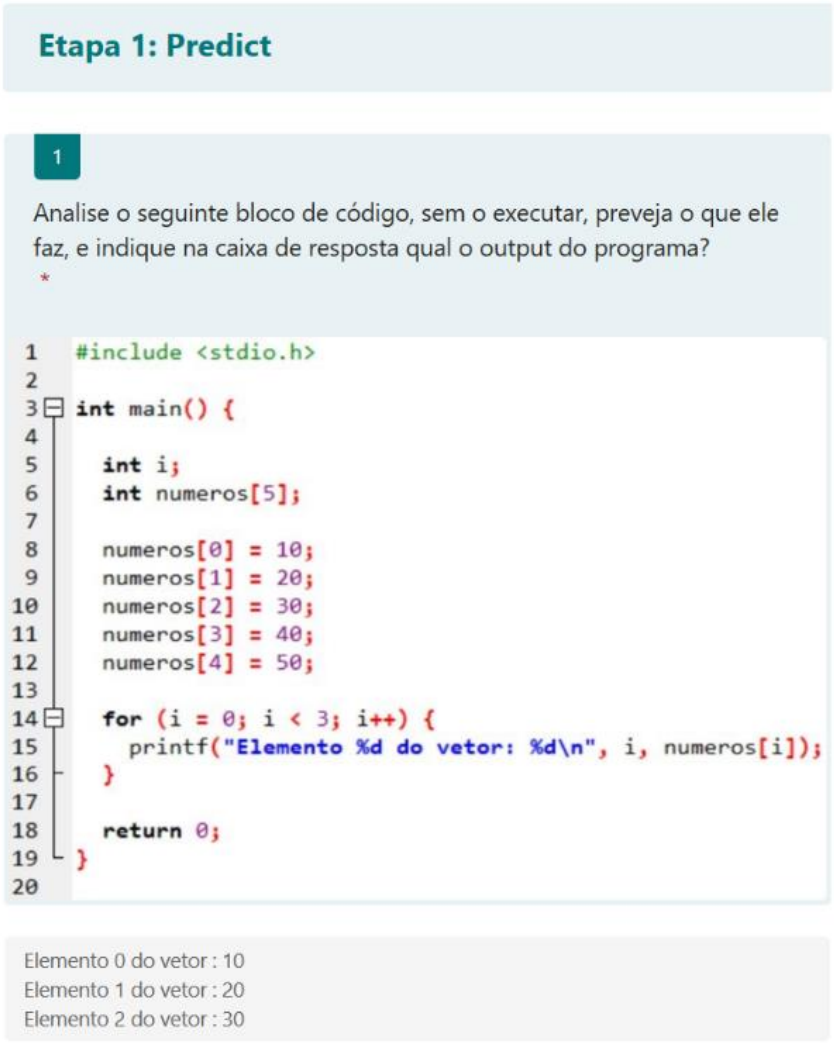
Sendo esta a primeira vez que os alunos tomaram contato com a abordagem pedagógica PRIMM, disponibilizei um RED, formulário Microsoft Forms, para a primeira sessão, no seguinte link: https://forms.office.com/e/PMB2CqPVNr, que deu suporte às cinco fases da abordagem pedagógica. Dada a novidade e entusiasmo por parte dos alunos, todos eles realizaram a primeira sessão prática PRIMM individualmente.

Na primeira fase da abordagem pedagógica, ***Predict***, os alunos leram um completo programa em C, onde tiveram de prever o qual seria o output do programa e registar a sua resposta no RED.

A **Figura 21** apresenta um exemplo de resposta correta de um aluno

Figura 21

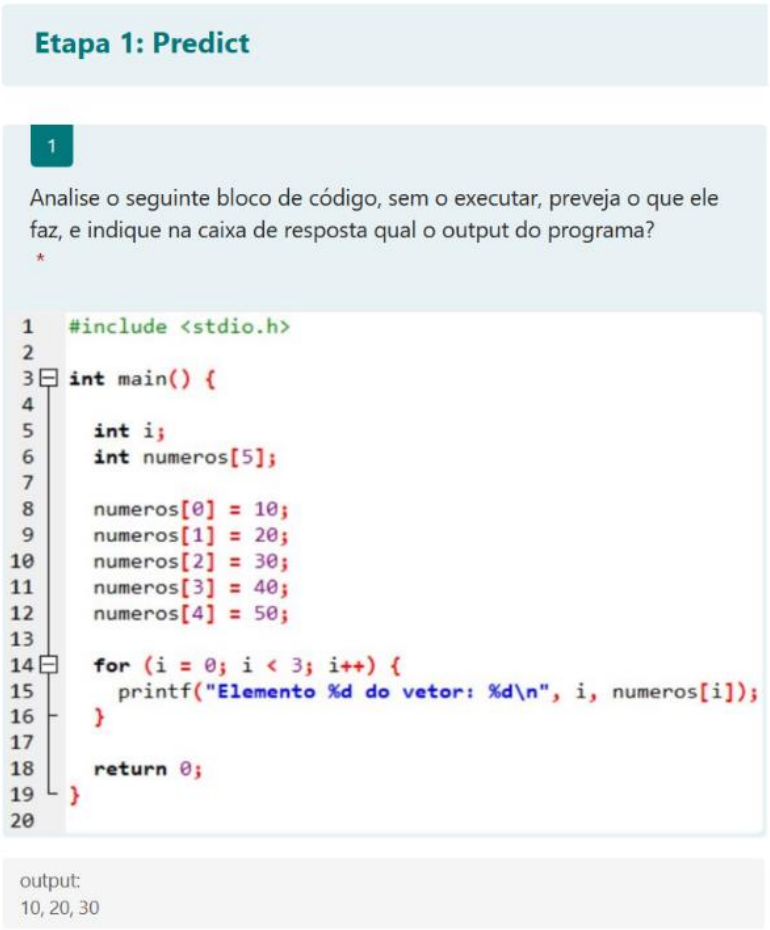
*Exemplo de resposta de aluno nº 16 da fase **Predict** abordagem PRIMM*



A **Figura *22***, apresenta um exemplo de resposta incorreta de outro aluno.

Figura 22

*Exemplo de resposta de aluno nº 29 da fase **Predict** abordagem PRIMM*



Etapa 1: Predict

1

Analise o seguinte bloco de código, sem o executar, preveja o que ele faz, e indique na caixa de resposta qual o output do programa?
*

```c
#include <stdio.h>

int main() {

  int i;
  int numeros[5];

  numeros[0] = 10;
  numeros[1] = 20;
  numeros[2] = 30;
  numeros[3] = 40;
  numeros[4] = 50;

  for (i = 0; i < 3; i++) {
    printf("Elemento %d do vetor: %d\n", i, numeros[i]);
  }

  return 0;
}
```

output:
10, 20, 30

Na segunda fase da abordagem pedagógica, ***Run***, os alunos executaram o programa da etapa anterior num editor de C, e verificaram se o output do programa é o mesmo que previram na etapa ***Predict***. A **Figura 23**, mostra um exemplo de resposta de um aluno.

Figura 23

*Exemplo de resposta de aluno nº 16 da fase **Run** abordagem PRIMM*

Etapa 2: Run

Copie o seguinte bloco de código para ambiente de desenvolvimento **DevC** ou **Online GDB**, compile e execute o programa.

```
#include <stdio.h>

int main() {

  int i;
  int numeros[5];

  numeros[0] = 10;
  numeros[1] = 20;
  numeros[2] = 30;
  numeros[3] = 40;
  numeros[4] = 50;

  for (i = 0; i < 3; i++) {
    printf("Elemento %d do vetor: %d\n", i, numeros[i]);
  }

  return 0;
}
```

2

O output do programa foi o mesmo que previu na etapa 1? *

Sim

Não

3

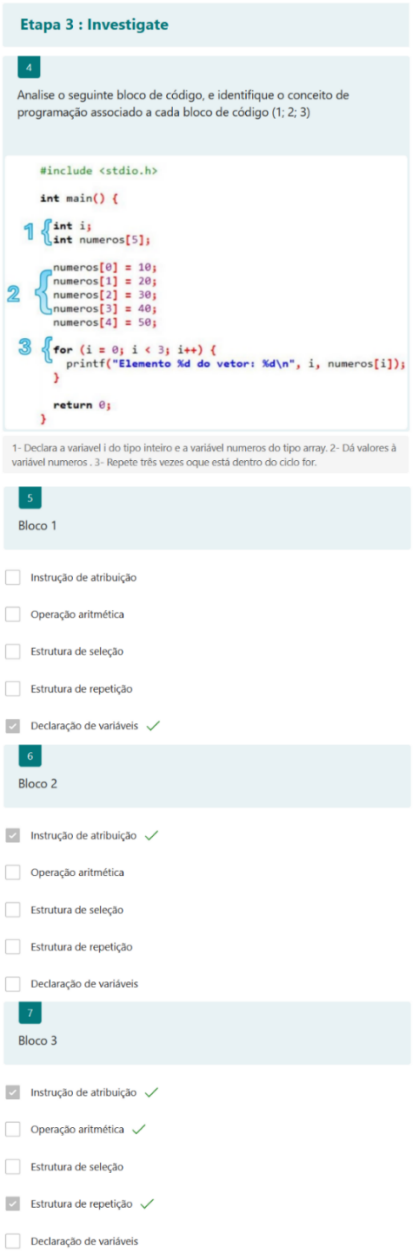
Coloque/Cole aqui o output do programa da etapa Run. *

Elemento 0 do vetor: 10 Elemento 1 do vetor: 20 Elemento 2 do vetor: 30

Após os alunos verificarem o output do programa, quer seja igual ao que previram, quer seja diferente, solicitei aos alunos para realizar a fase ***Investigate***, como forma de treino para se habituarem a ler blocos de código e tentar perceber qual o significado cada linha de código. A **Figura 24**, mostra o exemplo da resposta de um aluno.

Figura 24

Exemplo de resposta de aluno nº 16 da fase ***Investigate*** *abordagem PRIMM*



De seguida os alunos prosseguiram para a próxima fase, ***Modify***, onde lhes foi solicitado para alterar uma parte do programa para listar todas as posições do vetor. A **Figura 25**, apresenta um exemplo de resposta correta de um aluno.

Figura 25

*Exemplo de resposta de aluno nº 16 da fase **Modify** abordagem PRIMM*

Etapa 4 : Modify

8

Altere o seguinte programa, para que apresente no ecrã, todos os elementos do vetor:
Responda apenas a linha de código que irá sofrer alteração.

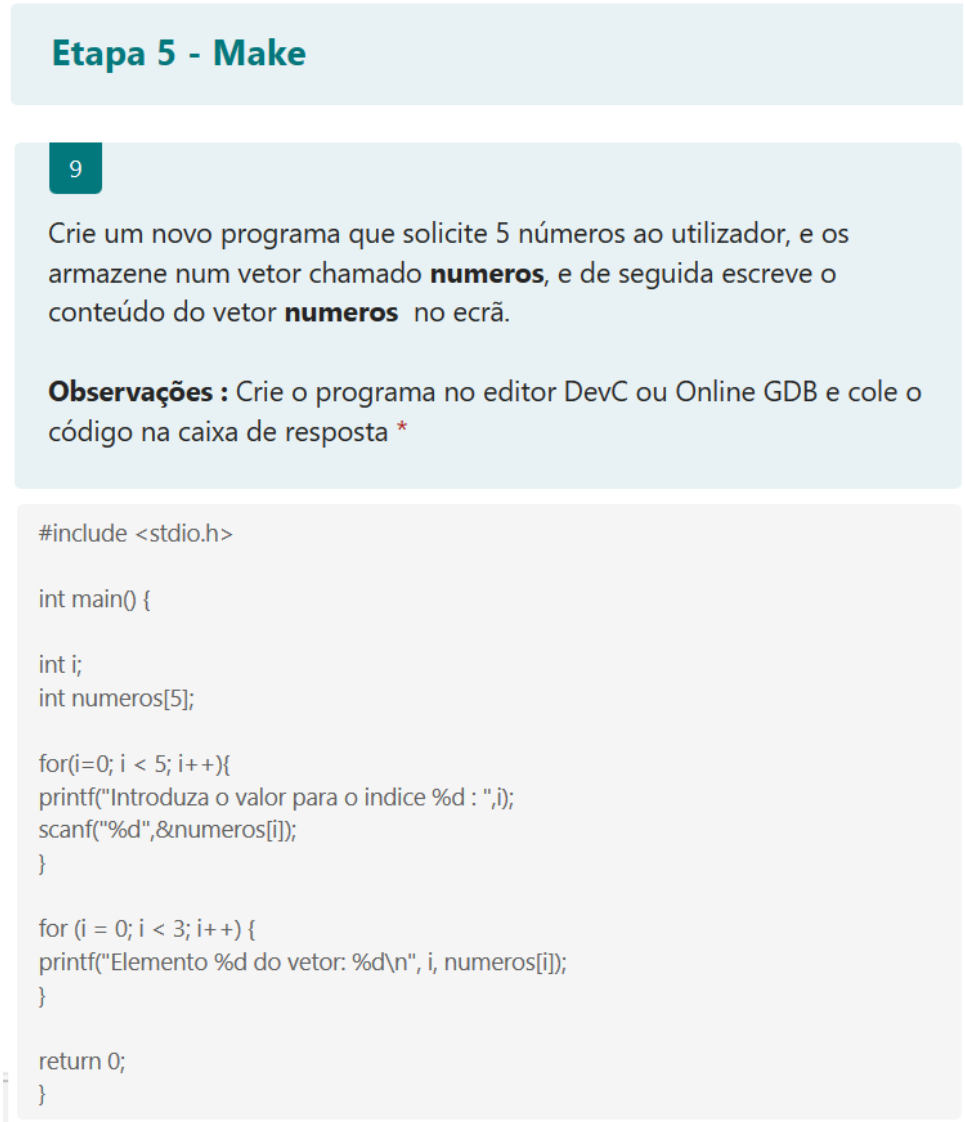
```
#include <stdio.h>

int main() {

  int i;
  int numeros[5];

  numeros[0] = 10;
  numeros[1] = 20;
  numeros[2] = 30;
  numeros[3] = 40;
  numeros[4] = 50;

  for (i = 0; i < 3; i++) {
    printf("Elemento %d do vetor: %d\n", i, numeros[i]);
  }

  return 0;
} *
```

```
for(i=0; i<5; i++)
```

Por fim, na última fase, ***Make***, tendo por base o programa disponibilizado na fase ***Predict,*** foi solicitado aos alunos que criem um novo programa que solicite 5 números ao utilizador, os armazene num vetor chamado numeros, e de seguida apresente no ecrã o conteúdo do vetor (vide **Figura 26**).

Figura 26

*Exemplo de resposta de aluno nº 16 da fase **Make** abordagem **PRIMM***



Etapa 5 - Make

9

Crie um novo programa que solicite 5 números ao utilizador, e os armazene num vetor chamado **numeros**, e de seguida escreve o conteúdo do vetor **numeros** no ecrã.

Observações : Crie o programa no editor DevC ou Online GDB e cole o código na caixa de resposta *

```
#include <stdio.h>

int main() {

int i;
int numeros[5];

for(i=0; i < 5; i++){
printf("Introduza o valor para o indice %d : ",i);
scanf("%d",&numeros[i]);
}

for (i = 0; i < 3; i++) {
printf("Elemento %d do vetor: %d\n", i, numeros[i]);
}

return 0;
}
```

A maioria dos alunos realizaram a primeira sessão PRIMM dentro do tempo expectável, à exceção de três alunos. Durante a realização desta sessão, foi notória a existência de alunos com diferentes ritmos na realização da sessão. Estes ritmos traduzem-se em dificuldades que alguns alunos sentiam, nomeadamente, nos ciclos *for* para percorrer os vetores. na declaração nos vetores e na afetação de valores aos elementos do vetor.

Na segunda parte da aula, prossegui, como planeado, com os conceitos teóricos sobre pesquisa de um valor no vetor, calcular a soma/média/maior/menor de um vetor com elementos numéricos. Os alunos de uma forma geral perceberam os conceitos à exceção de um aluno que me solicitou para explicar novamente como é feita a pesquisa de um valor no vetor.

Para consolidar estes conceitos teóricos, preparei outro RED, formulário Microsoft Forms no seguinte link: https://forms.office.com/e/T6Wt4Q4mfZ, para realizar a segunda sessão PRIMM. Disponibilizei na plataforma TEAMS todos passos necessários à realização desta segunda sessão.

A aula termina, sem que todos os alunos concluíssem a segunda sessão PRIMM.

Após a aula terminar cheguei a conclusão que, na segunda parte da aula não deveria ter avançado para conceitos teóricos sobre percorrer um vetor e os algoritmos para calcular a soma/média/maior/menor de um vetor, mas sim, deveria ter feito a correção da primeira sessão ***PRIMM*** e só na terceira aula é que deveria ter lecionado os conceitos teóricos que lecionei na segunda parte da aula #2.

4.3.4. Aula #3 - 07/03/2024 (135 minutos)

Na terceira aula, a maioria dos alunos foram assíduos e pontuais, exceto um aluno que faltou. Nesta aula esteve presente apenas o T2.

Esta aula foi a continuação da aula #2, e teve como objetivo a finalização da segunda sessão PRIMM e realização da correção da primeira e segunda sessão PRIMM.

Durante a realização da segunda sessão PRIMM, interagi diversas vezes com alunos, e apercebi-me que eles se sentiam mais à vontade nas fases ***Predict***, ***Run*** e ***Modify***, relativamente às fases ***Investigate*** e ***Make***, alguns alunos revelaram algumas dificuldades. Estas dificuldades estavam diretamente relacionadas com o nível de complexidade dos exercícios na fase ***Make,*** mais concretamente em determinar a quantidade de números pares do vetor, o menor número do vetor.

De seguida apresento um exemplo de resposta de aluno com algumas dificuldades na realização da fase ***Investigate***, em que não respondeu de acordo com o que tinha solicitado (vide **Figura 27**).

Figura 27

Exemplo de resposta de aluno nº 6 da fase ***Investigate*** *2ª Sessão PRIMM*

A **Figura 28**, apresenta um exemplo de resposta correta de um aluno que foi de encontro exatamente ao que tinha solicitado.

Figura 28

*Exemplo de resposta de aluno nº 23 da fase **Investigate** 2ª Sessão PRIMM*

Etapa 3 : Investigate

4

Analise o seguinte bloco de código, indique na caixa de resposta o que faz o bloco de código entre a linha 4 e 10.

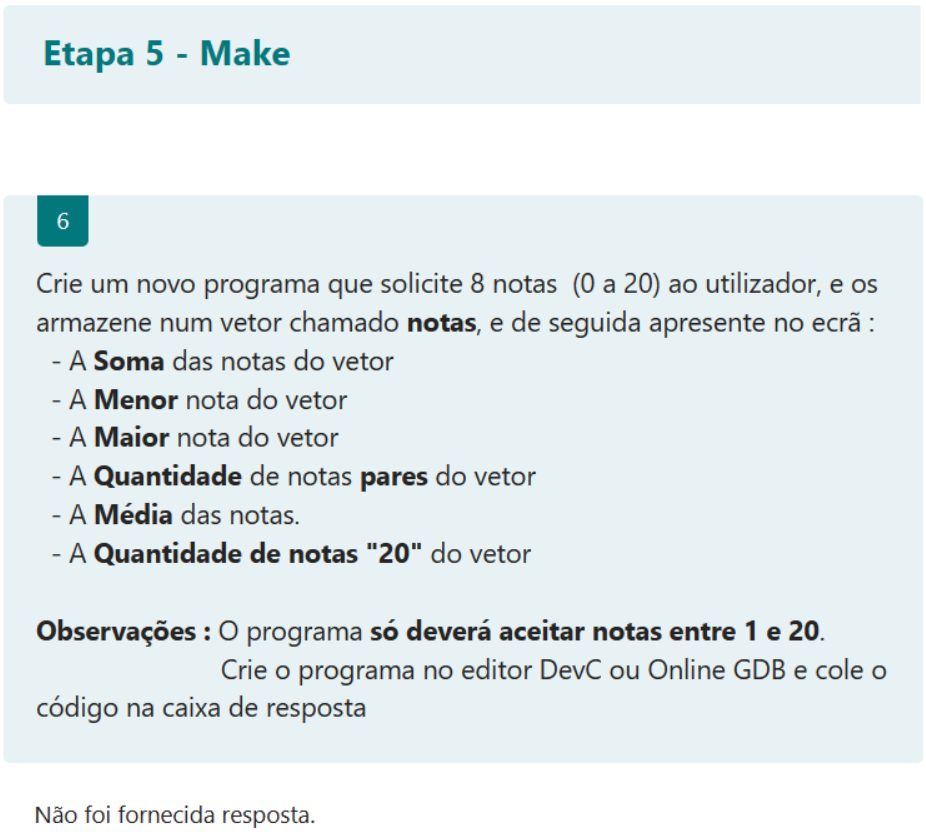
```
1  #include <stdio.h>
2  main()
3  {
4     int numeros[5];
5     int i;
6     for (i=0;i<=4;i++)
7      {
8       printf("Introduza um numero inteiro para a posicao %d\n",i);
9       scanf("%d",&numeros[i]);
10     }
11    printf("\n\n");
12
13    for (i=0;i<=4;i++)
14       printf("%d\n",numeros[i]);
15 }
```

Linha 4 declaração do array numeros com 5 elementos;
linha 5 é a declaração da variavel i;
linha 6 é o ciclo for para controlar os índices do array de 0 a 4;
linha 7 inicio do bloco de código do ciclo for;
linha 8 pede ao utilizador para introduzir um número 5 vezes para preencher o array;
linha 9 leitura do valor anterior para o array;
linha 10 do bloco do ciclo for;

Em relação à fase ***Make***, apresento um exemplo de um aluno, que sentiu dificuldades, não respondeu questão e não chamou o Professor Estagiário para esclarecer qualquer dúvida realização do exercício (vide **Figura 29**).

Figura 29

*Exemplo de ausência de resposta de aluno nº 29 da fase **Make** 2ª Sessão PRIMM*



Ainda na fase ***Make***, apresento um exemplo de resposta de um aluno que sentiu algumas dificuldades na resolução do exercício, o professor estagiário orientou o aluno, conseguindo resolver uma parte do exercício (vide **Figura 30**).

Figura 30

*Exemplo de resposta de aluno nº 18 da fase **Make** 2ª Sessão PRIMM*

Etapa 5 - Make

6

Crie um novo programa que solicite 8 notas (0 a 20) ao utilizador, e os armazene num vetor chamado **notas**, e de seguida apresente no ecrã :
- A **Soma** das notas do vetor
- A **Menor** nota do vetor
- A **Maior** nota do vetor
- A **Quantidade** de notas **pares** do vetor
- A **Média** das notas.
- A **Quantidade de notas "20"** do vetor

Observações : O programa **só deverá aceitar notas entre 1 e 20**.
Crie o programa no editor DevC ou Online GDB e cole o código na caixa de resposta

```
#include <stdio.h>
main()
{ //Somar todas as notas
int notas[8];
int i;
int soma = 0;
int menor = 21;
int maior = 0;

for (i=0;i<=7;i++)
{
printf("Introduza um numero inteiro para a posicao %d\n",i);
scanf("%d",&notas[i]);
}
printf("\n\n");
for (i=0;i<=7;i++){
soma = soma + notas[i];

if(notas[i]> maior)
maior = notas[i];

if(notas[i]< menor)
menor = notas[i];
}
printf("O total das notas do vetor é %d:\n",soma);
printf("A nota maior do vetor é %d:\n", maior);
printf("A nota menor do vetor é %d:\n", menor);

}
```

Após todos os alunos terminarem a segunda sessão PRIMM, procedi à correção da primeira e segunda sessão PRIMM. Os **Anexos BB** e **Anexos CC** apresentam uma proposta de solução para a primeira e segunda sessão PRIMM respetivamente.

Durante a correção, interagi diversas vezes com os alunos, e apercebi-me que os alunos se sentiram muito à vontade nas fases ***Predict***, ***Run*** e ***Modify***, ao contrário das fases ***Investigate*** e ***Make***, em que alguns alunos revelaram algumas dificuldades.

4.3.5. Aula #4 - 11/03/2024 (90 minutos)

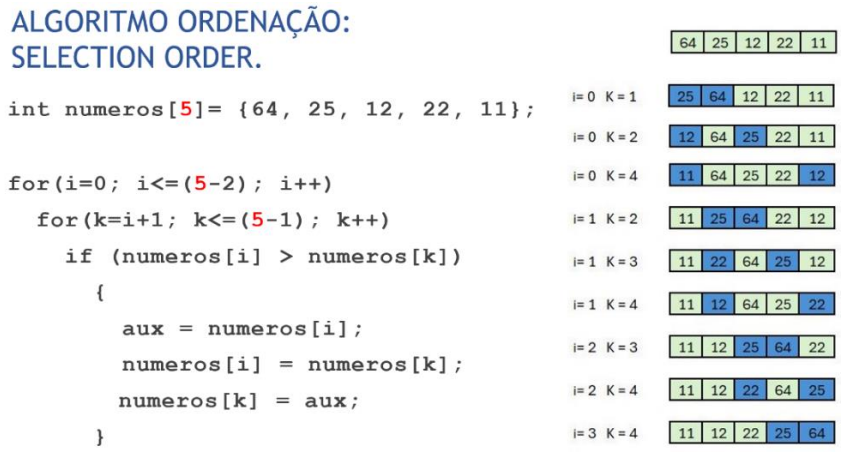
Na quarta aula, a maioria dos alunos foram assíduos e pontuais, exceto dois alunos que chegaram atrasados e outros dois que faltaram. Nesta aula estiveram presentes os dois turnos (T1 e T2).

Comecei a aula, com recurso a uma apresentação PowerPoint (vide **Anexo S**), com o resumo da aula anterior, onde questionei os alunos se tinham dúvidas sobre os conceitos fundamentais abordados, não houve nenhum aluno que se manifestasse.

De seguida apresentei o sumário e prossegui com os objetivos delineados para a quarta aula (vide **anexo K**). Na primeira parte da aula, foi apresentado o algoritmo de ordenação de vetores, *selection order*. Enquanto explicava passo a passo o algoritmo de ordenação, a maioria dos alunos estavam atentos, no entanto, quando terminei a explicação, questionei se havia dúvidas, e uma grande maioria respondeu que sim. Volto a explicar o algoritmo passo a passo, e solicito aos alunos que copiem para o caderno o algoritmo (vide **Figura 31**).

Figura 31

Algoritmo Selection Order



Questionando novamente os alunos se tinham dúvidas, ao qual responderam que não, no entanto, reconhecendo que o algoritmo não seja fácil, expliquei novamente o algoritmo.

Na segunda parte da aula, apresentei aos alunos o Desafio da Sequência Mágica, onde, expliquei aos alunos que a realização deste desafio consiste em reutilizar ou modificar alguns blocos de código que eles tinham desenvolvido na primeira e segunda sessão PRIMM.

Criei uma tarefa na plataforma Teams, onde anexei o guião para a realização do desafio da Sequência Mágica (vide **Anexo U**), em que apresento o objetivo, tempo estimado resolução, recursos e as etapas do desafio. Apresentei, igualmente, aos alunos a grelha com a Rúbrica de avaliação do desafio.

4.3.6. Aula #5 - 12/03/2024 (90 minutos)

Na quinta aula, a maioria dos alunos foram assíduos e pontuais, exceto um aluno que chegou atrasado e outro que faltou. Nesta aula esteve presente apenas o T2.

Comecei a aula, com recurso a uma apresentação PowerPoint (vide **Anexo T**), onde comecei por apresentar o sumário da aula, recordei novamente aos alunos o guião que estes tiveream que seguir para a realização do desafio Sequência Mágica.

Apesar de as duas sessões PRIMM terem sido realizadas individualmente pelos alunos, o guião do desafio Sequência Mágica promove a criação de grupos para a sua realização. No entanto, houve alguns alunos que me questionaram se era possível realizar o desafio individualmente, ao qual respondi que sim. Desta forma quatro alunos realizaram-no individualmente, os restantes alunos organizaram-se em grupos, mais concretamente, quatro grupo de dois alunos e um grupo de três alunos.

Constatei que, alguns alunos numa fase inicial começaram por ter dificuldades em iniciar a realização do desafio da Sequência Mágica. Nesse sentido, projetei no quadro o guião do desafio, onde apresenta um bloco de código que gera um número aleatório (vide **Figura 32**), com o objetivo de os alunos terem uma base de trabalho.

Figura 32

Exemplo em como gerar número aleatório

```
#include <time.h>
#include <stdlib.h>
…
int numero;
int main(){
      srand(time(NULL));
      numero = rand();
} …
```

Recordei aos alunos que a realização de desafio da Sequência Mágica, essencialmente, consiste na junção e modificação de blocos de código que foram realizados nos exercícios práticos das duas sessões PRIMM. A partir daqui os alunos começaram a realizar o desafio.

Enquanto alunos realizavam o desafio, eu deslocava-me pela sala para ir monitorizando o avanço dos alunos. Logo numa fase inicial, foi notório, que alguns alunos/grupos tinham um ritmo de trabalho mais rápido que outros. Os alunos/grupos que apresentavam um ritmo mais lento, colocavam-me mais dúvidas durante a realização do desafio, no qual eu prontamente as esclareci-a.

Apercebi-me que a maioria dos alunos estava com dificuldades na aplicação do operador matemático módulo, para delimitar um número aleatório entre um e cinquenta. Desta forma, pedi aos alunos que parassem momentaneamente, e

prestassem todos atenção, onde expliquei no quadro para toda a turma a aplicação do operador módulo.

A turma prosseguiu com a realização do desafio, e com o aproximar do fim da aula, dois alunos já tinham terminado o desafio, e começaram a ajudar outros grupos que estavam com mais dificuldades na realização, desempenhando um papel de auxílio ao Professor Estagiário.

O **Anexo V**, apresenta um exemplo de resolução do desafio da Sequência Mágica realizado pelo aluno nº 16.

A aula termina sem que todos terminassem o desafio. Constatei que foram os alunos que sentiram mais dificuldades na primeira e segunda sessão PRIMM, que não conseguiram resolver o desafio no tempo estipulado. Informei os alunos que na próxima aula, o primeiro tempo de quarenta e cinco da seria para terminar o desafio.

O que tinha planeado abordar na quinta aula (vide **Anexo K**), foi integralmente conseguido.

4.3.7. Aula #6 - 14/03/2024 (135 minutos)

Na sexta aula, a maioria dos alunos foram assíduos e pontuais, exceto cinco alunos que chegaram atrasados e outro que faltou. Nesta aula esteve presente apenas o T2.

Comecei por apresentar, com recurso a uma apresentação PowerPoint (vide **Anexo X**), o sumário da aula, onde referi que o primeiro tempo de quarenta e cinco minutos seria para os alunos/grupos que não tivessem terminado o desafio, o terminassem. O segundo tempo seria para cada aluno/grupo apresentar o desafio à turma, e o por fim, último tempo, seria para os alunos realizarem a sua autoavaliação, a avaliação da intervenção e ficha de avaliação final.

Com o decorrer da aula, termina o primeiro tempo, e alguns alunos/grupos ainda não tinham terminado o desafio da Sequência Mágica, e solicitaram mais um tempo de quarenta e cinco minuto para o finalizarem. Concedi mais um tempo, e todos

alunos/grupos conseguiram terminar. Houve dois grupos que ainda tinham dúvidas no algoritmo de ordenação, *selection order*, pelo que, voltei a explicar-lhes o algoritmo.

Os alunos/grupos que tinham terminado o desafio dentro do tempo estipulado, ajudaram outros grupos que ainda estavam a terminar, auxiliando o Professor Estagiário.

Por questões de limitação de tempo, cada aluno/grupo apresentou o desafio da Sequência Mágica à turma no computador onde desenvolveu o desafio, em vez de o projetar no quadro com recurso ao computador do professor. Após a apresentação, projetei uma solução possível para o desafio (vide **Anexo W**).

Por fim, solicitei aos alunos para acederem ao Teams onde disponibilizei dois links, um com um questionário de avaliação da intervenção e outro com o questionário de autoavaliação.

Pelo facto de alunos solicitarem mais um tempo de quarenta e cinco minutos para terminarem o desafio, não foi possível cumprir na íntegra o que tinha planeado abordar na sexta aula (vide **Anexo L**). Desta forma, informei os alunos que na próxima aula, iriam realizar no primeiro tempo de quarenta e cinco minutos, a ficha de avaliação final.

4.3.8. Aula #7 - 18/03/2024 (90 minutos)

Na sétima aula, a maioria dos alunos foram assíduos e pontuais, exceto um aluno que chegou atrasado e dois que faltaram. Nesta aula esteve presente o T1 e o T2.

Comecei por apresentar, com recurso a uma apresentação PowerPoint (vide **Anexo X**), o sumário da aula, onde referi que o primeiro tempo de quarenta e cinco minutos seria para os alunos realizarem individualmente a ficha de avaliação final.

Após a apresentação, solicitei aos alunos que para aceder ao Teams onde disponibilizei o link, https://forms.office.com/e/CM2XK22T1h, para o questionário de avaliação final (vide **anexo AA**).

5. Avaliação da Intervenção Pedagógica

Segundo o programa da disciplina, sugere que seja feita uma avaliação diagnóstica no início do ano letivo, para identificar grupos diferenciados e estabelecer um plano de ação para cada grupo de alunos, com objetivo de aquisição das competências essenciais definidas no programa.

Deverá ser privilegiada a observação direta do trabalho desenvolvido pelo aluno durante as aulas, utilizando para isso instrumentos de avaliação diversificados que permitam registar o desempenho do aluno nas situações que lhe são proporcionadas e a progressão na aprendizagem ao longo do ano letivo, nomeadamente quanto ao interesse, à participação no trabalho, à capacidade de trabalhar em grupo, à qualidade do trabalho realizado e à forma como o aluno gere e organiza a autoavaliação.

A par da avaliação contínua, permitindo o registo da evolução do aluno aula a aula e a recuperação, em tempo útil, de qualquer dificuldade, deverão ser previstos momentos de avaliação, procedendo-se à aplicação de provas de carácter prático ou teórico-prático que permitam avaliar os conhecimentos e competências adquiridos (Direcção-Geral de Formação Vocacional, 2005).

Neste sentido, para a avaliação da minha intervenção, idealizei quatro tipos de avaliação: a diagnóstica, a formativa, a sumativa e avaliação da minha intervenção (vide **Figura 33**).

Figura 33

Resumo dos tipos de avaliação usadas na intervenção

Instrumentos de Recolha de dados	Tipo Avaliação	Ferramenta
Ficha Avaliação Diagnóstica (pré-teste)	Diagnóstica (Q1)	Critérios Gerais de Classificação
Grelha Observação Direta	Formativa (Q3)	Critérios Gerais de Classificação
Grelha Observação Sessões PRIMM	Formativa (Q3)	Critérios Gerais de Classificação
Grelha Avaliação do Desafio	Formativa (Q3)	Rúbrica
Ficha Avaliação (pós-teste)	Sumativa (Q1)	Critérios Gerais de Classificação
Questionario Avaliação da Intervenção	Avaliação da Intervenção (Q2)	Google Forms

5.1. Operacionalização da Avaliação Diagnóstica

A avaliação diagnóstica tem como objetivo aferir os conhecimentos dos alunos sobre uma dada temática, de forma que o professor tenha uma visão sobre os conhecimentos, competências desenvolvidas pela turma, para planificar uma temática a ser ensinada à turma.

Nas diversas observações efetuadas, a intenção foi tomar contacto com a turma, conhecê-la, observar os alunos individualmente e observar a prática letiva da Professora Cooperante.

As observações foram essenciais para eu planificar a minha intervenção. As notas retiradas nas diversas observações representam um instrumento de recolha de dados e de certa forma uma avaliação diagnóstica.

Para a além das observações, recorri à Taxonomia de *Bloom* para a construção de uma ficha de avaliação diagnóstica, da qual passo de seguida a explicar.

5.1.1. Taxonomia de *Bloom* aplicada à Avaliação

Como instrumento de apoio didático-pedagógico, a Taxonomia de *Bloom* tem como objetivo geral contribuir com todos aqueles que direta ou indiretamente se ocupam com problemas referentes a currículo e avaliação (Trevisan & Amaral, 2016).

A taxonomia original de *Bloom* define seis principais categorias do domínio cognitivo: conhecimento, compreensão, aplicação, análise, síntese e avaliação. As categorias são ordenadas da mais simples para a mais complexa e, possuem uma hierarquia cumulativa, sendo a categoria mais simples pré-requisito para a próxima (Trevisan & Amaral, 2016).

A **Figura 34**, apresenta a Taxonomia de *Bloom* original.

Figura 34

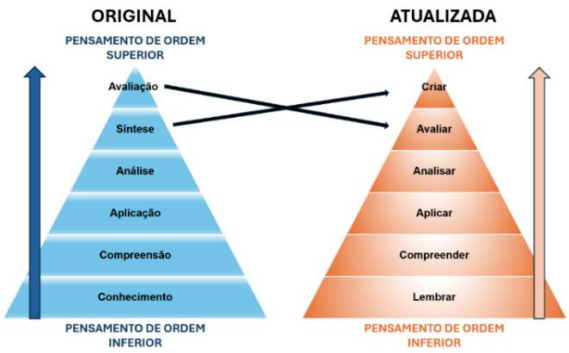
Taxonomia Bloom original

Em 1990, a taxionomia passou por um processo de revisão e, em 2001, foi publicada por Lorin Anderson e seus colaboradores. Na Taxionomia de *Bloom* atualizada foram combinados o tipo de conhecimento a ser adquirido e o processo utilizado para a aquisição desse conhecimento (Anderson & Krathwohl, 2001).

O tipo de conhecimento passou a ser designado por substantivos e os processos para atingi-los passaram a ser descritos por verbos. O nível do conhecimento, compreensão e síntese foram renomeados para lembrar, compreender e criar, respetivamente, conforme mostra a **Figura 35**.

Figura 35

Taxonomia Bloom original versus Atualizada



5.1.2. Avaliação Diagnóstica

Como a minha intervenção foi no início do módulo 4, realizei uma ficha de avaliação diagnóstica (vide **Anexo P**), uma semana antes da minha intervenção.

Como já referi acima, a ficha de avaliação diagnóstica foi criada de acordo com os níveis taxonomia de *Bloom*. Para uma melhor compreensão, a **Tabela 1,** apresenta para cada questão da ficha de avaliação diagnóstica, os níveis da taxonomia de *Bloom*, o tipo de questão, o conteúdo e a dificuldade.

Tabela 1

Ficha de avaliação diagnóstica de acordo níveis da Taxonomia de Bloom Original

Nível de Taxonomia Bloom	Tipo de questão	Conteúdo	Dificuldade	Nº Questão
Conhecimento	Verdadeiro ou falso / Escolha múltipla	Definição de algoritmo	Fácil	1
Compreensão	Escolha múltipla	Variáveis locais e globais	Fácil	2
Aplicação	Escolha múltipla	Ciclos	Médio	3
Análise	Escolha múltipla	Definição de variáveis	Médio	4
Síntese	Escolha múltipla	Estruturas de decisão	Difícil	5
Avaliação	Desenvolvimento de código	Funções / Programa	Difícil	6 e 7

A ficha de avaliação diagnóstica (pré-teste), foi aplicada aos alunos enquanto se encontravam a frequentar o módulo 3. Estiveram presentes 26 alunos, dos quais, mais de metade (73%) teve nota positiva, e apenas 7 alunos (27%) tiveram nota inferior a 9,5 valores. De salientar que, dos alunos que tinham o módulo 2 em atraso, 5 alunos, pouco mais de metade (55%), tiveram nota inferior a 9,5 valores e os restantes 4 alunos tiveram nota positiva.

A **Tabela 2**, apresenta resumo das notas do módulo 1, módulo 2 e notas da ficha de avaliação diagnóstica.

Tabela 2

Situação da turma após realização da ficha de avaliação diagnóstica

Nº	Grupo (Turno)	Notas M1 (0-20)	Notas M2 (0-20)	Pré-teste (0-20)
1	Controlo (T1)	14	15	16
2	Controlo (T1)	18	17	20
28	Controlo (T1)	10	Em atraso	4
3	Controlo (T1)	11	Em atraso	14,5
4	Controlo (T1)	14	11	11
5	Controlo (T1)	10	Em atraso	15,5
6	Experimental (T2)	12	10	Faltou
7	Controlo (T1)	12	10	5,5
8	Controlo (T1)	13	Em atraso	14,5
9	Controlo (T1)	11	10	3
10	Controlo (T1)	15	Em atraso	1,5
11	Controlo (T1)	10	Em atraso	2
12	Controlo (T1)	15	12	10
13	Controlo (T1)	18	13	7
14	Controlo (T1)	16	15	16
15	Experimental (T2)	13	11	13,5
16	Experimental (T2)	20	19	18
17	Experimental (T2)	12	14	Faltou
18	Experimental (T2)	11	12	Faltou
19	Experimental (T2)	12	12	13,5
20	Experimental (T2)	17	15	14,5
21	Experimental (T2)	11	12	Faltou
22	Experimental (T2)	15	13	12
23	Experimental (T2)	10	Em atraso	14
24	Experimental (T2)	12	Em atraso	7,5
25	Experimental (T2)	13	13	13,5
26	Experimental (T2)	15	15	16,5
27	Experimental (T2)	12	Em atraso	9,5
29	Experimental (T2)	16	12	12
30	Controlo (T1)	18	17	14

5.2. Avaliação Formativa

A avaliação formativa desempenha um papel fundamental no processo de ensino e de aprendizagem; informa sobre o estado dos alunos, os problemas que apresentam, como estão a progredir. Um feedback relevante e imediato promove a autonomia dos alunos e a autorregulação dos seus próprios processos de aprendizagem, e permite ao professor decidir sobre o processo de ensino, promovendo a adaptação de estratégias de acordo com as necessidades dos alunos (Dorotea & Pedro, 2015).

Tendo a disciplina de PSI uma componente prática muito presente, nas aulas práticas são realizados diversos exercícios práticos com o objetivo de consolidar os conceitos teóricos, e de recolha de informação da evolução das aprendizagens por parte dos alunos, de forma à Professora Cooperante ir dando feedback aos alunos sobre a sua evolução.

A avaliação formativa, enquanto principal modalidade de avaliação, integra o processo de ensino e de aprendizagem fundamentando o seu desenvolvimento (Portaria n.o 235-A, 2018).

Neste sentido, no inico de cada aula fazia sempre uma revisão sobre os conceitos lecionados na aula anterior e dava feedback aos alunos. O meu é objetivo era ativar o conhecimento prévio, identificação de dificuldades, estimular a motivação e atenção dos alunos de forma a melhorar o seu desempenho. Outra forma de dar feedback aos alunos, era durante a realização dos exercícios, pedia-lhes que refletissem sobre a sua resolução, e que identificassem melhorias a aplicar.

No decorrer da minha intervenção pedagógica, avaliei de forma formativa os alunos com três instrumentos de recolha de informação; grelha de observação direta, grelha registo das sessões PRIMM e grelha de registo do procedimento prático.

A grelha de observação direta é constituída por catorze indicadores, e cada indicador foi preenchido de acordo com critérios gerais de classificação (vide **Anexo E**). Em cada aula da minha intervenção pedagógica, registei para cada aluno os indicadores da grelha de observação direta, com os níveis de um a cinco, sendo um o nível mais baixo e cinco o nível mais alto. Com base na análise da grelha de observação

direta, concluo que, de uma forma geral, os alunos apresentam um bom desempenho em relação aos objetivos de aprendizagem propostos.

Em relação à grelha de registo das sessões PRIMM, é constituída pelas cinco etapas da abordagem PRIMM, e cada etapa foi preenchida de acordo com critérios gerais de classificação (vide **Anexo O**). Para eu conseguir perceber quais das etapas de cada sessão PRIMM em que os alunos tiveram mais dificuldades, criei uma escala com níveis de um a cinco, e avaliei cada etapa com um nível dessa escala (vide **Figura 36**). Os níveis da escala de avaliação estão descritos da seguinte forma:

1. Aluno não respondeu.
2. Aluno tentou resolver, mas não atingiu os resultados pretendidos.
3. Aluno atingiu resultado mínimos.
4. Aluno ultrapassou os resultados mínimos, mas não atingiu o excelente.
5. Aluno atingiu excelente, e até apresenta outras formas de solução.

Figura 36

Grelhas de registo da 1ª e 2ª sessão PRIMM

Data : 05/03/2024	Registo 1ª Sessão PRIMM														
Etapas Sessões PRIMM \ Alunos	A6	A15	A16	A17	A18	A19	A20	A21	A22	A24	A25	A26	A27	A29	Total por Etapa PRIMM
1ª Etapa - Predict	2	5	5	5	3	5	3	5	1	1	5	3	5	2	**50**
2ª Etapa - Run	1	5	5	5	3	5	5	5	5	1	5	5	5	5	**60**
3ª Etapa - Investigate	4	2	4	4	4	4	2	4	2	3	3	3	4	3	**46**
4ª Etapa - Modify	5	5	5	5	5	5	5	3	5	5	5	5	5	5	**68**
5ª Etapa - Make	3	4	4	4,5	5	3	1	4	3	4	4	5	4	5	**53,5**
Total Por Aluno	**15**	**21**	**23**	**24**	**20**	**22**	**16**	**21**	**16**	**14**	**22**	**21**	**23**	**20**	

Data : 07/03/2024	Registo 2ª Sessão PRIMM														
Etapas Sessões PRIMM \ Alunos	A6	A15	A16	A17	A18	A19	A20	A21	A22	A24	A25	A26	A27	A29	Total por Etapa PRIMM
1ª Etapa - Predict	3	5	5	5	4	5	3	5	5	5	4	4	3	5	**61**
2ª Etapa - Run	5	5	5	5	5	5	5	5	5	5	5	4	5	5	**69**
3ª Etapa - Investigate	1	4	5	5	4	3	2	4	4	5	1	4	2	4	**48**
4ª Etapa - Modify	3	2	5	5	5	3	5	3	4	5	3	5	5	5	**58**
5ª Etapa - Make	1	5	4	5	2	1	1	4	5	4	1	4	3	1	**41**
Total Por Aluno	**13**	**21**	**24**	**25**	**20**	**17**	**16**	**21**	**23**	**24**	**14**	**21**	**18**	**20**	

Com base na análise das grelhas de sessão PRIMM, concluo que de uma forma geral, as etapas ***Predict*** e ***Run***, foram as que os alunos obtiveram melhores resultados, no entanto realço que na primeira sessão PRIMM, houve 4 alunos que obtiveram um

mau resultado na realização da etapa ***Predict***. Já as etapas ***Investigate*** e ***Make***, foram as que os alunos apresentaram piores resultados.

Por último, a grelha de registo do procedimento prático é constituída por quatro critérios (vide **Anexo M**), e cada critério foi preenchido de acordo com uma rúbrica existente no AE4O (vide **Anexo N**). Durante a realização e posterior correção do desafio da Sequência Mágica, registei para cada aluno, os quatro critérios com os níveis de um a cinco, sendo um o nível mais baixo e cinco o nível mais alto (vide **Figura 37**).

Figura 37

Grelhas de registo da realização do desafio Sequência Mágica

Data : 12/03/2024	Procedimento Prático														
Critérios \ Alunos	Grupo I		Grupo II	Grupo III		Grupo IV	Grupo V		Grupo VI		Grupo VII		Grupo VIII	Grupo IX	
	A21	A18	A17	A23	A15	A16	A6	A25	A19	A20	A27	A22	A24	A26	A29
Qualidade Trabalho (com base nos objetivos do Procedimento)	5	5	5	5	5	5	4	4	4	4	3	3	4	4	4
Organização do trabalho/ Criatividade	3	3	3	3	3	5	3	3	3	3	3	4	3	4	4
Gestão do tempo/ Cumprimento de prazos	3	3	4	3	3	5	4	4	3	3	4	4	4	4	4
Empenho	4	4	5	5	5	5	3	3	3	3	5	4	3	4	4

Com base na análise da grelha registo do desafio Sequência Mágica, concluo que de uma forma geral, os alunos obtiveram bons resultados e realizaram com sucesso este desafio.

5.3. Avaliação Sumativa

Na minha intervenção, eu e Professora Cooperante, acordámos em utilizar a ficha de avaliação final como instrumentos de recolha para avaliar a competência de conhecimentos e capacidades conceptuais e factuais, e utilizar a Observação direta como instrumento de recolha para avaliar a competência dos Conhecimentos processuais e comunicacionais.

A ficha de avaliação final (vide **Anexo AA**), foi criada de acordo com os níveis taxonomia de *Bloom*. Para uma melhor compreensão, a **Tabela 3** apresenta para cada questão da ficha de avaliação final, os níveis da taxonomia de *Bloom*, o tipo de questão, o conteúdo e a dificuldade.

Tabela 3

Ficha de avaliação final de acordo níveis da Taxonomia de Bloom Original

Nível de Taxonomia *Bloom*	**Tipo de questão**	**Conteúdo**	**Dificuldade**	**Nº Questão**
Conhecimento	Escolha múltipla	Definição de vetor	Fácil	1
Compreensão	Escolha múltipla	Declaração de um vetor	Fácil	2
Compreensão	Escolha múltipla	Vetores / Variáveis	Fácil	3, 4 e 5
Aplicação	Resposta aberta	Atribuição valor a elemento vetor	Médio	6
Aplicação	Escolha múltipla	Atribuição valor a elemento vetor	Médio	7
Análise	Escolha múltipla	ciclos / Atribuição valor a elemento vetor	Médio	8
Análise	Resposta aberta	Atribuição valor a elemento vetor	Médio	9
Síntese	Desenvolvimento de código	Funções	Difícil	10
Avaliação	Desenvolvimento de código	Programa	Difícil	11

A ficha de avaliação final (pós-teste), foi aplicada aos alunos na última aula da minha intervenção. Estiveram presentes 29 alunos, dos quais, uma grande maioria (89%) teve nota positiva, e apenas 3 alunos (11%) tiveram nota inferior a 9,5 valores. De salientar que, dos alunos que tinham o módulo 2 em atraso, apenas 1 aluno (11%), teve nota inferior a 9,5 valores e os restantes tiveram nota positiva.

A **Tabela 4**, apresenta resumo das notas do módulo 1, módulo 2 e notas da ficha de avaliação diagnóstica e ficha de avaliação final.

Tabela 4

Situação da turma após realização da ficha de avaliação sumativa

Nº	Grupo (Turno)	Notas M1 (0-20)	Notas M2 (0-20)	Pré-teste (0-20)	Pós-teste (0-20)
1	Controlo (T1)	14	15	16	13
2	Controlo (T1)	18	17	20	18,5
28	Controlo (T1)	10	Em atraso	4	16,5
3	Controlo (T1)	11	Em atraso	14,5	16
4	Controlo (T1)	14	11	11	15,5
5	Controlo (T1)	10	Em atraso	15,5	4
6	Experimental (T2)	12	10	Faltou	10
7	Controlo (T1)	12	10	5,5	16
8	Controlo (T1)	13	Em atraso	14,5	12
9	Controlo (T1)	11	10	3	10
10	Controlo (T1)	15	Em atraso	1,5	7,5
11	Controlo (T1)	10	Em atraso	2	11
12	Controlo (T1)	15	12	10	14
13	Controlo (T1)	18	13	7	Faltou
14	Controlo (T1)	16	15	16	14
15	Experimental (T2)	13	11	13,5	15,5
16	Experimental (T2)	20	19	18	19
17	Experimental (T2)	12	14	Faltou	14
18	Experimental (T2)	11	12	Faltou	13
19	Experimental (T2)	12	12	13,5	11
20	Experimental (T2)	17	15	14,5	Faltou
21	Experimental (T2)	11	12	Faltou	8,5
22	Experimental (T2)	15	13	12	13
23	Experimental (T2)	10	Em atraso	14	10
24	Experimental (T2)	12	Em atraso	7,5	11
25	Experimental (T2)	13	13	13,5	11,5
26	Experimental (T2)	15	15	16,5	14
27	Experimental (T2)	12	Em atraso	9,5	17
29	Experimental (T2)	16	12	12	15
30	Controlo (T1)	18	17	14	17,5

5.4. Autoavaliação do Alunos

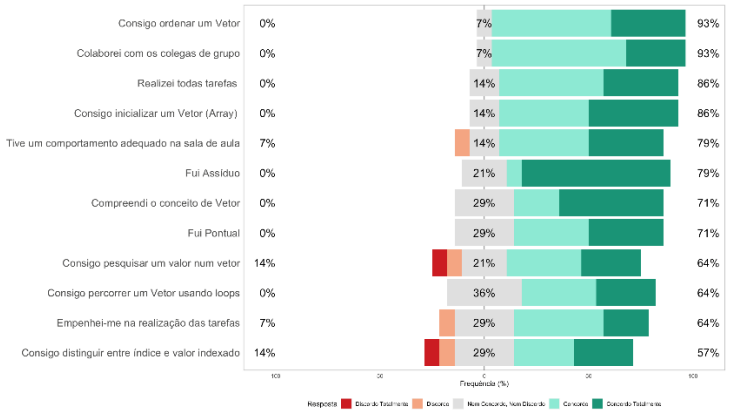
Na aula onde foi pedido aos alunos o preenchimento do questionário sobre a avaliação da intervenção, foi solicitado aos alunos que preenchessem, igualmente, o questionário de autoavaliação (vide **Anexo Z**). Este questionário foi criado com uma escala de *likert*, com cinco níveis de resposta:

1. Discordo totalmente.
2. Discordo parcialmente.
3. Nem concordo e nem discordo.
4. Concordo.
5. Concordo totalmente.

Os resultados apresentados na **Figura 38**, são referentes à autoavaliação dos alunos, e demonstram que as respostas dos alunos foram genericamente positivas. Em relação aos itens com maior concordância, foram "*Fui assíduo*" e "*Compreendi o conceito de vetor*". Já os itens com maior discordância, foram "Consigo distinguir entre índice e valor indexado" e "Consigo pesquisar um valor num vetor".

Figura 38

Frequências relativas das respostas dos participantes sobre a Autoavaliação dos alunos (n=14)



5.5. Avaliação da Intervenção Pedagógica

Na última aula da minha intervenção, disponibilizei um questionário aos alunos como instrumento de avaliação da minha intervenção (vide **Anexo Y**). A ferramenta para o efeito foi o Microsoft Forms. O link para o questionário é https://forms.office.com/e/FSwbGJBWdQ.

O questionário de avaliação da intervenção pedagógica serviu como um instrumento para medir a efetividade da intervenção pedagógica, quais as etapas da abordagem pedagógica PRIMM que os alunos mais e menos gostaram e por último quais os aspetos que alunos mais e menos gostaram nas aulas.

6. Dimensão Investigativa

Neste capítulo, irei apresentar o problema associado à dimensão investigativa, as questões elegidas, a metodologia adotada e os instrumentos utilizados para a recolha de dados e análise de dados.

6.1. Problema e Questões de Investigação

No decorrer da observação das aulas, constatei, como já referi em capítulos anteriores, que a Professora Cooperante adotou a abordagem pedagógica PrBL e que os alunos sentem algumas dificuldades na aprendizagem da programação.

Deparei-me, essencialmente, com duas dificuldades por parte dos alunos, a primeira associada à aprendizagem da linguagem C e os seus conceitos inerentes; a segunda associada à leitura e análise de blocos do código da linguagem C.

Neste sentido, a problemática de investigação está ligada à abordagem pedagógica utilizada, PRIMM. Em conjunto com a Professora Cooperante consideramos o seguinte problema de investigação: "Qual o papel da abordagem pedagógica PRIMM na melhoria da aprendizagem e dos conceitos de programação em linguagem C relativos à Estruturas de Dados Estáticas?".

As questões investigativas para dar resposta ao problema de investigação são:

$Q_{1:}$ Qual o efeito da utilização da abordagem pedagógica na aprendizagem dos conceitos relacionados com as estruturas de dados estáticas em linguagem C?

$Q_{2:}$ Que vantagens são reconhecidas pelos alunos na compreensão dos conceitos decorrentes da utilização da abordagem pedagógica?

$Q_{3:}$ Quais as principais dificuldades sentidas pelos alunos na utilização da abordagem pedagógica?

Neste estudo existem dois grupos, o grupo de controlo, T1, e o grupo experimental, T2. No T1 foi utilizada a metodologia *PrBL*, utilizada pela Professora Cooperante, enquanto no T2 foi aplicada a abordagem pedagógica PRIMM durante a minha intervenção. Este estudo é de pequena dimensão devido ao número de intervenientes na prática de ensino supervisionada ***n<=30***.

Os designs quasi-experimentais aplicam-se nas situações em que é difícil ou impossível um total controlo experimental. O verdadeiro mundo da educação, ou seja, o mundo com o qual o investigador em educação se confronta, está repleto de sérias limitações, relativamente à capacidade do investigador para selecionar os sujeitos ou atribuir-lhes as condições de manipulação. Assim, é possível que os sistemas escolares não aceitem novos programas para uma testagem com base experimental, não permitem que turmas intactas sejam desfeitas ou divididas para proporcionar amostras aleatórias ou equivalentes, não permitem ainda que seja dado um "tratamento" a algumas turmas e recusado a outras e, finalmente, não concedam a oportunidade de testagem previamente, com antecedência, a implementação de um determinado programa ou mudança (Tuckman, 2005).

O T1 e o T2 não foram criados de forma totalmente aleatória, mas sim, seguindo um critério de ordem alfabética pelo nome do aluno e/ou ordem de inscrição na turma. Neste sentido, a metodologia de investigação escolhida para este estudo foi, um estudo quantitativo, quase experimental, com avaliação realizada em dois momentos. No primeiro momento, a turma é submetida, antes da intervenção, a uma ficha de avaliação diagnóstica (pré-teste), e num segundo momento, após intervenção, a turma foi submetida a uma ficha de avaliação final (pós-teste).

Como variável dependente neste estudo, considero "Aprendizagem das Estruturas de Dados Estáticas C", será operacionalizada pelo pré-teste e pós-teste. Como variável independente considero a metodologia utilizada, PrBL no grupo de controlo e PRIMM no grupo experimental.

Apresento de seguida as hipóteses pré-teste e pós-teste testadas para o estudo em causa, nomeadamente, a hipótese nula e a hipótese um.

Hipótese pré-teste:

- H0: Os conhecimentos dos alunos sobre programação são equivalentes entre o grupo de controlo e o grupo experimental.
- H1: Os conhecimentos dos alunos sobre programação são diferentes estatisticamente entre o grupo de controlo e o grupo experimental.

Hipótese pós-teste:

- H0: A utilização da abordagem pedagógica PRIMM permite aos alunos obterem resultados equivalentes na aprendizagem de programação quando comparada com a utilização de uma abordagem baseada em problemas.
- H1: A utilização da abordagem pedagógica PRIMM permite aos alunos obterem melhores resultados na aprendizagem de programação quando comparada com a utilização de uma abordagem baseada em problemas.

6.2. Recolha de Dados

Para dar resposta às questões de investigação, apresento os seguintes instrumentos de recolha de dados que utilizei:

- Recolha documental das notas dos testes dos módulos anteriores;
- Ficha de avaliação diagnóstica (pré-teste);
- Grelhas de observação direta;
- Grelha de registo do desempenho dos alunos nas sessões PRIMM;
- Grelha de registo do procedimento prático (desafio Sequência Mágica);
- Ficha de avaliação final (pós-teste);
- Questionário de avaliação da intervenção;
- Questionário de autoavaliação dos alunos;

Com estes dados recolhidos e com resumos estatísticos efetuados, permitiram analisar o que correu bem e o que correu menos bem durante a prática de ensino supervisionada. De seguida passo a apresentar os resultados obtidos.

6.3. Apresentação dos Resultados

Tendo por base os dados recolhidos, tentei obter resposta a cada uma das questões de investigação.

6.3.1. Análise comparativa entre pré-teste e pós-teste

A análise comparativa entre o pré-teste e o pós-teste, pretende dar resposta à questão de investigação 1: Qual o efeito da utilização da abordagem pedagógica na aprendizagem dos conceitos relacionados com as estruturas de dados estáticas em linguagem C?

A **Tabela 5** apresenta as notas entre zero e vinte valores, obtidas pelos alunos no pré-teste e pós-teste. Nesta tabela, apenas estão listados 24 alunos, em vez dos 30 alunos, porque houve 6 alunos que faltaram ao pré-teste e/ou pós-teste. As notas destes 6 alunos foram retiradas para que a análise dos resultados entre pré-teste e pós-teste fosse coerente. Dos 6 alunos retirados, 1 pertence ao grupo de controlo e 5 ao grupo experimental.

Tabela 5

Notas pré-teste e pós-teste

Aluno	Grupo (Turno)	Nota pré-teste	Nota pós-teste
1	Controlo (T1)	16,00	13,00
2	Controlo (T1)	20,00	18,50
28	Controlo (T1)	4,00	16,50
3	Controlo (T1)	14,50	16,00
4	Controlo (T1)	11,00	15,50
5	Controlo (T1)	15,50	4,00
7	Controlo (T1)	5,50	16,00
8	Controlo (T1)	14,50	12,00
9	Controlo (T1)	3,00	10,00
10	Controlo (T1)	1,50	7,50
11	Controlo (T1)	2,00	11,00
12	Controlo (T1)	10,00	14,00
14	Controlo (T1)	16,00	14,00
30	Controlo (T1)	14,00	17,50
15	Experimental (T2)	13,50	15,50
16	Experimental (T2)	18,00	19,00
19	Experimental (T2)	13,50	11,00
22	Experimental (T2)	12,00	13,00
23	Experimental (T2)	14,00	10,00
24	Experimental (T2)	7,50	11,00
25	Experimental (T2)	13,50	11,50
26	Experimental (T2)	16,50	14,00
27	Experimental (T2)	9,50	17,00
29	Experimental (T2)	12,00	15,00

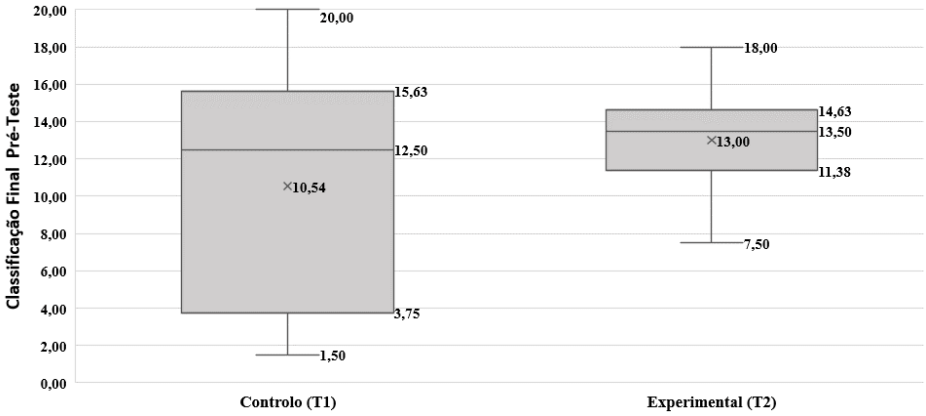
De uma forma global, os resultados das notas do teste inicial de conhecimentos (pré-teste) e do teste de avaliação final de conhecimentos (pós-teste), apresentam que

a maioria dos alunos que obtiveram nota negativa no pré-teste, conseguiram melhorar e tiveram positiva no pós-teste.

Quanto à análise dos resultados médios dos alunos no teste inicial de conhecimentos (pré-teste), representados na **Figura 39**, revelaram valores médios superiores no grupo experimental (M=13,00; DP=3,05; Med=13,50) relativamente ao grupo de controlo (M=10,54; DP=6,18; Med=12,50). Foi no grupo de controlo que se verificou a pontuação máxima no teste, 20 pontos, e a pontuação mínima 1,5 pontos. Em termos médios, considerando que a pontuação máxima da prova era 20 pontos, considera-se que os resultados globais dos dois grupos se situaram num nível satisfatório, verificando-se no grupo de controlo um maior número de resultados inferiores a 10 pontos, num total de 5, em comparação com o grupo experimental onde se verificaram apenas 2 resultados inferiores a 10.

Figura 39

Gráfico Blox-plot das Classificações Finais pré-teste



Para analisar a significância estatística das diferenças entre os resultados médios obtidos no pré-teste, de modo a compreender se os grupos são equivalentes em termos de conhecimentos, procedeu-se à aplicação de testes comparativos de médias para amostras independentes. Os pressupostos de aplicação do teste paramétrico t-Student foram analisados, em particular a normalidade da distribuição através do teste de Shapiro-Wilk (W=0,95; p=0,286), e a homogeneidade das variâncias através do

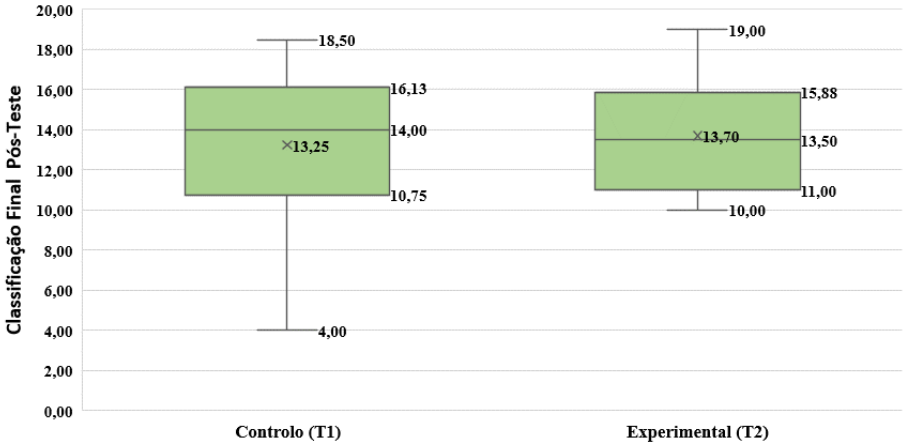
teste de Levene baseado na mediana (F(1,22)=9.12; *p*=0,006). Verificando-se que não está garantida a homogeneidade da variância optou-se pela utilização do teste não paramétrico Mann-Whitney U que analisa de forma comparativa os valores da mediana de cada um dos grupos.

A aplicação do Mann-Whitney U mostrou que as diferenças entre os grupos não se apresentam estatisticamente significativas (U= 61.5; *p*= 0,319). Deste modo aceitamos H0, ou seja, os conhecimentos dos alunos no teste inicial são estatisticamente equivalentes em ambos os grupos.

Após a experiência de ensino procedeu-se à aplicação de um teste de conhecimentos de programação (pós-teste), que integrava conteúdos avaliados no pré-teste e conteúdos novos aprendidos pelos alunos durante a sequência de aulas onde decorreu a experiência. A análise dos resultados médios dos alunos no teste final de conhecimentos (vide **Figura 40**), revelaram valores médios superiores no grupo experimental (M=13,70; DP=2,94; Med=13,50) relativamente ao grupo de controlo (M=13,25; DP=4,05; Med=14,00).

Figura 40

Gráfico Blox-plot das Classificações Finais pós-teste



Foi no grupo de experimental que se verificou a pontuação máxima no teste, 19 pontos, e a pontuação mínima, 4 pontos, ocorreu no grupo de controlo. Em termos

médios, considerando que a pontuação máxima da prova era 20 pontos, considera-se que os resultados globais dos dois grupos mantiveram-se satisfatórios. Salienta-se que no grupo experimental todos os alunos tiveram resultados superiores a 10 pontos e no grupo de controlo apenas 2 alunos tiveram resultados inferiores a 10, este aspeto foi uma melhoria em relação ao teste inicial.

Para analisar a significância estatística das diferenças entre os resultados médios obtidos no teste de conhecimento, de modo a compreender se a estratégia de ensino PRIMM apresenta efeitos significativos nas aprendizagens dos alunos, procedeu-se à aplicação de testes comparativos de médias para amostras independentes. Os pressupostos de aplicação do teste paramétrico t-Student foram analisados, em particular a normalidade da distribuição através do teste de Shapiro-Wilk (W=0,962; p=0,473), e a homogeneidade das variâncias através do teste de Levene baseado na mediana (F(1,22)=0,745; p=0,3,97). Garantidos os pressupostos a aplicação do teste t-Student mostrou que as diferenças entre os grupos não se apresentam estatisticamente significativas (t(22)= 0,299; p= 0,384). Deste modo, aceitamos H0, ou seja, os conhecimentos dos alunos no teste final são estatisticamente equivalentes em ambos os grupos.

Os resultados sinalizam que a utilização da abordagem pedagógica PRIMM não apresentou um efeito significativo nos resultados dos alunos. Estes resultados podem ser explicados em parte pelas dimensões das amostras, pela duração da experiência de ensino e pela proximidade das duas abordagens pedagógicas que procuram colocar os alunos a resolver pequenos problemas de programação como forma de aquisição de conhecimentos.

Tendo em consideração que por vezes os resultados dos alunos podem ser influenciados pelas suas competências prévias procurámos analisar os resultados tendo como covariável os resultados dos alunos no segundo módulo da disciplina, quer no pré-teste quer no pós-teste. Verificados os pressupostos à aplicação do teste ANCOVA, distribuição normal (W=0,92; p=0,06) e homogeneidade das variâncias (F(1,22)=1.66; p=0,21) os resultados indicam que as diferenças entre os grupos não

se apresentam estatisticamente significativas no pré-teste (F(1,21)=1,17; *p=0,291*) e no pós-teste (F(1,21)=0,013; *p=0,91*).

6.3.2. Análise ao Questionário de Avaliação da Intervenção

O questionário de avaliação da Intervenção (vide **Anexo Y**), foi aplicado na quinta aula da intervenção pedagógica e pretendeu dar resposta à questão de investigação 2: Que vantagens são reconhecidas pelos alunos na compreensão dos conceitos decorrentes da utilização da abordagem pedagógica?

Este questionário é constituído por quatro questões:

1º **Assinale o nível que mais se adequou às aulas ministradas pelo professor?** – esta questão foi criada com uma escala de likert com cinco níveis de resposta.

2º **Qual(ais) a(s) fase(s) da abordagem Pedagógica PRIMM que mais gostou?** - esta questão é constituída por cinco opções de escola múltipla.

3º **Indique os aspetos que menos gostou nas aulas?** – questão de reposta aberta

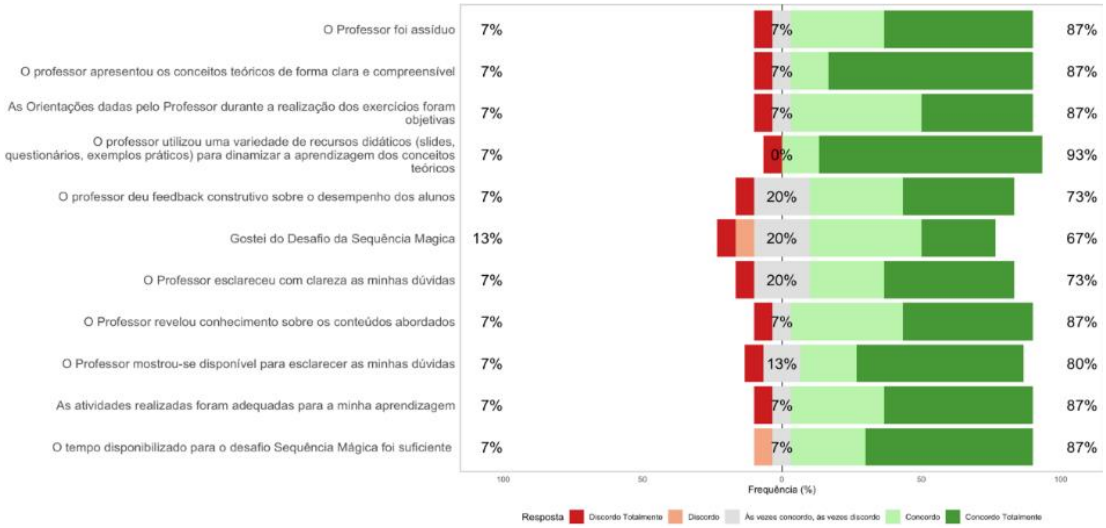
4º **Indique os aspetos mais gostou nas aulas?** - questão de resposta aberta.

Este questionário foi aplicado ao T2, e todos os alunos responderam à exceção de um aluno que faltou.

Os resultados apresentados na **Figura 41**, são referentes à primeira questão do questionário, e demonstram que as respostas dos alunos foram genericamente positivas.

Figura 41

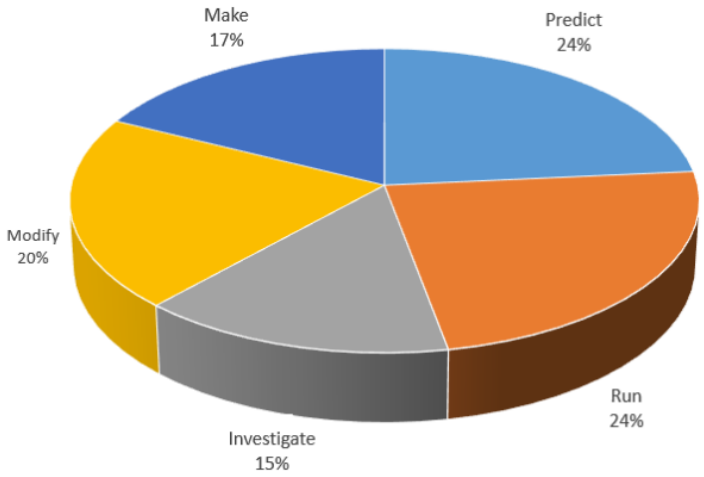
Frequências relativas das respostas dos participantes sobre a avaliação da intervenção (n=14)



Em relação a segunda questão do questionário, os resultados apresentados na **Figura 42**, mostram quais as etapas da abordagem pedagógica PRIMM que os alunos mais gostaram. As etapas ***Predict*** e ***Run*** foram as que os alunos mais gostaram, contrastando com as etapas ***Investigate*** e ***Make*** que foram as que os alunos menos gostaram.

Figura 42

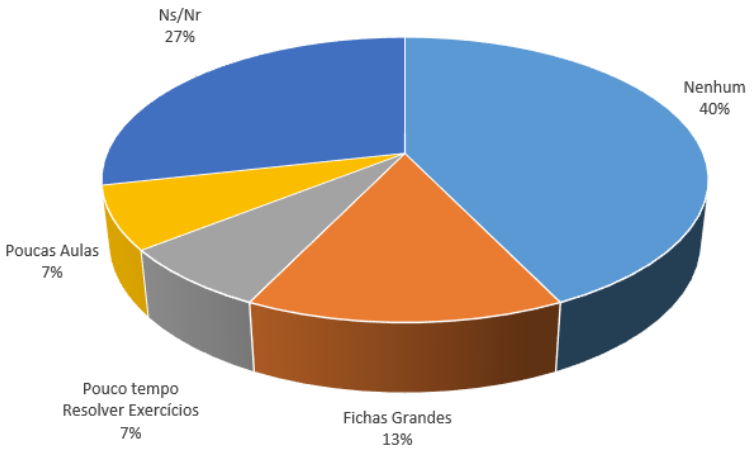
Distribuição das respostas dos participantes sobre as etapas da abordagem PRIMM que mais gostaram (n=14)



Em relação à terceira questão do questionário, agrupei as respostas dos alunos pelos aspetos por eles mencionados para poder ter uma visão global. A **Figura 43** apresenta os resultados agrupados pelos aspetos que os alunos menos gostaram na intervenção. Uma grande maioria dos alunos não apontou nenhum aspeto que não tenham gostado, já uma minoria dos alunos, indicaram que, os aspetos que menos gostaram foram, "*Fichas Grandes*", "*Poucas aulas*" e "*Pouco tempo para resolver os exercícios*".

Figura 43

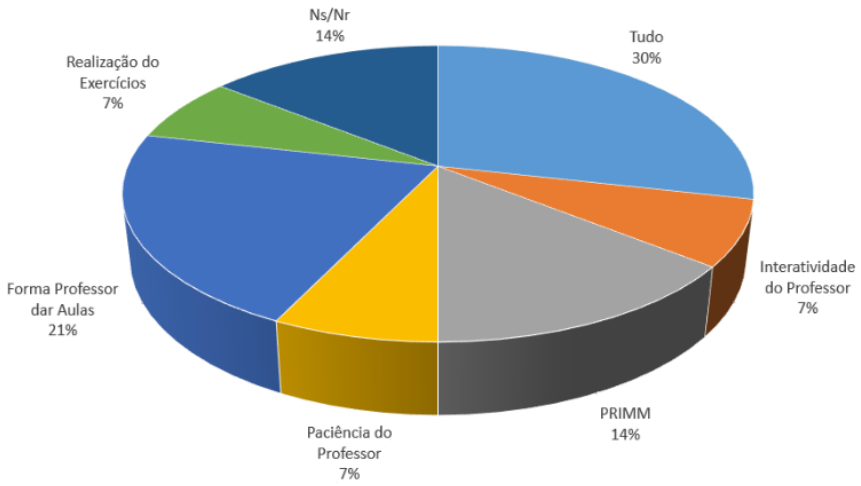
Distribuição das respostas dos participantes sobre os aspetos que menos gostaram nas aulas (n=14)



Em relação à quarta questão do questionário, agrupei as respostas dos alunos pelos aspetos por eles mencionados para poder ter uma visão global. A **Figura 44** apresenta os resultados agrupados pelos aspetos que os alunos mais gostaram na intervenção. Uma grande maioria dos alunos referiu que gostou de "*tudo*", "*A forma de o professor dar as aulas*" e "*PRIMM*", já uma minoria dos alunos, indicaram os aspetos que mais gostaram foram, "*Paciência do professor*", "*Realização dos exercícios*" e "*Interatividade do professor*".

Figura 44

Distribuição das respostas dos participantes sobre os aspetos que mais gostaram nas aulas (n=14)



6.3.3. Análise às grelhas das sessões PRIMM e observação direta

A terceira questão de investigação, "*Quais as principais dificuldades sentidas pelos alunos na utilização da abordagem pedagógica?*", foi respondida através de grelha de registo sessões PRIMM e grelhas de observação direta das aulas.

Após a realização das duas sessões PRIMM, concluo que as etapas que os alunos mais gostaram foram a ***Predict*** e ***Run*** e a etapas em que os alunos sentiram mais dificuldades foi ***Investigate*** e ***Make***, conforme já apresentado na **Figura 36**.

6.4. Conclusão

O relatório que resultou da observação e planificação da minha intervenção pedagógica ocorrida na ESACF, na turma do 1º ano do curso TGPSI, foi desenvolvido no âmbito da unidade curricular Iniciação à Prática Profissional III (IPP III). Esse relatório incluiu a caracterização do contexto escolar, o enquadramento curricular, a planificação da intervenção pedagógica, a avaliação da intervenção pedagógica e, a definição da problemática e questões investigação. A elaboração das duas sessões PRIMM e do cenário de aprendizagem, tornaram-se fundamentais na aplicação dos conceitos teóricos das aulas que lecionei, bem como, no envolvimento e cooperação dos alunos na realização das atividades práticas.

A implementação, avaliação da intervenção pedagógica e a análise da dimensão investigativa, foi realizada na unidade curricular IPP IV e culminou com a realização deste relatório.

Para a primeira questão de investigação "Qual o efeito da utilização da abordagem pedagógica na aprendizagem dos conceitos relacionados com as estruturas de dados estáticas em linguagem C?", pretendia-se averiguar se a abordagem pedagógica PRIMM foi um fator diferenciador na aprendizagem das estruturas de dados estáticas em linguagem C. Os alunos do grupo de controlo e do grupo experimental, foram submetidos a um pré-teste antes de intervenção e um pós-teste após a intervenção, e concluí que os conhecimentos dos alunos, no pós-teste, em ambos os grupos, são estatisticamente equivalentes e a abordagem PRIMM não foi fator diferenciador em relação ao PrBL pelos seguintes motivos; 1) as abordagens PrBL e PRIMM são parecidas, na medida em que colocam os alunos a desenvolver programas, mas de forma diferente, 2) a duração da aplicação da abordagem PRIMM foi curto, apenas duas aulas, onde não foi possível explorar com tempo as etapas da abordagem PRIMM, já Sentance & Waite, (2017), levou a cabo um estudo para implementar e avaliar a abordagem PRIMM em sala de aula, onde obteve resultados positivos, no entanto o estudo teve a duração de 14 semanas.

Relativamente à segunda questão de investigação, pretende averiguar "Que vantagens são reconhecidas pelos alunos na compreensão dos conceitos decorrentes

da utilização da abordagem pedagógica?", consegui, com base no questionário da avaliação da intervenção (vide **Anexo Y**), averiguar que, com base nos itens, "*As atividades realizadas foram adequadas para a minha aprendizagem*" e "*O professor utilizou uma variedade de recursos didáticos para dinamizar a aprendizagem dos conceitos teóricos*" os alunos reconheceram as vantagens na utilização da abordagem PRIMM e do desafio da Sequência Mágica na realização das atividades práticas. Observei ainda, que a maioria dos alunos participou ativamente nas aulas práticas demonstrando interesse e boa compreensão pelos conteúdos, mas também colaboração entre si, promovendo um bom ambiente na sala de aula.

Relativamente à terceira questão de investigação, "*Quais as principais dificuldades sentidas pelos alunos na utilização da abordagem pedagógica?*", averiguei quais das cinco etapas os alunos mais gostaram (fundamentado, vide **Figura 36**).

Em relação à primeira etapa, ***Predict***, os alunos gostaram desta etapa, por ser a primeira e aquela que despertou mais curiosidade, observei nos alunos entusiasmo na fase inicial da resolução das atividades práticas.

Na segunda etapa, ***Run***, os alunos também gostaram igualmente desta etapa, pelo facto de ser uma etapa simples de realizar.

Na terceira etapa, ***Investigate***, foi uma das etapas que os alunos menos gostaram, esta etapa requeria por parte dos alunos, a leitura, concentração e interpretação de um programa na linguagem C.

Na quarta etapa, ***Modify***, sendo uma etapa que não requeria muito esforço por ar e dos alunos, é considerada uma etapa que os alunos gostaram.

Em relação, à quinta e última etapa, **Make**, é uma etapa que requer por parte dos alunos o desenvolvimento de um programa completo em C, revelou foi uma etapa na qual os alunos menos gostaram.

Concluo, que as principais dificuldades sentidas pelos alunos, foram a leitura de um bloco de código ou programa em C, e o desenvolvimento de um programa novo programa, no entanto o balanço foi positivo, uma vez que foi possível desafiar os

alunos, proporcionar-lhes o acesso a novos conhecimentos, desenvolver competências e dar resposta ao problema de investigação. Para além destas dificuldades, observei três áreas em que os alunos também deveriam melhorar; 1) pontualidade, onde alguns apresentam atrasos ocasionais. 2) organização, em que alguns alunos não possuíam material escolar. 3) concentração, alguns alunos apresentam momentos de dispersão durante as aulas.

Em termos globais, concluo, que os objetivos da minha intervenção foram atingidos, que os alunos não foram prejudicados com a aplicação da abordagem pedagógica PRIMM e que as etapas ***Preditc*** e ***Run*** foram as que mais ajudaram a manter os alunos mais envolvidos da resolução das atividades práticas. Da mesma forma, Santos, 2022, concluiu num estudo efetuado numa turma do 1º ano do curso profissional de TGPSI que, a utilização da abordagem PRIMM, teve um impacto positivo no processo de aprendizagem inicial de programação estruturada, ajudou a aumentar a motivação e envolvimento dos alunos na atividades de sala de aula.

7. Balanço Reflexivo

Durante aproximadamente vinte anos exerci a minha atividade profissional em várias empresas na área da Tecnologia da Informação (TI), na quais desempenhei diversas funções nomeadamente, planeamento, análise de requisitos, desenvolvimento de software, testes de software e por último a implementação. Contudo, nos últimos anos também desempenhei papel de formador interno a novos colaboradores que ingressavam no departamento de informática. Com esta última função de formador interno, despertou em mim algo que sempre gostei de realizar, que é, a passagem do conhecimento. Já nessa altura, senti necessidade de criar diversos RED, para dar suporte à passagem do conhecimento, mas também, para estarem disponíveis para que os novos colaboradores os pudessem consultar sempre que necessário.

No ano de 2019, surgiu oportunidade de lecionar um ano letivo inteiro com horário completo uma turma do curso TGPSI, onde não hesitei, e agarrei este desafio de ser professor pela primeira vez. A gostar do desafio, e sentindo a necessidade de melhorar as minhas competências pedagógicas, inscreve-me, em 2020, no MEI na UL, com objetivo de obter a habilitação profissional para a docência. Foram quatro anos de trabalho intenso, mas muito enriquecedores pelas experiências vividas entre os colegas de turma e pelos conhecimentos na área pedagógica que adquiri, e que já os que consegui colocar em prática.

Todas as disciplinas foram importantes, contudo, as disciplinas de Didática da Informática e Introdução à Prática Pedagógica, foram essenciais neste meu percurso, para aprofundar o conhecimento na componente pedagógica.

O MEI culmina com a realização de uma intervenção pedagógica. Antes de a realizar, observei durante cerca de quatro meses as aulas da turma onde realizei a intervenção. Nestes meses tive contacto com a turma, observei os alunos, as metodologias e estratégias utilizadas pela Professora Cooperante. Foi muito proveitoso, a observação de aulas, porque rapidamente deixei de ter um papel passivo, em que só observava, e passei a ter um papel mais ativo na turma. Nas aulas práticas, já me sentia mais um professor dentro sala a ajudar os alunos na resolução de

exercícios práticos. Isto fez com que, começasse a conhecer melhor os alunos, planificar as atividades e a escolha da abordagem pedagógica que realizei na minha intervenção pedagógica.

Foi precisamente na planificação, onde senti mais dificuldades, das quais destaco: Que atividades a realizar? Qual a abordagem a utilizar? Que cenário de aprendizagem aplicar? nas seis aulas destinadas à intervenção. Para me ajudar nesta área da planificação, recorri à ajuda da Professora Cooperante e Professor Orientador que se tornaram fundamentais e me ajudaram a encontrar soluções.

A intervenção pedagógica acabou por ter sete aulas, mais uma aula do que inicialmente tinha previsto, e decorram, no meu ponto de vista de forma positiva. A escolha da abordagem pedagógica, PRIMM, foi escolhida com o intuito de aliar a aprendizagem das estruturas de dados estáticas a uma abordagem de ensino diferenciadora.

A escolha da abordagem pedagógica PRIMM, revelou-se uma escolha acertada, dado que as atividades que realizei assentes nesta abordagem, promoveram o trabalho colaborativo, o entusiasmo e o empenho dos alunos na realização das atividades.

Termino este relatório, afirmando que foi uma experiência única e foi mais um passo no meu percurso profissional, porque agora, posso dizer que, sou Professor, e um Professor também aprende todos dias com os seus alunos.

8. Referências

Anderson, L. W., & Krathwohl, D. R. (2001). *A taxonomy for learning, teaching, and assessing: A revision of Bloom's taxonomy of educational objectives: complete edition*. Addison Wesley Longman, Inc..

ANQUEP, C. P. (2018). *ANQUEP*. https://www.anqep.gov.pt/np4/cursos_profissionais.html

Damas, L. (2007). *Linguagem C . tradução João Araújo Ribeiro, Orlando Bernardo Filho.* (10ª).

Decreto Lei n.º 55. (2018). Decreto Lei n.º 55. *Diário Da Repúbilica, 1.ª série*(129), 2928–2943. https://dre.pt/application/conteudo/115652962

Direcção-Geral de Formação Vocacional. (2005). *Programa componente de formação técnica programação e sistemas de informação*.

Dorotea, N., & Pedro, N. (2015). Provas Digitais Online na Avaliação Formativa: Exploração das Práticas e Conceções dos Professores. *Atas Da IX Conferência Internacional de TIC Na Educação – Challenges*, 484–497. http://www.homeschoollaboratory.com/the-retention-toolkit-2-spacing-effect

dos alunos à saída da Escolaridade Obrigatória, P. (2016). *Despacho Nº 9311/2016, de 21 de julho*. https://diariodarepublica.pt/dr/detalhe/decreto-lei/396-2007-628017

EQAVET, D. (2021). *EQAVET – Quadro de Referência Europeu de Garantia da Qualidade para o Ensino e Formação Profissional*. https://www.dgert.gov.pt/eqavet-quadro-de-referencia-europeu-de-garantia-da-qualidade-para-o-ensino-e-formacao-profissional

Esau Taiwo, O., Christianah, A. O., Oluwatobi, A. N., Aderonke, K. A., & Kehinde, A. J. (2020). COMPARATIVE STUDY of TWO DIVIDE and CONQUER SORTING ALGORITHMS: QUICKSORT and MERGESORT. *Procedia Computer Science*, *171*(2019), 2532–2540. https://doi.org/10.1016/j.procs.2020.04.274

Gomes, A., Henriques, J., & Mendes, A. J. (2008). Uma proposta para ajudar alunos com dificuldades na aprendizagem inicial de programação de computadores. *Revista Educação, Formação & Tecnologias, May 2008*. https://www.researchgate.net/publication/277878385_Uma_proposta_para_ajudar_alunos_com_dificuldades_na_aprendizagem_inicial_de_programacao_de_computadores

Hanks, B., Fitzgerald, S., McCauley, R., Murphy, L., & Zander, C. (2011). Pair programming in education: a literature review. *Computer Science Education*, *21*(2), 135–173. https://doi.org/10.1080/08993408.2011.579808

Pi, R. (2021). *Teaching programming in schools: A review of approaches and strategies*. *November*.

Piedade, J., Nuno, D., Fábio, S. F., & Ana, P. (2019). A cross-analysis of block-based and visual programming apps with computer science student-teachers. *Education Sciences*, *9*(3). https://doi.org/10.3390/educsci9030181

Piedade, J., Pedro, A., & Matos, J. F. (2018). Cenários de aprendizagem como estratégia de planificação de aulas na formação inicial de professores: o exemplo da área de informática. *Educação e Tecnologias: Professores e Suas Práticas*, *November*, 13–36.

Piteira, M., & Costa, C. (2013). Learning computer programming. *Proceedings of the 2013 International Conference on Information Systems and Design of Communication*, 75–80. https://doi.org/10.1145/2503859.2503871

Piteira, M., & Haddad, S. R. (2011). Innovate in your program computer class. *Proceedings of the 2011 Workshop on Open Source and Design of Communication*, 49–54. https://doi.org/10.1145/2016716.2016730

Portaria n.º 235-A, de de 23 de agosto 2018. (2018). *Portaria n.º 235-A/2018, de 23 de agosto*. https://diariodarepublica.pt/dr/detalhe/portaria/235-a-2018-116154369

Portaria n.º 782, 2009. (2009). *Portaria n.º 782/2009, de 23 de julho*. https://diariodarepublica.pt/dr/detalhe/portaria/782-2009-493227

Portaria nº916. (2005). *Portaria n.º 916/2005, de 26 de setembro*. https://diariodarepublica.pt/dr/detalhe/portaria/916-2005-147643

Projeto, E. (2023). *Projeto educativo 2023 | 2026*. 1–52. https://ae4outubro.pt/documentos/projeto-educativo

Regulamento Interno. (2014). *Regulamento Interno*. https://ae4outubro.pt/documentos/regulamento-interno

Reis, P. (2011). *Observação de aulas e avaliação do desempenho docente. (Ministério da Educação, Ed.)*. http://www.ccap.min-edu.pt/pub.htm

Santos, M. J. F. dos. (2022). *Universidade de lisboa: Vol. I*. http://hdl.handle.net/10451/57750

Sentance, S., & Waite, J. (2017). PRIMM: Exploring pedagogical approaches for teaching text-based programming in school. *ACM International Conference Proceeding Series*, 113–114. https://doi.org/10.1145/3137065.3137084

Stinson, J. E., & Milter, R. G. (1996). Problem-based learning in business education: Curriculum design and implementation issues. *New Directions for Teaching and Learning*, *1996*(68), 33–42. https://doi.org/10.1002/tl.37219966807

Trevisan, A. L., & Amaral, R. G. do. (2016). A Taxionomia revisada de Bloom aplicada à avaliação: um estudo de provas escritas de Matemática. *Ciência & Educação (Bauru)*, *22*(2), 451–464. https://doi.org/10.1590/1516-731320160020011

Tuckman, B. W. (2005). *Manual de Investigação em Educação - Como conceber e realizar o processo de investigação em Educação*.

Túlio, A., & Cruz, F. (2023). *EQAVET - LISTA DE PARCERIAS Ano Letivo 2022-2023*. https://ae4outubro.pt/garantia-da-qualidade?view=article&id=154&catid=14

Williams, L., & Kessler, R. (2002). *Pair Programming Illuminated by Williams and Kessler*.

Xie, B., Loksa, D., Nelson, G. L., Davidson, M. J., Dong, D., Kwik, H., Tan, A. H.,

Hwa, L., Li, M., & Ko, A. J. (2019). A theory of instruction for introductory programming skills. *Computer Science Education*, *29*(2–3), 205–253. https://doi.org/10.1080/08993408.2019.1565235

9. Anexos

A. Lista de Entidades/Parceiro de Acolhimento de Estágios

Entidades de Acolhimento de Estágios	Entidades Empregadoras
Associação Inválidos do Comércio	Associação Luiz Pereira Motta
Comissão de Reformados, Pensionistas e Idosos da Póvoa de Santo Adrião	Casa de Repouso Solaris I
Casa do Artista	Centro Cultural e Social de Santo António dos Cavaleiros
Casa de Repouso Amélia Sena	Hospital Beatriz Ângelo
Hospital Privado da Trofa, S.A.	Lar Encosta da Saúde
Instituição de Apoio Social da Freguesia de Bucelas	
Residência Sénior Colinas do Cruzeiro	
Residência Sénior Quinta dos Apóstolos	
Boost Fit Club – Loures	
Colégio Crescer no Infantado	
Cooperativa Agrícola de Loures, CRL	
Escola de Tecnologias Navais	
AJSB- Assessoria e Gestão, Lda	
AECI - Arquitetura, Construção e Empreendimentos Imobiliários, S.A.	
AR TELECOM – Acessos e Redes de Telecomunicações, S.A.	
Associação "O Saltarico"	
Associação Rede do Progresso - "A Ponte"	
Bizarro e Milho, S. A. (Springfield)	
BMWT – Consultoria e Serviços Informáticos, Lda	
CARD4B, Systems S.A.	
Caseking Iberia Unipessoal, Lda.	
Centro de Dados da Defesa	

Centro Hospitalar Universitário Lisboa Norte, E.P.E. (Hospital de Santa Maria)	
CONFESPANHA - CONFECÇÕES, S.A.	
Conta Oculta, Lda. - Contabilidade, fiscalidade e documentação	
Cooperativa Agrícola de Loures, CRL	
Creche Rosa Azul	
FUNTRONICA, LDA	
GELPEIXE - Alimentos Congelados, S.A.	
GMC – Informática e Contabilidade, Lda.	
Gonti - Contabilidade e Gestão, Lda.	
GREEMOVEMENT, LDA	
Hospital da Luz, S.A.	
Hovione Farmaciência S.A.	
Ícones Irreverentes - Informática Unipessoal Lda	
INEDITHUNDER- Contabilidade e Serviços, Unipessoal, Lda.	
Juvex 3 – Equipamentos e Serviços Lda.	
LOGIC WISE, Lda	
LPoint - F. Rocha, Artigos Desportivos, Lda.	
Modelo Continente Hipermercados, S.A	
Montiqueijo - Queijos de Montemuro, Lda.	
OVERWAN, Lda	
SANTOS & VALE – SUL. DISTRIBUIÇÃO, LDA	
SDSR - Sport Division SR, S.A. (Sportzone)	
SEPHORA - Portugal Perfumaria, Lda	
Serconfins- Serviços Contabilísticos e Fiscais	
Servimetro, Serviços de Metrologia S.A.	
Somarmeios - Contabilidade e Gestão, Unipessoal, Lda.	
Teleperformance Portugal S.A.	

Tojal Conta III- Contabilidade, Fiscalidade e Auditoria Unipessoal, Lda.	
Transportes Aéreos Portugueses S.A.	
Triunfadrenalina	
UNISELF - Sociedade de Restaurantes Públicos e Privados, S.A.	
Vista Pró Índico - Investimentos Imobiliários, Lda.	

B. Critérios Avaliação Disciplina PSI

Agrupamento de Escolas 4 de OUTUBRO

REPÚBLICA PORTUGUESA
EDUCAÇÃO

PERFIL DO ALUNO			Competências Específicas		Ponderação (%)	Processos de recolha de Informação
Áreas de Competência		Descritores de Desempenho				
A	Linguagens e textos	1, 2	Conhecimentos e capacidades conceptuais e factuais	• Compreender as técnicas básicas de implementação de linguagens de programação • Conhecer a semântica de linguagens baseando-se em técnicas operacionais e técnicas de implementação de interpretadores • Analisar problemas reais da área da informática • Desenvolver soluções de software	65	Teste Trabalho de pesquisa Procedimento prático Trabalho de projeto Relatório de projeto Apresentação oral Observação direta
B	Informação e comunicação	1,2				
C	Raciocínio e resolução de problemas	1, 2, 3				
D	Pensamento crítico / Pensamento criativo	1, 2, 3				
E	Relacionamento interpessoal	1, 2, 3	Conhecimentos e capacidades processuais e comunicacionais	• Planear e apresentar de soluções informáticas • Diagnosticar e corrigir falhas no funcionamento de sistemas informáticos • Analisar de forma objetiva as linguagens de programação existentes • Instalar e efetuar a manutenção de aplicações informáticas • Utilizar ambientes e linguagens de programação orientada a objetos, procedimentais e visuais.	35	
F	Desenvolvimento pessoal e autonomia	1, 2, 3, 4				
G	Bem-estar, saúde e ambiente	1, 2, 3				
H	Sensibilidade estética e artística	1				
I	Saber científico, técnico e tecnológico	1,2,3				
J	Consciência e domínio do corpo	1,2,3				

Processo de recolha de informação avaliativa	Ferramenta	Observações
Teste	Critérios gerais de classificação	
Trabalho de pesquisa	Rubrica	
Procedimento prático	Critérios gerais de classificação/Rubrica	
Trabalho de projeto	Rubrica	
Relatório de projeto	Rubrica	
Apresentação oral	Rubrica	
Observação direta	Rubrica	

C. Planificação anual módulos 1, 2 3 e 4 da Professora Cooperante

Agrupamento de Escolas 4 de OUTUBRO

REPÚBLICA PORTUGUESA
EDUCAÇÃO

Programação e Sistemas de Informação - **1º Ano** - **Técnico Gestão e Programação de Sistemas Informáticos**

Nível de Ensino: **Curso Profissional** Professora: **Ana Gato**

Planificação Anual

Ano letivo de 2023/2024

Domínio/Conteúdos	Aprendizagens essenciais	Conteúdos	Materiais/Recursos	Processos de recolha de informação	Calendarização (aulas de 45 min)
Módulo 1 Introdução à Programação e Algoritmia	1. Apreender conceitos sobre a lógica de programação 2. Aplicar instruções e sequências lógicas na resolução de problemas 3. Utilizar as regras e as diferentes fazes na elaboração de um algoritmo 4. Aplicar fluxogramas 5. Identificar os diferentes tipos de dados 6. Identificar variáveis e constantes 7. Utilizar as regras de tipos em geral 8. Enumerar e identificar os operadores aritméticos, relacionais e lógicos 9. Utilizar operadores e funções pré-definidas 10. Realizar testes e correção de erros	1. Introdução à Lógica de Programação 1.1. Lógica 1.2. Sequência Lógica 1.3. Instruções 1.4. Algoritmos 2. Desenvolvimento de Algoritmos 2.1. Pseudocódigo 2.2. Regras e Fases de Construção de um Algoritmo 2.3. Fluxogramas 2.3.1. Introdução ao Fluxograma 2.3.2. Simbologia 3. Constantes, Variáveis e Tipo de Dados 3.1. Constantes 3.2. Variáveis 3.3. Tipos de Dados 4. Operadores e Funções Pré-Definidas 4.1. Operadores Aritméticos 4.2. Operadores Relacionais 4.3. Operadores Lógicos 4.4. Funções Pré-Definidas 5. Teste e Correção de erros	• Exposições/explicações orais feitas pela professora; • Realização de exercícios práticos no computador; • Fichas de trabalho; • Trabalho de grupo; • Apresentações de exemplos práticos; • Computadores; • Projetor multimédia; • Software: Visual G, Linguagem C, Internet Explorer; • Internet; • Grelhas de avaliação;	• Testes de avaliação • Questionários • Relatórios/questionários de atividades práticas • Apresentações orais e práticas • Listas de verificação/grelhas de observação	1 Semestre 34 aulas

Domínio/Conteúdos	Aprendizagens essenciais	Conteúdos	Materiais/Recursos	Processos de recolha de informação	Calendarização (aulas de 45 min)
Módulo 2 Mecanismos de Controlo e Execução de um programa	1. Conhecer vários tipos de variáveis 2. Compreender a estrutura de um programa. 3. Conhecer estruturas de decisão e de repetição	1. Exemplos em linguagem natural envolvendo mecanismos intuitivos de Decisão Binária e Decisão Múltipla 2. Exemplos em linguagem natural envolvendo mecanismos de repetição condicionada por uma expressão lógica 3. Desenvolvimento de algoritmos, fazendo uso de uma linguagem gráfica com o objetivo de analisar o seu fluxo de execução sequencial 4. Estrutura de um programa 5. Tipos de variáveis. Tipos simples 6. Instruções: Afetação, Input e Output de informação 7. Mecanismos de controlo de programa 8. Seleção simples 9. Seleção múltipla 10. Repetição condicional 11. Repetição incondicional	• Exposições/explicações orais feitas pela professora; • Realização de exercícios práticos no computador; • Fichas de trabalho; • Trabalho de grupo; • Apresentações de exemplos práticos; • Computadores; • Projetor multimédia; • Software: Visual G, Linguagem C, Internet Explorer; • Internet; • Grelhas de avaliação;	• Testes de avaliação • Questionários • Relatórios/questionários de atividades práticas • Apresentações orais e práticas • Listas de verificação/grelhas de observação • Utilização prática das TIC	1 Semestre 34 aulas

Domínio/Conteúdos	Aprendizagens essenciais	Conteúdos	Materiais/Recursos	Processos de recolha de informação	Calendarização (aulas de 45 min)
Módulo 3 Programação Estruturada	1. Adquirir a noção de subprograma; 2. Conhecer as regras de declaração de subprogramas; 3. Conhecer as regras de execução de subprogramas; 4. Utilizar corretamente parâmetros; 5. Distinguir os diferentes tipos de subprogramas; 6. Elaborar programas com recurso a subprogramas; 7. Conhecer as regras para a criação de bibliotecas de subprogramas; 8. Conhecer os mecanismos de utilização de bibliotecas de subprogramas.	1. Conceitos Básicos 2. Variáveis 2.1. Globais e Locais 2.2. Passagem por Parâmetros 3. Subprogramas 3.1. Estrutura do Subprograma 3.1.1. Procedimentos 3.1.2. Funções 3.2. Recursividade 4. Construção de Bibliotecas	• Exposições/explicações orais feitas pela professora; • Realização de exercícios práticos no computador; • Fichas de trabalho; • Trabalho de grupo; • Apresentações eletrónicas; • Vídeos relacionados com os temas; • Apresentações de exemplos práticos; • Computadores; • Projetor multimédia; • Internet; • Grelhas de avaliação; • Software: Windows, Microsoft PowerPoint, programa de edição de páginas Web(Microsoft FrontPage ou programa online), GIMP, Internet Explorer, Mozilla Firefox e Adobe Reader;	• Testes de avaliação • Questionários • Relatórios/questionários de atividades práticas • Apresentações orais e práticas • Listas de verificação/grelhas de observação • Utilização prática das TIC	2 Semestre 34 aulas

Domínio/Conteúdos	Aprendizagens essenciais	Conteúdos	Materiais/Recursos	Processos de recolha de informação	Calendarização (aulas de 45 min)
Módulo 4 Estruturas de dados Estáticas	1. Saber fazer a distinção entre uma variável simples e uma variável estruturada. 2. Saber o que é uma *String*. 3. Manipular uma *String*. 4. Diferenciar índice e valor indexado num *Array*. 5. Dominar os algoritmos de manipulação de *Arrays*. 6. Conhecer diferentes sistemas operativos e mecanismos de segurança;	1. Definição de *String* como variável capaz de guardar um número finito de valores do tipo CHAR 2. Declaração e Manipulação de variáveis do tipo *String* 3. Definição de Array como variável capaz de "agregar" um número finito de valores do mesmo 7. tipo 4. Declaração e Manipulação de variáveis do tipo *Array* 5. Estudo de algoritmos de manipulação de *Arrays* 6. Iniciação 7. Pesquisa sequencial 8. Inserção e remoção de elementos de um *array*: No Inicio (à Cabeça); no Fim (à Cauda). 9. Ordenação crescente ou decrescente dos elementos de um *array* 10. Inserção e remoção de elementos em *arrays* ordenados 11. *Array* de *Array* (ou *Array* multi-dimensional)	• Exposições/explicações orais feitas pela professora; • Utilização de ambientes digitais fechados; • Apresentações eletrónicas; • Vídeos relacionados com os temas; • Apresentações de exemplos práticos; • Computadores; • Projetor multimédia; • Software; • Internet; • Grelhas de avaliação;	Testes de avaliação Questionários Relatórios/questionários de atividades práticas Apresentações orais e práticas Listas de verificação/grelhas de observação Utilização prática das TIC	2 Semestre 34 aulas

D. Grelha Observação do Professor Cooperante

Quadro 17 – Grelha de observação focada: estratégias de ensino

Nome do professor:	Ano:	Turma:
Disciplina:	N.º de alunos:	Hora:
Observador:	Sala:	Data:

Comportamentos com impactos educativos positivos	Nada evidente	Algo evidente	Bem evidente
1. Expressa-se muito bem, tanto oralmente como por escrito.			
2. Fornece instruções de forma clara e concisa.			
3. Utiliza eficazmente as experiências, as ideias e os conhecimentos prévios dos alunos.			
4. Estimula e encoraja a participação dos alunos.			
5. Explica os conteúdos difíceis de mais de uma maneira.			
6. Utiliza diversas actividades na aula.			
7. Apresenta exemplos e demonstrações de determinados conteúdos.			
8. Utiliza meios audiovisuais diversos.			
9. Capta a atenção dos alunos.			
10. Reage e adapta-se às alterações de atenção dos alunos.			
11. Evidencia um entusiasmo sincero pelo tema da aula.			
12. Adequa as estratégias de ensino aos conteúdos.			
13. Adequa as estratégias de ensino à idade e às necessidades dos alunos.			
14. Demonstra excelente conhecimento dos conteúdos.			
15. Proporciona oportunidades aos alunos para que apliquem os conhecimentos.			
16. Estabelece relações entre os novos tópicos e os tópicos já conhecidos.			
17. Utiliza textos na aula.			
18. Estabelece relações entre os tópicos da aula e os tópicos das aulas anteriores e posteriores.			
19. Atribui aos alunos tempo adequado para responderem às perguntas.			
20. Termina com as distracções dos alunos de forma construtiva.			

Comentários gerais:

E. Grelha Observação Direta dos Alunos

Data :		Observação de Aulas																														
	Indicadores \ Alunos	A1	A2	A3	A4	A5	A6	A7	A8	A9	A10	A11	A12	A13	A14	A15	A16	A17	A18	A19	A20	A21	A22	A23	A24	A25	A26	A27	A28	A29	A30
1	Ponutualidade																														
2	Assiduidade																														
3	Participação																														
4	Comportamento																														
5	Respeita as regras de segurança/funcionamento da sala aula																														
6	Demonstra cooperação com os pares (trabalho em grupo)																														
7	Demostra interesse pelos temas																														
8	Demostra raciocinio e aprendizagem dos temas																														
9	Demostra capacidade na realização das tarefas																														
10	Demostra autonomia na realização das tarefas																														
11	Manuseamento do Software usado nas aulas																														
12	Uso da Plataforma de Gestão de Conteúdos (Teams)																														
13	Segurança Responsabilidade e Respeito em ambientes digitais																														
14	Comunicar e Colaborar																														

Legenda	
Nível	Valores (0-20)
Nível 1	0 a 4
Nível 2	5 a 9
Nível 3	10 a 13
Nível 4	14 a 17
Nível 5	18 a 20

F. Cenário de Aprendizagem Template Mini

Modelo de **Cenário** de **Aprendizagem**

TEL@ FTE LAB

Título: Sequência Mágica

Imagem que caracterize o cenário:

Objetivo Geral:
- Realizar um programa em C

Atividades:
- Os alunos irão efetuar a análise de requisitos do programa
sob acompanhamento do professor.
- Criar a interface para o programa
- Criar e iniciar os vetores do tipo Inteiro
- Gerar um número aleatório entre 1 e 50
- Pesquisar números nos vetores/arrays
- Preencher vetores/arrays
- Ordenar vetores/arrays

Objetivos Específicos:
- Distinguir uma variável simples de uma varável estruturada
- Distinguir índice e um valor indexado num Array
- Dominar os algoritmos de manipulação de Arrays
- Recordar as noções estruturas de decisão e repetição
- Compreender a definição de Array como variável capaz e agregar um número finito de valores do mesmo tipo
- Efetuar inserção e remoção de elementos de um Array
- Ordenar elementos num Array de forma crescente

Papéis:
- **Alunos**: mediante a análise de requisitos do programa os alunos deverão tirar as suas dúvidas com o professor, solicitar ao professor feedback e realizar o programa apresentado
- **Professor**: Irá desempenhar papel de moderador e orientador durante a criação do programa. Irá estimular a criatividade e o raciocínio dos alunos.

Interações:
- Os alunos irão trabalhar em grupos de 2, de forma a colaborarem entre si, com objetivo de desenvolverem capacidades de interajuda e relacionamento interpessoal.

Espaços:
- Sala de aula

Autor: Carlos Lopes Nº 23559, carloslopes1@edu.ulisboa.pt.

Trabalho desenvolvido no âmbito da unidade curricular de Iniciação à Prática Profissional III, do Mestrado em Ensino de Informática do Instituto de Educação da Universidade de Lisboa, no decorrer do ano letivo 2023/2024

Resumo da narrativa:

Durante as aulas da minha intervenção os alunos irão realizar diversos exercícios que permitem a manipulação de vetores de tipos de dados simples, tais como, inserção, pesquisa, ordenação e manipulação de cadeias de caracteres. Posteriormente os alunos irão criar o programa Sequência Mágica na linguagem de programação C. Este programa consiste em gerar duas sequências de números inteiros. A primeira sequência é composta por 5 números inteiros entre 1 e 50, a segunda sequência e composta por 2 números inteiros entre 1 e 9 gerados aleatoriamente pelo programa. Ambas as sequencias são armazenadas em dois vetores, respetivamente, pela ordem em que são gerados. Por fim, os vetores deverão ser apresentados ordenados de forma ascendente.

Durante os 14 tempos de 45 minutos, os alunos deverão conseguir criar o programa.

Technology Enhanced Learning @ Future Teacher Education Lab (PTDC/MHC-CED/0588/2014)

G. Cenário de Aprendizagem Template Normal

Modelo de **Cenário** de **Aprendizagem**

TEL@ FTE LAB

Disciplina: Programação e Sistemas de Informação
Módulo/ Unidade didática: Módulo 4 – Estruturas de dados Estáticas
Turma: 1º PGI
Autor: Carlos Lopes

Breve descrição

Este cenário de aprendizagem enquadra-se na disciplina de Programação e Sistemas de Informação do curso Técnico de Gestão e Programação de Sistemas informáticos, do ensino profissional, no módulo 4.

Este cenário de aprendizagem descreve um desafio que consiste no desenvolvimento de um programa em C, na medida em que os alunos irão colocar em prática os conhecimentos adquiridos nos módulos 1, 2, 3 e 4 mais concretamente: declarar variáveis, conhecer vários tipos de dados, compreender estrutura de um programa, conhecer estruturas de decisão e repetição, conhecer a noção de subprograma (procedimento/função), saber declarar e manipular um *array*, diferenciar entre índice e valor indexado, dominar os algoritmos de manipulação de *arrays*, Inserir e remover elementos num *arrays*

Este desafio será realizado em três fases. Numa primeira fase, o professor irá apresentar os conteúdos teóricos, para toda a turma em simultâneo pelo professor. Numa segunda fase, o professor irá colocar aos alunos diversos problemas cuja resolução irá de encontro à concretização do desafio. Na terceira e última fase, os alunos irão juntar a resolução dos diversos problemas de forma a terem o desafio realizado.

Objetivos de Aprendizagem

Este cenário tem por objetivo o desenvolvimento de um programa em C.

No final, o aluno deverá ser capaz de:

- Declarar variáveis
- Conhecer vários tipos de dados
- Compreender estrutura de um programa
- Conhecer estruturas de decisão e repetição
- Conhecer a noção de subprograma (procedimento/função)
- Saber declarar e manipular um *array*
- Diferenciar entre índice e valor indexado
- Inserir e remover elementos num *array*
- Ordenar elementos num *array*

Papel dos Alunos

Na primeira fase, em que todos os alunos da turma irão estar a recolher requisitos, efetuar questões ao professor, terão o papel de em conjunto fazerem um brainstorming sobre o projeto.

Na segunda fase e terceira fase, o papel dos alunos será trabalhar de forma individual, e desenvolver o projeto.

Com o desenvolvimento do projeto, pretende-se que os alunos adquiram as seguintes competências:

- Linguagens e textos;
- Informação;
- Compreender a Linguagem de programação C;
- Raciocínio e Resolução de Problemas;
- Pensamento Critico e Pensamento Computacional;
- Relacionamento Interpessoal;
- Desenvolvimento Pessoal e Autonomia;
- Saber científico, técnico e tecnológico;

Papel do Professor

Na primeira fase, o professor deverá apresentar o desafio à turma. Perante a turma, promover a comunicação entre alunos e professor para que os alunos possam colocar questões e efetuar esclarecimento sobre recolha de requisitos.

Na segunda fase, o professor deverá acompanhar, esclarecer todas as dúvidas dos alunos, assumindo papel de orientador durante a resolução dos diversos problemas que serão colocados aos alunos.

Este cenário irá promover ao professor, tendo por base DigCompEdu, um conjunto de competências apresentado em (http://www.digcomptest.eu/index.php?pg= competenciasDigitais).

Competências profissionais dos educadores

Área 1 – Envolvimento pessoal

1.3- Prática reflexiva – reflexão sobre os requisitos e o desenvolvimento do programa na sala de aula.

Competências pedagógicas dos educadores

Área 2 – Recursos digitais

2.1- Seleção – selecionar e identificar os recursos digitais para o ensino e aprendizagem, dos quais destaco, o computador com acesso à internet para aceder ao Teams da turma.

2.2 - Criação e modificação – Criação de recursos digitais, neste caso questionários do Google Forms, e disponibilizá-los aos alunos.

Área 3 – Ensino e aprendizagem

3.1- Ensino – Planificar e implementar dispositivos e recursos digitais no processo de ensino, de modo a melhorar a eficácia das intervenções pedagógicas. Gerir e orquestrar adequadamente estratégias de ensino digital. Experimentar e desenvolver novos formatos e métodos pedagógicos para o ensino.

3.2 – Orientação - Usar tecnologias digitais para proporcionar orientação e assistência oportuna e dirigida.

3.3 – Aprendizagem Colaborativa - Usar tecnologias digitais para promover e melhorar a colaboração do aprendente. Permitir que os aprendentes usem tecnologias digitais enquanto parte de tarefas colaborativas, como meio de melhorar a comunicação, a colaboração e a criação colaborativa de conhecimento. O desenvolvimento do projeto, irá permitir uma colaboração entre aluno-aluno e aluno-professor.

3.4 - Aprendizagem autorregulada - Usar tecnologias digitais para apoiar a aprendizagem autorregulada dos aprendentes, i.e., permitir que planeiem, monitorizem e reflitam sobre a sua própria aprendizagem, forneçam evidências de progresso, partilhem ideias e encontrem soluções criativas.

Área 4 – Avaliação

4.1 – Estratégia de avaliação - Usar tecnologias digitais para a avaliação formativa, neste caso concreto o Google Forms. Melhorar a diversidade e adequação dos formatos e abordagens de avaliação.

4.3 – Feedback e planificação - Usar tecnologias digitais para fornecer feedback oportuno e direcionado aos aprendentes.

Área 5 – Capacitação dos aprendentes

5.1 – Acessibilidade e inclusão - Garantir acessibilidade a recursos e atividades de aprendizagem para todos os aprendentes, incluindo os que têm necessidades especiais. Ter em consideração e dar resposta às expetativas, capacidades, usos e conceções erróneas (digitais) dos aprendentes, bem como ao uso contextual, físico e cognitivo que fazem das tecnologias digitais.

5.3 – Envolvimento ativo - Usar tecnologias digitais para promover o envolvimento ativo e criativo dos aprendentes com um assunto específico. Usar tecnologias digitais no âmbito de estratégias pedagógicas que fomentem as competências transversais dos aprendentes, a reflexão profunda e a expressão criativa.

Área 6 – Promoção da competência digital dos aprendentes

6.1 – Literacia da informação e dos media - Incorporar atividades, tarefas e avaliações de aprendizagem que requeiram que os aprendentes articulem necessidades de informação; encontrem informação e recursos em ambientes digitais; organizem, processem, analisem e interpretem informação; e comparem e avaliem criticamente a credibilidade e a fiabilidade da informação e das suas fontes.

6.2 – Comunicação e colaboração digital - Incorporar atividades, tarefas e avaliações de aprendizagem que requeiram que os aprendentes usem, eficaz e responsavelmente, tecnologias digitais para comunicação, colaboração e participação cívica.

6.3 – Criação de conteúdo digital - Incorporar atividades, tarefas e avaliações de aprendizagem que requeiram que os aprendentes se expressem através de meios digitais, modifiquem e criem conteúdo digital em diferentes formatos. Ensinar aos aprendentes como os direitos de autor e as licenças se aplicam ao conteúdo digital, como referenciar fontes e atribuir licenças.

6.4 – Uso responsável - Tomar medidas que garantam o bem-estar físico, psicológico e social dos aprendentes enquanto usam tecnologias digitais. Capacitar os aprendentes para gerir riscos e usar tecnologias digitais de forma segura e responsável.

6.5 – Resolução de problemas digitais - Incorporar atividades, tarefas e avaliações de aprendizagem que requeiram que os aprendentes identifiquem e resolvam problemas técnicos ou transfiram criativamente conhecimento tecnológico para novas situações.

Ferramentas e Recursos

Para desenvolver este cenário de aprendizagem serão necessários as seguintes ferramentas e recursos para os professores e alunos:

- Computador
- Acesso Internet
- DevC
- Projetor
- Tela
- Google Forms para aplicação na abordagem aprendizagem PRIMM
- Google Forms para questionários de avaliação desempenho do professor

Pessoas e lugares

Todas as fases deste cenário, irão decorrer na sala de aula, com o acompanhamento do professor cooperante e do professor (que faz a intervenção).

Metodologias de Aprendizagem

A metodologia adotada para o desenvolvimento deste cenário de aprendizagem, é PRIMM (Predict, Run, Investigate, Modify, Make).

A utilização desta metodologia neste cenário, permite aos alunos a consolidação dos conhecimentos adquiridos nas aulas bem como as competências que se pretendem para o aluno no Séc. XXI, são elas:

- Comunicação: Quer seja na primeira fase do projeto, na análise dos requisitos a comunicar com o professor, ou segunda e terceira fase do projeto, durante a realização do projeto a comunicar com o professor, os alunos irão desenvolver a competência da comunicação

- Colaboração: O programa será realizado individualmente ou em grupo, no enanto poderá haver uma entreajuda entre os alunos para a resolução de problemas que decorrerão no desenvolver do projeto.

- Criatividade: O projeto irá permitir aos alunos desenvolver esta competência.

- Pensamento crítico: Irão desenvolver esta competência nas resoluções dos problemas que irão surgir no decorrer do projeto.

No que diz respeito às estratégias de ensino serão utilizadas a resolução de problemas e o debate entre alunos e professor.

Tempos

Para este cenário de aprendizagem estão previstas 14 tempos de 45 minutos, totalizando 630 minutos.

A última aula, será de apresentação à turma dos projetos de todos os alunos e autoavaliação e avaliação de desempenho do professor

Avaliação

A avaliação do projeto irá ser efetuada durante o seu desenvolvimento da seguinte forma:

- Avaliação diagnóstica: Irei utilizar o teste sumativo do final do módulo 3, como forma de avaliação diagnóstica.

- Avaliação formativa: Durante a realização do desafio os alunos irão receber um feedback constante através da observação direta sobre o desempenho, o comportamento, interesse, criatividade e qualidade do trabalho apresentado.

- Avaliação desempenho do professor: Será apresentado aos alunos um questionário para avaliação da prática letiva por parte do professor.

Narrativa do **Cenário** de **Aprendizagem**

Título:

A narrativa do Cenário deve ser redigida para descrever a visão do ensino-aprendizagem da perspetiva do professor ou da perspetiva dos alunos. Considere-a como uma história que descreve a experiência de aprendizagem. Deve ter cerca de 500 palavras e pode descrever uma experiência de aprendizagem tão longa ou tão curta quanto se pretenda, por vezes numa só aula, mas normalmente abrangendo mais do que uma aula, como por exemplo um projeto cuja conclusão possa demorar várias aulas.

Durante as aulas da minha intervenção os alunos irão realizar diversos exercícios que permitem a manipulação de vetores de tipos de dados simples, tais como, inserção, pesquisa, ordenação e manipulação de cadeias de caracteres. Posteriormente os alunos irão criar o programa Sequência Mágica na linguagem de programação C. Este programa consiste em gerar duas sequências de números inteiros. A primeira sequência é composta por 5 números inteiros entre 1 e 50, a segunda sequência e composta por 2 números inteiros entre 1 e 9 gerados aleatoriamente pelo programa. Ambas as sequencias são armazenadas em dois vetores, respetivamente, pela ordem em que são gerados. Depois gerados os números e os vetores preenchidos, o programa deverá apresentá-los no ecrã ordenados por ordem crescente.

Os questionários foram criados no Goolge Forms:

Link para questionário de desempenho do professor: https://forms.gle/BPnFXv45jsx9bKv57

H. Cronograma Anual da Professora Cooperante

Agrupamento de Escolas 4 de OUTUBRO

REPÚBLICA PORTUGUESA
EDUCAÇÃO

ANO LETIVO 2023/2024

Curso Profissional de **Técnico de Gestão e Programação de Sistemas Informáticos**

CALENDARIZAÇÃO DA PLANIFICAÇÃO MODULAR ANUAL

1º Ano

Disciplina: PROGRAMAÇÃO E SISTEMAS DE INFORMAÇÃO

Identificação dos módulos/UFCD	Tempos letivos previstos	Tempos letivos previstos	Início do Módulo	Final do Módulo
N.º e Designação	Horas	45 minutos		
Módulo 1 - Introdução à Programação e Algoritmia	22	29	18/09/2023	17/10/2023
Módulo 2 - Mecanismos de controlo de execução	53	70	19/10/2023	09/01/2024
Módulo 3 - Programação estruturada	20	27	11/01/2024	15/02/2024
Módulo 4 - Estruturas de dados estáticas	32	42	19/02/2024	09/04/2024
Módulo 5 - Estruturas de dados compostas	26	35	11/04/2024	16/05/2024
Módulo 6 - Estruturas de dados dinâmicas	11	15	20/05/2024	04/06/2024
Módulo 7 - Tratamento de ficheiros	18	24	08/06/2024	27/06/2024
TOTAL DE TEMPOS LETIVOS ANUAIS	182	242		

Loures, 03 de outubro de 2023

A Docente:

Ana Rosa Gato

Coordenação do Ensino Profissional

I. Plano da Aula #1

<table>
<tr><td colspan="2">Escola Secundária Dr. António Carvalho Figueiredo</td><td colspan="2">Professor Cooperante: Ana Gato</td><td colspan="2">Módulo 4 - Programação e Sistemas de Informação</td><td>Turma: 1º PGI</td></tr>
<tr><td>Professor: Carlos Lopes</td><td>Data: 19/ fevereiro, segunda</td><td colspan="2">Hora: 08:30 – 10:00</td><td>Duração: 2 x 45min = 90min</td><td>Aula: 1</td><td>Sala: TIC 1</td></tr>
<tr><td colspan="4">Sumário:
Apresentação do projeto de intervenção.
Revisão dos conceitos da linguagem de programação C lecionados no módulo 1, 2 e 3.
Introdução aos tipos de dados Estruturados - Arrays</td><td colspan="3" rowspan="2">Recursos:
• Sala de informática
• Computadores com ligação à Internet
• Videoprojector
• Plataforma Teams
• Software DevC
• Recursos Educativos Digitais fornecidos pelo professor</td></tr>
<tr><td colspan="4">No final da aula, o aluno deverá ser capaz de:
• Compreender o conceito Array (vetor)
• Distinguir índice de valor indexado
• Conseguir declarar um Array de valores inteiros
• Aceder um a um elemento do Array</td></tr>
<tr><th>Conteúdos</th><th>Objetivos</th><th>Atividades</th><th>Metodologias e Estratégias</th><th>Avaliação</th><th>Tempos (mins)</th></tr>
<tr><td>• Revisão dos conceitos sobre Linguagem C</td><td>• Recordar conceitos de variável, estruturas de decisão, repetição</td><td>• Resumo do que foi lecionado nos módulos anteriores</td><td rowspan="2">• Expositivo
• Demonstrativo
• Ativo
• Interrogativo</td><td rowspan="2">• Formativa: Grelha de observação de empenho</td><td>20</td></tr>
<tr><td>• Tipos de dados estruturados: Array(vetor)</td><td>• Compreender o que é um Array, index (índice), valor indexado
• Compreender como se acede a um elemento do Array</td><td>• Realização de exercícios propostos pelo professor</td><td>70</td></tr>
</table>

J. Plano da Aula #2 e 3

Escola Secundária Dr. António Carvalho Figueiredo		Professor Cooperante: Ana Gato	Módulo 4 - Programação e Sistemas de Informação		Turma: 1º PGI
Professor: Carlos Lopes	Data: 20/ fevereiro, terça	Hora: 15:25 – 16:55	Duração: 5 x 45min = 225min	Aulas: 2 e 3	Sala: TIC 1
Sumário: Aceder e Afetar um elemento de um *Array* Percorrer/Iterar sobre todos os elementos de um *Array* Aplicação da abordagem PRIMM nos exercícios da aula			Recursos: • Sala de informática • Computadores com ligação à Internet • Videoprojector • Plataforma Teams / Forms • Software DevC • Recursos Educativos Digitais fornecidos pelo professor		
No final da aula, o aluno deverá ser capaz de: • Leitura de código em C • Afetar um elemento do *Array* • Iterar um *Array* • Conseguir declarar um *Array* de valores inteiros • Aceder um a um elemento do *Array*					

Conteúdos	Objetivos	Atividades	Metodologias eEstratégias	Avaliação	Tempos (mins)
• Leitura de código	• Analisar código de um programa • Modificar código de um programa	• Ler programa em C facultado pelo professor num formulário • Prever o que o programa faz • Registar o que programa faz • Executar o programa facultado pelo professor • Verificar se aluno previu com sucesso o que o programa faz • Modicar o programa de acordo com alterações propostas pelo professor	• Expositivo • Demonstrativo • Ativo • Interrogativo	• Formativa: Grelha de observação de empenho	50
• Afetar um elemento do *Array* • Aceder a um elemento do *Array* • Percorrer elementos do *Array*	• Compreender o que é um *Array*, *index* (índice), valor indexado. • Compreender como se acede a um elemento do *Array* • Avaliar conhecimento adquirido	• Realização de exercícios propostos pelo professor: o Programa que soma elementos de um *Array*; o Programa que conte os números pares de um *Array*;			175

K. Plano da Aula #4 e 5

<table>
<tr><td colspan="2">Escola Secundária Dr. António Carvalho Figueiredo</td><td>Professora Cooperante: Ana Gato</td><td colspan="2">Módulo 4 - Programação e Sistemas de Informação</td><td>Turma: 1º PGI</td></tr>
<tr><td>Professor: Carlos Lopes</td><td>Data: 26 e 27 fevereiro</td><td>Hora: 15:25 – 16:55</td><td>Duração: 4 x 45min = 180min</td><td>Aulas: 4 e 5</td><td>Sala: TIC 1</td></tr>
<tr><td colspan="3">Sumário:
Algoritmo de ordenação “selection order”
Apresentação do desafio da Sequência Mágica
Implementação do desafio da Sequência Mágica</td><td colspan="3" rowspan="2">Recursos:
• Sala de informática
• Computadores com ligação à Internet
• Videoprojector
• Plataforma Teams
• Software DevC
• Recursos Educativos Digitais fornecidos pelo professor</td></tr>
<tr><td colspan="3">No final das aulas, o aluno deverá ser capaz de:
• Ordenar um array utilizando algoritmo “selection order”;
• Aplicar os conceitos lecionados nas aulas 1, 2 e 3;
• Implementar o desafio da Sequência Mágica;</td></tr>
</table>

Conteúdos	Objetivos	Atividades	Metodologias eEstratégias	Avaliação	Tempos (mins)
• Algoritmo ordenação “*selection order*”	• Implementar o algoritmo de ordenação “*selection order*” • Modificar código de um programa • Avaliar conhecimento adquirido	• Realização de exercícios propostos pelo professor: o Programa que ordena elementos de um *Array*;	• Expositivo • Demonstrativo • Ativo • Interrogativo	• Formativa: Grelha de observação de empenho	120
• Implementação do desafio Sequência Mágica	• Implementar o programa Sequencia Mágica	• Implementação do programa Sequência Mágica			60

L. Plano da Aula #6

Escola Secundária Dr. António Carvalho Figueiredo		Professor Cooperante: Ana Gato	Módulo 4 - Programação e Sistemas de Informação		Turma: 1º PGI
Professor: Carlos Lopes	Data: 29 fevereiro, terça	Hora: 15:25 – 16:55	Duração: 3 x 45min = 135min	Aulas: 6	Sala: TIC 1
Sumário: Implementar o desafio da Sequência Mágica Apresentação à turma dos desafios implementados pelos alunos Autoavaliação Avaliação da intervenção pedagógica			Recursos: • Sala de informática • Computadores com ligação à Internet • Videoprojector • Plataforma Microsoft Forms • Formulário autoavaliação • Formulário de avaliação da intervenção Pedagógica		
No final da aula, o aluno deverá ser capaz de: • Implementar o desafio da Sequência Mágica; • Apresentar o desafio à turma					

Conteúdos	Objetivos	Atividades	Metodologias eEstratégias	Avaliação	Tempos (mins)
• Finalizar o desafio	• Finalizar desafio Sequencia Mágica (caso necessário)	• Implementação do programa Sequência Mágica	• Expositivo • Demonstrativo • Ativo • Interrogativo	• Formativa: Grelha de observação de empenho	45
• Apresentação do desafio Sequência Mágica à turma	• Apresentar o programa Sequencia Mágica	• Rever as atividades práticas realizadas.			45
	• Avaliar-se a si próprio.	• Preenchimento formulário autoavaliação	• Interrogativo	• Autoavaliação	25
	• Avaliação da intervenção pedagógica	• Preenchimento formulário avaliação da intervenção	• Interrogativo	• Avaliação da intervenção	20

M. Grelha de Registo do Procedimento Prático

Data : 12/03/2024	Procedimento Prático														
Critérios \ Alunos	Grupo I		Grupo II	Grupo III		Grupo IV	Grupo V		Grupo VI		Grupo VII		Grupo VIII	Grupo IX	
	A21	A18	A17	A23	A15	A16	A6	A25	A19	A20	A27	A22	A24	A26	A29
Qualidade Trabalho (com base nos objetivos do Procedimento)															
Organização do trabalho/ Criatividade															
Gestão do tempo/ Cumprimento de prazos															
Empenho															

N. Rúbrica – Procedimento Prático

Rubrica de Avaliação do Procedimento Prático					
Critérios	**Níveis de desempenho**				
	5	**4**	**3**	**2**	**1**
Qualidade do trabalho (com base nos objetivos do Procedimento)	- Apresenta **todos** os tópicos solicitados. - **Toda** a informação está formatada/estruturada corretamente conforme os objetivos definidos.	- Apresenta **todos** os tópicos solicitados. - **Mais de 50%** da informação está formatada/estruturada corretamente conforme os objetivos definidos.	- Apresenta **pelo menos 50%** dos tópicos solicitados. - **Mais de 50%** da informação está formatada/estruturada corretamente conforme os objetivos definidos.	- Apresenta **menos de 50%** dos tópicos solicitados. - **Menos de 50%** da informação não está formatada/estruturada corretamente conforme os objetivos definidos.	-Apresenta pelo menos **um tópico** solicitado.
Organização do trabalho/ Criatividade	- A informação está **muito bem organizada** de forma a facilitar a compreensão. - Realiza com iniciativa as atividades tomando as decisões **criativas** necessárias para atingir os objetivos definidos.	- A informação está **organizada** de forma a facilitar a compreensão. - Realiza com iniciativa as atividades tomando **algumas** decisões **criativas** necessárias para atingir os objetivos definidos.	- A informação está **organizada** de forma a facilitar a compreensão. - **Não** toma decisões criativas.	- A informação está **pouco organizada** de forma a facilitar a compreensão. - **Não** toma decisões criativas.	- A informação **não está organizada**. - **Não** toma decisões criativas.
Gestão do tempo/ Cumprimento de prazos	Realiza a **totalidade** da tarefa proposta **cumprindo** o prazo definido.	Realiza **mais de 50%** da tarefa proposta **cumprindo** o prazo definido.	Realiza **metade** da tarefa proposta **cumprindo** o prazo definido.	Realiza **menos de 50%** da tarefa proposta **no** prazo definido.	**Não realiza** a tarefa proposta no prazo definido.
Empenho	- Empenha-se **ativamente** na realização das tarefas. - Interessa-se, **de forma ativa**, em melhorar as suas competências.	- Empenha-se **ativamente** na realização das tarefas. - **Interessa-se** em melhorar as suas competências.	- **Empenha-se** na realização das tarefas. - Mostra **algum** interesse em melhorar as suas competências.	- Empenha-se **pouco** na realização das tarefas. - **Não** se Interessa em melhorar as suas competências.	- **Não** se empenha na realização das tarefas. - **Não** se interessa em melhorar as suas competências.

Observações:
Nível 1 – 0 a 4 valores; 0% a 19%
Nível 2 – 5 a 9 valores; 20% a 49%
Nível 3 – 10 a 13 valores; 50% a 69%
Nível 4 – 14 a 17 valores; 70% a 89%
Nível 5 – 18 a 20 valores; 90% a 100%

O. Grelha de Registo das Sessões PRIMM

Data :	Registo Sessão PRIMM																														
Etapas Sessões PRIMM \ Alunos	Grupo I			Grupo II			Grupo III			Grupo IV			Grupo V			Grupo VI				Grupo VII			Grupo IX			Grupo X		Grupo XI			Total por Etapa PRIMM
	A1	A2	A3	A4	A5	A6	A7	A8	A9	A10	A11	A12	A13	A14	A15	A16	A17	A18	A19	A20	A21	A22	A23	A24	A25	A26	A27	A28	A29	A30	
1ª Etapa - Predict																															0
2ª Etapa - Run																															0
3ª Etapa - Investigate																															0
4ª Etapa - Modify																															0
5ª Etapa - Make																															0
Total Por Aluno	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	

Legenda	
1	Mau
2	Insuficiente
3	Suficiente
4	Bom
5	Muito Bom

P. Questionário Avaliação Diagnóstica

Ficha Diagnóstico 1º PGI

1

Assinale a opção correta.
Um Algoritmo, é um conjunto de : (1.5 Pontos)

- instruções escritas com a ajuda de uma linguagem natural e que necessita de expressões precisas
- instruções finitas e ordenadas para resolver um problema
- instruções escritas de forma aleatória para resolver um problema
- instruções codificadas numa linguagem de programação

2

Dos seguintes nomes de identificadores (variáveis), indique os que **são inválidos**? (1.5 Pontos)

- [ ] Apelido
- [ ] área
- [ ] 10B
- [ ] Nome
- [ ] Preco/litro
- [ ] data_nascimento

3

Considere as seguintes linhas de código,
Qual é o seu Output? (1.5 Pontos)

```
int main() {
  int a = 10;
  for (int i = 0; i < 10; i++) {
    a++;
  }
  printf("%d\n", a);
  return 0;
}
```

- 10
- 20
- 30
- 40

4

Considere a seguinte linha de código, Qual é o tipo de dados adequado na declaração da variável **a** ? (1.5 Pontos)

```
__?__ a = 10.5;
```

- int
- float
- double

5

Qual será o output do seguinte programa? (1.5 Pontos)

```c
int main() {
  int a = 10;
  if (a > 5) {
    printf("a é maior que 5\n");
  } else {
    printf("a é menor ou igual a 5\n");
  }
  return 0;
}
```

- a é maior que 5
- a é menor ou igual a 5
- Nenhuma das anteriores

6

Escreva um programa em C que declare uma variável global chamada **a** e uma variável local chamada **b**. O programa deve atribuir o valor 10 à variável **a** e o valor 5 à variável **b**. Em seguida, o programa deve imprimir o valor das duas variáveis (4.5 Pontos)

Introduza a sua resposta

7

Escreva um programa em C solicite ao utilizador dois números inteiros e que apresente no ecrã a soma desses dois números.
A soma deverá ser realizada com recurso a uma função chamada **soma()** que recebe dois números inteiros como **parâmetros** e retorna a soma desses números.

(5.5 Pontos)

Introduza a sua resposta

8

Considerando a seguinte declaração do Vetor **notas**, qual é o Output do seguinte programa? (2.5 Pontos)

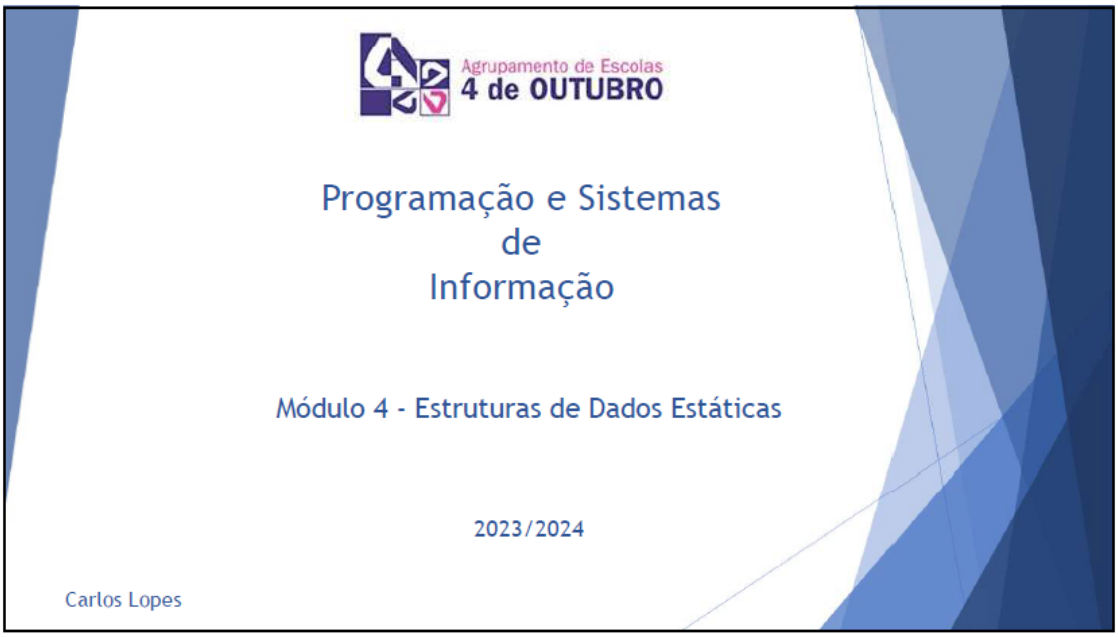
```c
#include <stdio.h>

int main() {
  int notas[] = {10, 8, 14, 19};

  printf("%d\n", notas[0]);
  return 0;
}
```

- 18
- 8
- 10
- 19

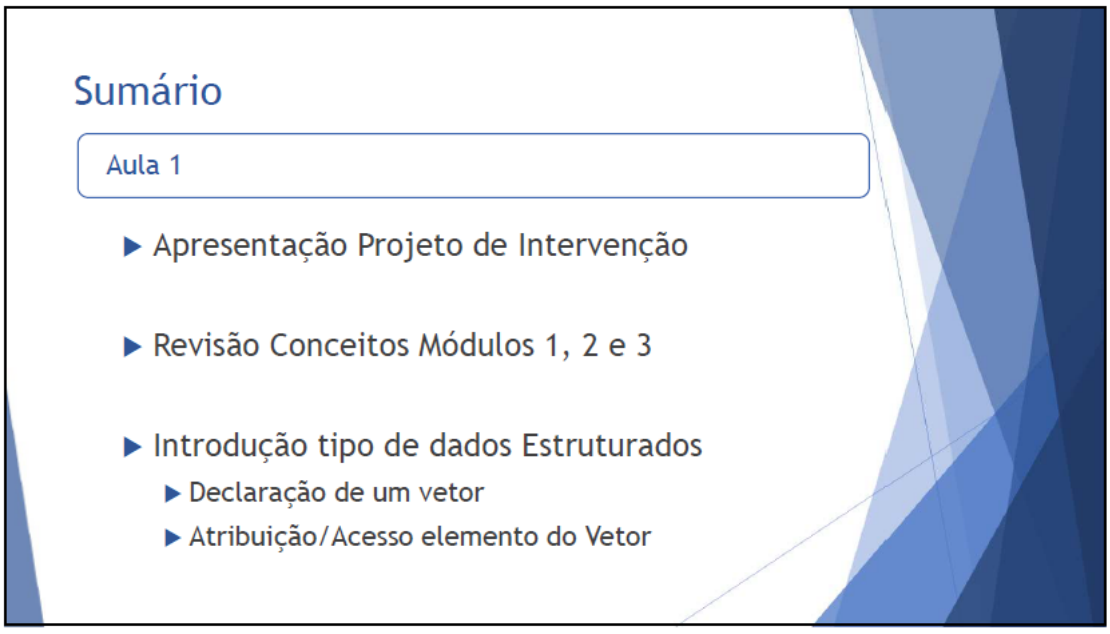
Submeter

Nunca revele a sua palavra-passe. Denunciar abuso

Q. Apresentação PowerPoint da Aula #1



1



2

Projeto Intervenção

- Lecionar 6 aulas
  - 04 Mar
    - Introdução às Estruturas de dados Estáticas
    - Exercícios práticos
  - 05 e 07 Mar
    - Aplicar abordagem PRIMM exercícios práticos
    - Algoritmos: Média, Soma, Pesquisa Valor num Vetor
  - 11 Mar
    - Algoritmo ordenação "*Selection Order*"
    - Apresentação do Desafio Sequência Mágica
  - 12 Mar
    - Realização do Desafio Sequência Mágica
  - 14 Mar
    - Realização do Desafio Sequência Mágica
    - Autoavaliação
    - Questionário de avaliação da intervenção
    - Ficha Diagnóstico Final

3

Revisão Conceitos Módulos 2 e 3

```
#include <stdio.h> //"adicionar ficheiro stdio.h ao programa"
Main()             //"onde programa começa"
{                  //Ínicio bloco instruções
printf             //Escreve uma string no ecrã
}                  //Fim bloco de instruções
int idade;         //Declarar variável idade do tipo inteiro
scanf()            //Leitura de dados para uma variável (&)
getchar()          //Leitura de dados para uma variável
const float PI = 3.14  //Declarar variável idade do tipo float
#define PI 3.14    //Declarar constante
```

4

Revisão Conceitos Módulos 2 e 3

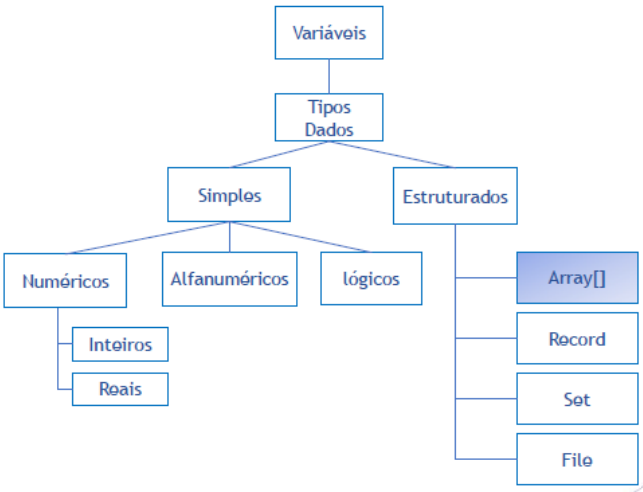
```
if (num > 0) {
   printf("O número %d é positivo.\n", num);
}else {
   printf("O número %d é negativo ou igual a zero.\n", num);
}
```

```
for (i = 1; i <= 10; i++) {
   printf("%d ", i);
}
```

```
int f1(int a, int b) {
   int resultado = a + b;
   return resultado;
}
```

5

TIPOS DADOS RESUMO



6

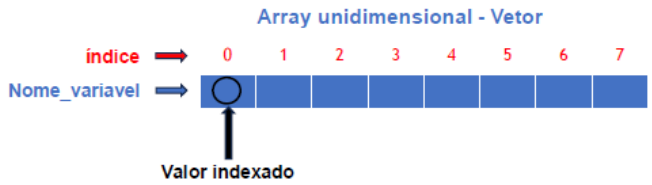
TIPOS DADOS ESTRUTURADOS

- A partir dos tipos de **dados simples** podem constituir-se outros tipos de dados mais complexos: **dados estruturados.**
  - **Array (Vetor)** - É identificado por um único nome e a sua estrutura é definida por vários elementos do mesmo tipo
  - **Record (Registo)** - Constituído por um conjunto de dados (logicamente relacionados) podendo ser de tipos diferentes
  - **Set (Conjunto)** - Constituído por um conjunto de dados semelhantes interrelacionados
  - **File (Ficheiro)** - É um tipo de dados que é armazenado em disco

7

VETOR (*ARRAY*)

- São variáveis, identificadas por um **nome**, que contêm um **conjunto de elementos** do mesmo **tipo.**
- São acessíveis através do respetivo nome da variável e do índice correspondente à sua posição no *Array*



8

DECLARAÇÃO DE UM VETOR (*Array*)

- `tipo nome_variável [nº de elementos];`
- Exemplos:
  - `int numeros[8];`

  - `float vencimentos[4];`

    ??
  - `int notas[6];`

    ??

9

ACEDER ELEMENTO DE UM VETOR (*Array*)

- Exemplos:
  - `int numeros[3];`

  - `numeros[0]` Primeiro elemento do vetor
  - `numeros[1]` Segundo elemento do vetor
  - `numeros[2]` Terceiro elemento do vetor

10

ATRIBUIÇÃO DE UM ELEMENTO DE UM VETOR

- `int numeros[8];`
  - `numeros[0] = 15;`
  - `numeros[2] = 7;`
  - …
  - `numeros[7] = 4;`

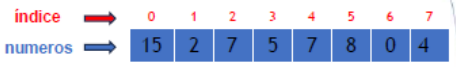
OU

```
for (i=0;i<=8;i++){
    printf("Introduza um número inteiro para a posição %d",i);
    scanf("%d",&numeros[i]);
  }
```

11

LEITURA DE UM ELEMENTO DE UM VETOR

- `int numeros[8];`



  - `printf("%d", numeros[1]);`
  - ? 2
  - `printf("%d", numeros[5]);`
  - ? 8

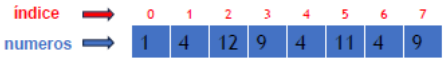
OU

```
for (i=0;i<=8;i++){
    printf("%d", numeros[i]);
}
```

12

INICIALIZAÇÃO AUTOMÁTICA DE UM VETOR

- É possível **declarar** e **inicializar**, automaticamente, todos os elementos de um vetor através da seguinte sintaxe:
  - `tipo nome_variavel[n] = {valor1, valor2, …, valorn};`
- Exemplo:
  - `int numeros[8] = {1, 4, 12, 9, 4, 11, 4, 9};`



13

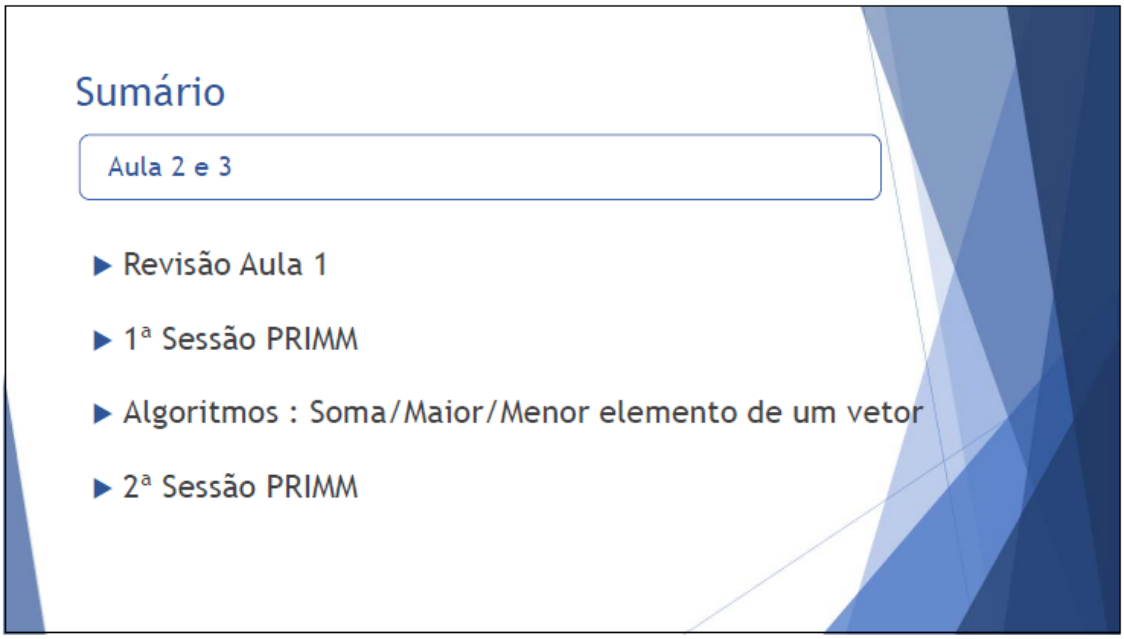
Exercício Prático

- Realizar no caderno em programa em C que:
  - **Declare** um vetor com 30 elementos inteiros com nome **notas_mod_2**
  - **Atribua** ao vetor a nota que obteve no módulo 2 de acordo seu N° aluno
    - Ex: Aluno N°1 atribuir à 1° posição do vetor a nota obtida módulo 2
    - Aluno N°2 atribuir à 2° posição do vetor a nota obtida módulo 2

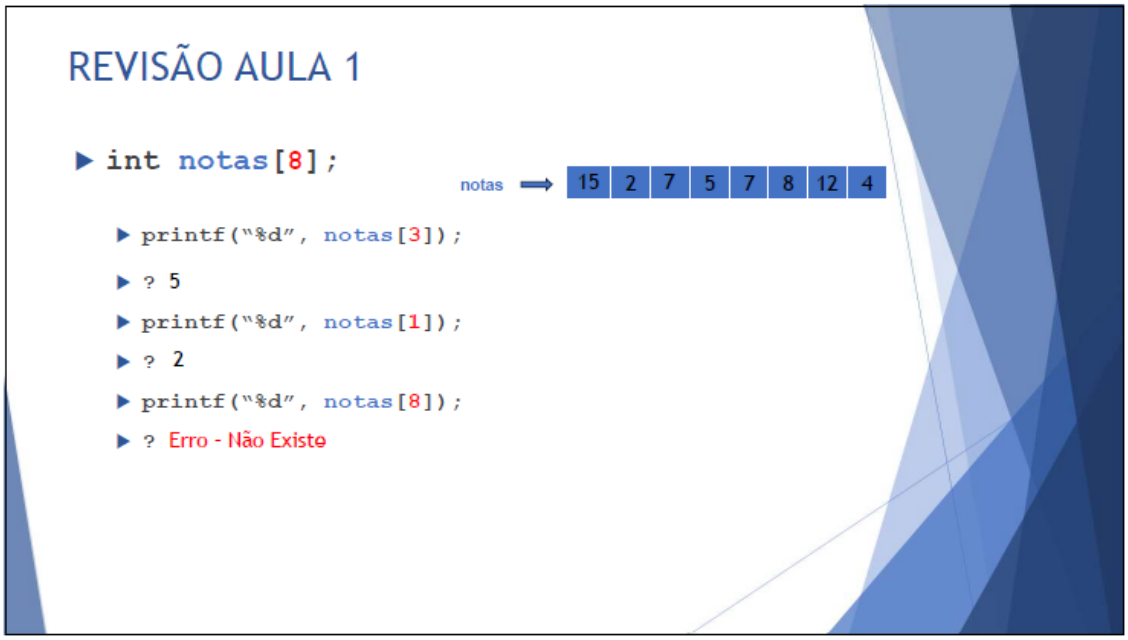
14

15

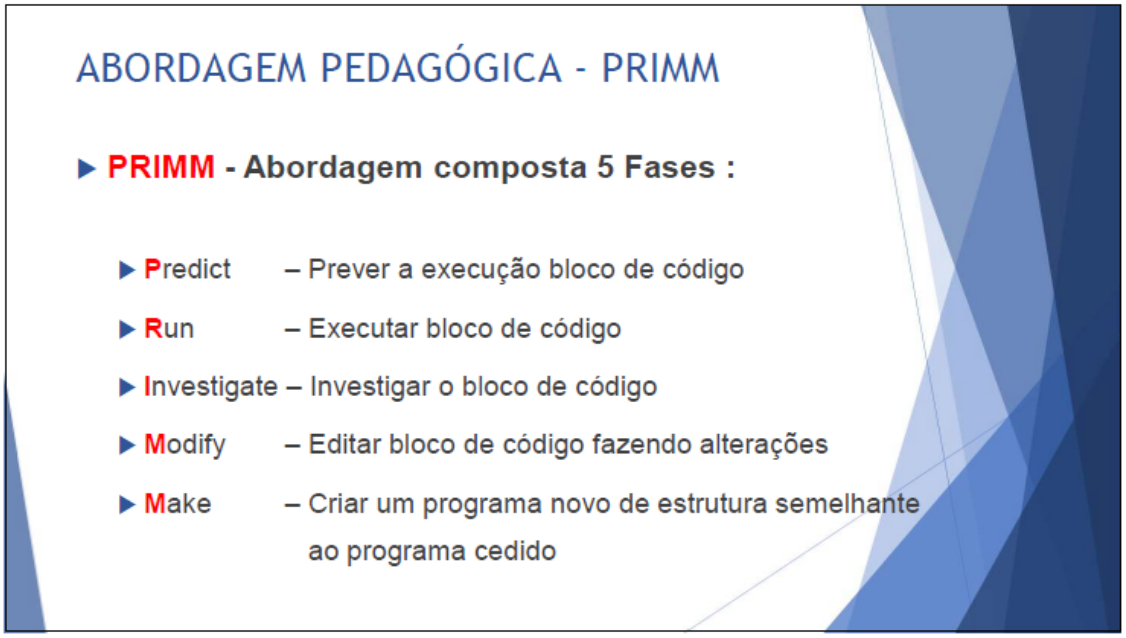
R. Apresentação PowerPoint da Aula #2 e #3



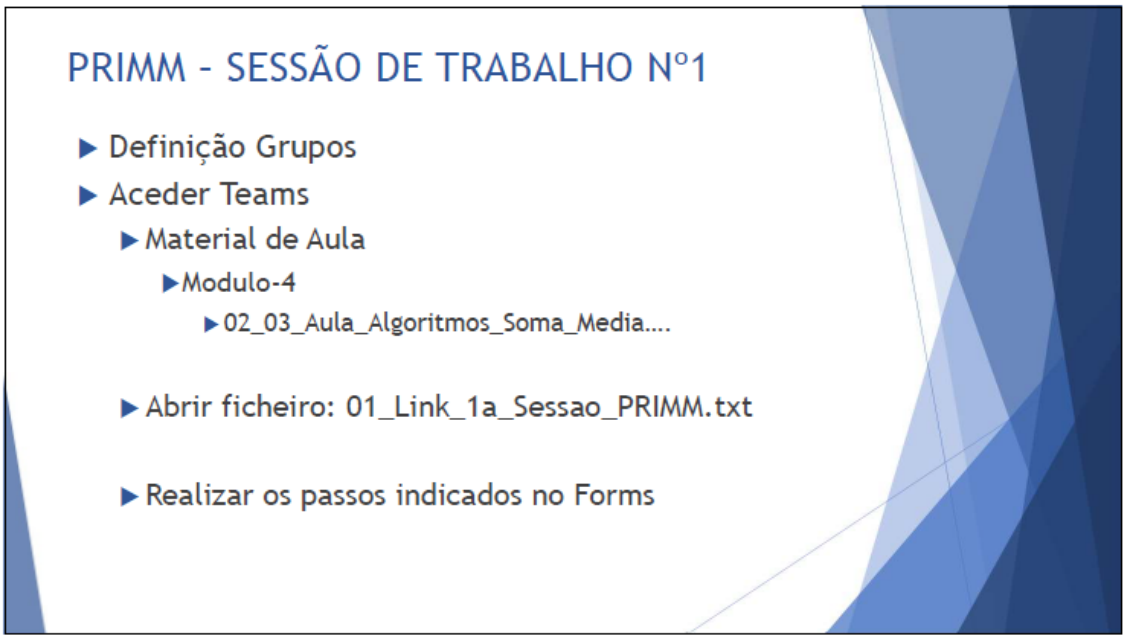
1



2


ABORDAGEM PEDAGÓGICA - PRIMM
PRIMM - Abordagem composta 5 Fases :
Predict – Prever a execução bloco de código
Run – Executar bloco de código
Investigate – Investigar o bloco de código
Modify – Editar bloco de código fazendo alterações
Make – Criar um programa novo de estrutura semelhante ao programa cedido

3


PRIMM - SESSÃO DE TRABALHO Nº1
Definição Grupos
Aceder Teams
Material de Aula
Modulo-4
02_03_Aula_Algoritmos_Soma_Media....
Abrir ficheiro: 01_Link_1a_Sessao_PRIMM.txt
Realizar os passos indicados no Forms

4

ALGORITMO:
DETERMINAR **SOMA** DOS VALORES DE UM VETOR

```
...
int numeros[8];
int soma = 0;
...
for (i=0; i<=7; i++)
    soma = soma + numeros[i];
...
```

5

ALGORITMO:
DETERMINAR **MÉDIA** DOS VALORES DE UM VETOR

```
...
int numeros[8];
int soma = 0;
float media = 0;
...
for (i=0; i<=7; i++)
    soma = soma + numeros[i];
media = soma/8;
```

6

ALGORITMO:
DETERMINAR **MAIOR** VALOR DE UM VETOR

```
…
int numeros[8];
int maior;
…
maior = numeros[0];
for (i=0; i<=7; i++)
    if (números[i]> maior)
        maior = numeros[i];
```

7

ALGORITMO:
PESQUISA SEQUENCIAL CICLO: **FOR**

```
Valor a pesquisar: 2 //Chave a Pesquisar
…
int numeros[8];
…
for (i=0; i<=7; i++)
    if (números[i] == 2)
        //encontrou CHAVE
```

8

PRIMM - SESSÃO DE TRABALHO Nº2

- Definição Grupos
- Aceder Teams
  - Material de Aula
    - Modulo-4
      - 02_03_Aula_Algoritmos_Soma_Media....
  - Abrir ficheiro: 02_Link_2a_Sessao_PRIMM.txt
  - Realizar os passos indicados no Forms

9

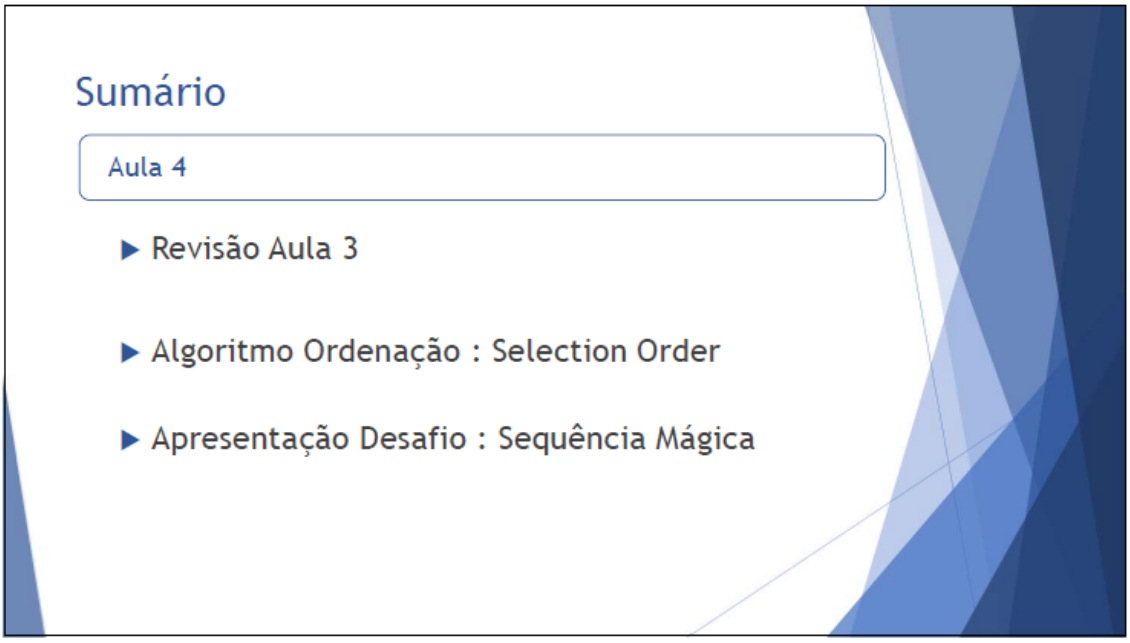
EXEMPLO PROGRAMA QUE SOLICITA 8 NÚMEROS

```
#include <stdio.h>
main()
{
   int numeros[8];
   int i;
   for (i=0;i<=7;i++)
    {
     printf("Introduza um numero inteiro para a posicao %d\n",i);
     scanf("%d",&numeros[i]);
    }
   printf("\n\n");
   for (i=0;i<=7;i++)
      printf("%d\n",numeros[i]);
}
```

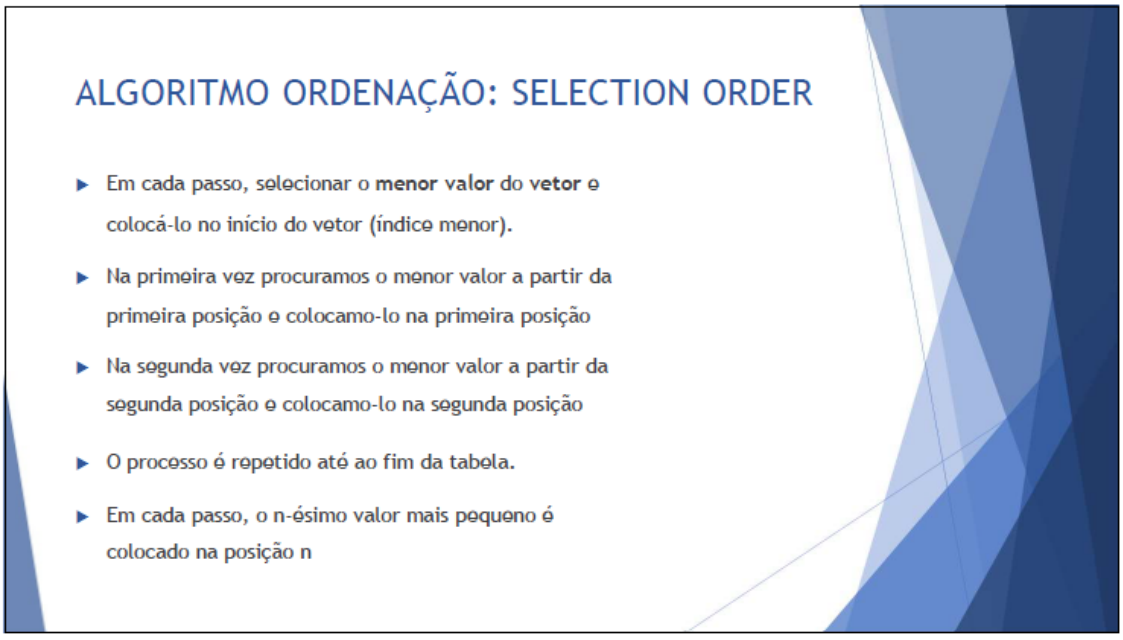
10

11

S. Apresentação PowerPoint da Aula #4



1



2

ALGORITMO ORDENAÇÃO: SELECTION ORDER

```
int numeros[n];

for(i=0; i<=(n-2); i++)
  for(k=i+1; k<=(n-1); k++)
    if (numeros[i] > numeros[k])
      {
        aux = numeros[i];
        numeros[i] = numeros[k];
        numeros[k] = aux;
      }
```

3

ALGORITMO ORDENAÇÃO: SELECTION ORDER.

```
int numeros[5]= {64, 25, 12, 22, 11};

for(i=0; i<=(5-2); i++)
  for(k=i+1; k<=(5-1); k++)
    if (numeros[i] > numeros[k])
      {
        aux = numeros[i];
        numeros[i] = numeros[k];
        numeros[k] = aux;
      }
```

	64	25	12	22	11
i=0 K=1	25	64	12	22	11
i=0 K=2	12	64	25	22	11
i=0 K=4	11	64	25	22	12
i=1 K=2	11	25	64	22	12
i=1 K=3	11	22	64	25	12
i=1 K=4	11	12	64	25	22
i=2 K=3	11	12	25	64	22
i=2 K=4	11	12	22	64	25
i=3 K=4	11	12	22	25	64

4

ALGORITMO ORDENAÇÃO: SELECTION ORDER

```
include <stdio.h>

main() {
   int numeros[10];
   int i,k,aux;

   //Guarda no vetor os números introduzidos pelo utilizador
   printf("Introduza 10 numeros inteiros\n\n");
   for (i=0;i<=9;i++)
      scanf("%d",&numeros[i]);
   printf("\n\n");

   //Ordenação do vetor por ordem crescente
   for(i=0;i<=8;i++)
       for(k=i+1;k<=9;k++)
          if (numeros[i] > numeros[k])
             {
               aux = numeros[i];
               numeros[i] = numeros[k];
               numeros[k] = aux;
             }
      //Escreve no ecrã o vetor por ordem crescente
      for(i=0;i<=9;i++)
         printf("%d ",numeros[i]);
      getch();
}
```

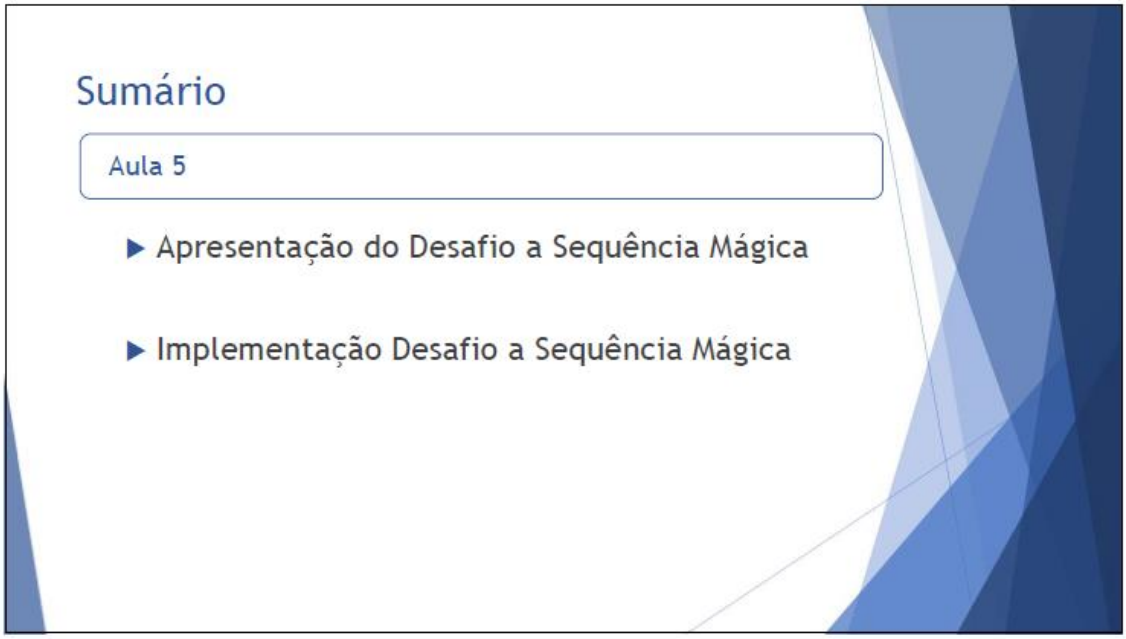
5

QUESTÕES

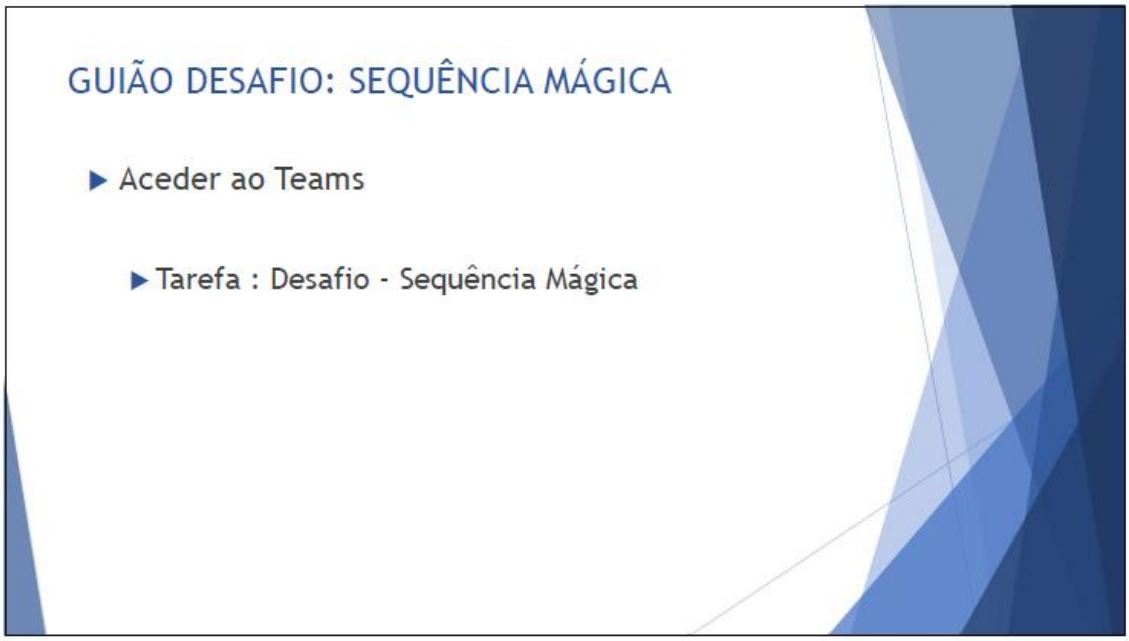
carlos.lopes@ae4outubro.pt

6

T. Apresentação PowerPoint Aula #5



1



2

U. Guião do Desafio : Sequência Mágica

Agrupamento de Escolas 4 de OUTUBRO

REPÚBLICA PORTUGUESA
EDUCAÇÃO

GUIÃO – DESAFIO **SEQUÊNCIA MÁGICA**

Objetivo: Com este desafio pretende-se que os alunos realizem em grupos de 2 elementos ou individualmente um programa em C que crie duas sequências de números inteiros.

A primeira sequência é composta por 5 números inteiros entre 1 e 50 gerados aleatoriamente.

A segunda sequência é composta por 2 números inteiros entre 1 e 9 gerados aleatoriamente pelo programa. Ambas as sequencias terão de ser armazenadas em dois vetores, pela ordem em que são gerados.

Por fim, os vetores deverão ser apresentados ordenados de forma ascendente.

Tempo Estimado: 90-120 minutos

Recursos:

- Computador com ambiente desenvolvimento **DevC** ou **online GDB** ou outro.
- Acesso à internet
- Slides do Professor
- Professor irá explicar a função **`srand()`** e **`rand()`** da biblioteca **`stdlib.h`**, que é utilizada para gerar números aleatórios.

  `rand()` – Retorna um número pseudoaleatório no intervalo de 0 a RAND_MAX. RAND_MAX é uma constante cujo valor padrão pode variar entre as implementações, mas é garantido que seja pelo menos 32767.

  `srand(unsigned int seed)` – função srand() alimenta o gerador de números aleatórios usado pela função **`rand()`**.

  `seed` – valor inteiro a ser usado como semente pelo algoritmo gerador de números pseudoaleatórios.

  `time(NULL)` – retorna o número de segundos decorridos desde 01/01/1970 00:00:00 horas, GMT.

Gerar número aleatório entre 1 e 10.

Exemplo:

```
#include <time.h>
#include <stdlib.h>
…
int numero;
int main(){
      srand(time(NULL));
      numero = rand();
} …
```

Etapas:

1. Etapa 1
   - Realização dos Grupos por parte dos alunos

2. Etapa 2
   - Desenvolver o programa Sequência Mágica, de acordo com os seguintes Tópicos:

     - Declaração dos vetores/variáveis
     - Definição dos limites inferior e superior com recurso a constantes
     - Programa deverá **garantir** que os números gerados aleatoriamente para a **primeira sequência** se situam entre **1 e 50**.
     - Armazenar a primeira sequência num vetor.

     Ex:

4	2	12	22	11

     - Programa deverá **garantir** que os números gerados aleatoriamente para a **segunda sequência** se situam entre **1 e 9**.
     - Armazenar a segunda sequência noutro vetor.

     Ex:

9	1

     - Utilizar a função srand() e rand() para gerar um número aleatório
     - O programa deverá apresentar no ecrã a primeira e a segunda sequência pela ordem quem os números foram gerados

- Por último, o Programa deverá apresentar no ecrã a **primeira** e **segunda** sequência ordenada com recurso ao algoritmo estudado na aula, **Selection Order**.

- Valorização de 10% se for utilizado ciclos de repetição para gerar as sequências.

- Valorização de 10% se for utilizado o recurso a funções (ex: gerar_Sequencias()).

Exemplo Output

```
Sequencia 1 ordem aletoria : 2 25 11 14 44
Sequencia 2 ordem aletoria : 4 1


Sequencia 1 ordenada : 2 11 14 25 44
Sequencia 2 ordenada : 1 4
--------------------------------
Process exited after 0.4615 seconds with return value 0
Press any key to continue . . .
```

V. Exemplo da resolução da Sequência mágica aluno nº 16

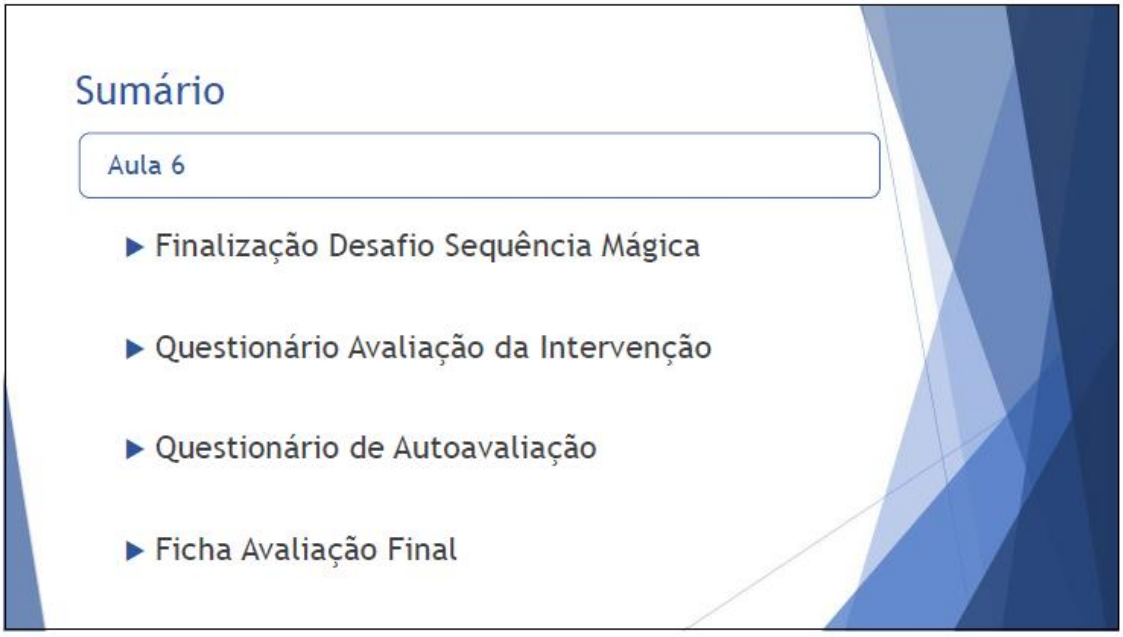
```c
#include <stdio.h>
#include <locale.h>
#include <math.h>
#include <time.h>
#include <stdlib.h>

int gerar_seq(int n_rep, int seq[],int limite){
    int i;
    srand(time(NULL));
    // gerar valores
    for(i=0;i<n_rep;i++){
        seq[i]=rand() % limite +1;
    }
    //retornar o array
    return (seq[i]);
}

int array_ord(int n_rep,int seq[]){
    int i,k,aux;
    for(i = 0 ; i < (n_rep-1) ; i++)
        for(k = (i+1) ; k < n_rep ; k++)
            if(seq[i] > seq[k]){
                aux=seq[i];
                seq[i]=seq[k];
                seq[k]=aux;
            }
    return (seq[i]);
}
main()
{
    setlocale(LC_ALL,"Portuguese");
    int n=5,limite=50;
    int seq1[n],i,k,aux;

    //gera os valores
    gerar_seq(n , seq1 , limite);

    //mostrar a primeira sequencia sem ordenacao
    printf("Primeira sequencia desordenada : ");
    for(i=0;i<n;i++)
        printf("%d\t",seq1[i]);

    printf("\n");

    //ordenar a primeira sequencia
    array_ord(n,seq1);

    // mostrar array1 ordenado
    printf("Primeira sequencia ordenada : ");
    for(i=0; i<n; i++)
        printf("%d\t",seq1[i]);

    printf("\n");
    n=2,limite=10;
    int seq2[n];

    //gerar os valores
    gerar_seq(n,seq2,limite);

    //mostrar a segunda sequencia sem ordenacao
    printf("Segunda sequencia desordenada : ");
    for(i=0;i < n;i++)
        printf("%d\t",seq2[i]);

    printf("\n");

    //ordenar a segunda sequencia
    array_ord(n,seq2);

    //mostrar a segunda sequencia ordenada
    printf("Segunda sequencia ordenada : ");
    for(i=0;i<n;i++)
        printf("%d\t",seq2[i]);

}
```

W. Exemplo da resolução da Sequência mágica do professor

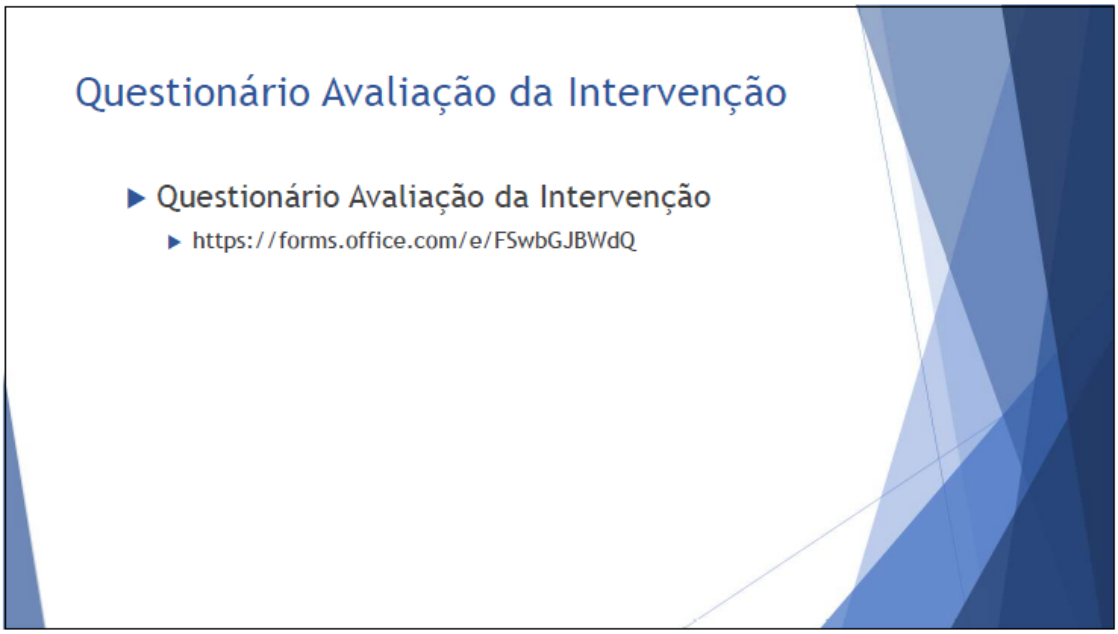
```c
#include <stdio.h>
#include <stdlib.h>
#include <time.h>

#define NUMERO_MAXIMO_SEQ_1 50
#define NUMERO_MAXIMO_SEQ_2 9

void gerarSequencias(int array_Seq_1[], int array_Seq_2[], int num_Elementos_Seq_1, int num_Elementos

int main() {
    // Inicializar o gerador de numeros aleatorios com a hora atual como semente
    srand(time(NULL));

    // Definir a quantidade de numeros da Sequencia 1 e Sequencia 2
    int i,k,aux;
    int num_Elementos_Seq_1 = 5;
    int num_Elementos_Seq_2 = 2;

    // Arrays para armazenar a Sequencia 1 e Sequencia 2
    int array_Seq_1[num_Elementos_Seq_1];
    int array_Seq_2[num_Elementos_Seq_2];

    // Gerar a Sequencia 1 e Sequencia 2
    gerarSequencias(array_Seq_1, array_Seq_2, num_Elementos_Seq_1, num_Elementos_Seq_2);

    // Mostrar a Sequencia 1 gererada
    printf("Sequencia 1 ordem aletoria : ");
    for (i = 0; i < num_Elementos_Seq_1; i++) {
        printf("%d ", array_Seq_1[i]);
    }
    printf("\n");
    // Mostrar a Sequencia 2 gererada
    printf("Sequencia 2 ordem aletoria : ");
    for (i = 0; i < num_Elementos_Seq_2; i++) {
        printf("%d ", array_Seq_2[i]);
    }

    printf("\n");
    printf("\n");
```

```c
    //Ordenar Sequencia 1 por ordem crescente
    for(i=0 ; i<=num_Elementos_Seq_1 - 2 ; i++){
        for(k=i+1 ; k<=num_Elementos_Seq_1 - 1 ; k++){
            if (array_Seq_1[i] > array_Seq_1[k])
            {
                aux = array_Seq_1[i];
                array_Seq_1[i] = array_Seq_1[k];
                array_Seq_1[k] = aux;
            }
        }

    }

    printf("\n");
    // Mostrar a Sequencia 1 ordenada
    printf("Sequencia 1 ordenada : ");
    for (i = 0; i < num_Elementos_Seq_1; i++) {
        printf("%d ", array_Seq_1[i]);
    }

    //Ordenar vetor Sequencia 2 por ordem crescente
    for(i=0 ; i<=num_Elementos_Seq_2 -2 ; i++){
        for(k=i+1 ; k<=num_Elementos_Seq_2 -1 ; k++){
            if (array_Seq_2[i] > array_Seq_2[k])
            {
                aux = array_Seq_2[i];
                array_Seq_2[i] = array_Seq_2[k];
                array_Seq_2[k] = aux;
            }
        }

    }

    printf("\n");
    // Mostrar a Sequencia 2 ordenada
    printf("Sequencia 2 ordenada : ");
    for (i = 0; i < num_Elementos_Seq_2; i++) {
        printf("%d ", array_Seq_2[i]);
    }

    return 0;
}

void gerarSequencias(int array_Seq_1[], int array_Seq_2[], int num_Elementos_Seq_1, int num_Elementos_Seq_2)

    int i;
    // Gerar numeros aleatorios para a Seq 1
    for (i = 0; i < num_Elementos_Seq_1; i++) {
        array_Seq_1[i] = rand() % NUMERO_MAXIMO_SEQ_1 + 1; // Seq 1 variam de 1 a 50
    }

    // Gerar numeros aleatorios para a Seq 2
    for (i = 0; i < num_Elementos_Seq_2; i++) {
        array_Seq_2[i] = rand() % NUMERO_MAXIMO_SEQ_2 + 1; // Seq 2 variam de 1 a 9
    }
}
```

X. Apresentação PowerPoint da Aula #6



1



2

Questionário Autoavaliação

- Questionário de Autoavaliação
  - https://forms.office.com/e/HuV4mFUrUy

3

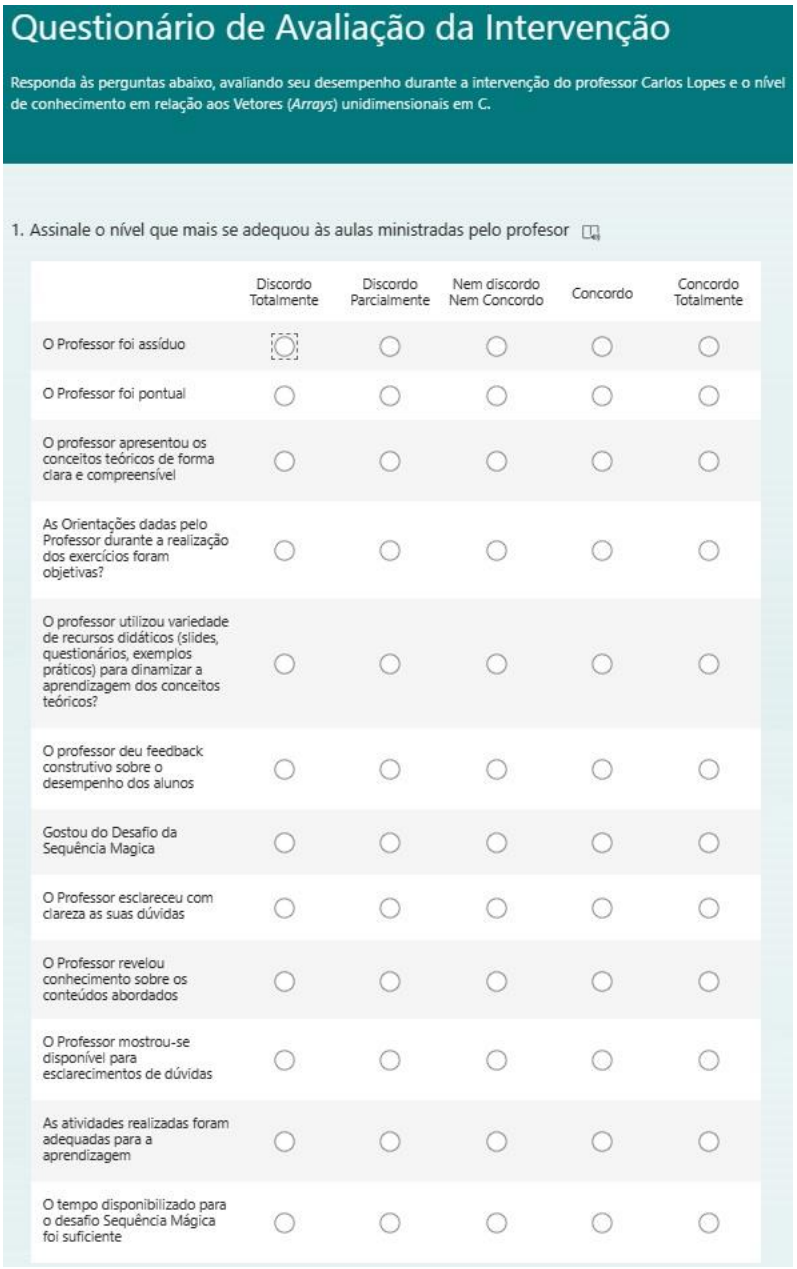
Ficha Avaliação

- Ficha Avaliação Final
  - https://forms.office.com/e/CM2XK22T1h

4

5

Y. Questionário de Avaliação da Intervenção



Questionário de Avaliação da Intervenção

Responda às perguntas abaixo, avaliando seu desempenho durante a intervenção do professor Carlos Lopes e o nível de conhecimento em relação aos Vetores (*Arrays*) unidimensionais em C.

1. Assinale o nível que mais se adequou às aulas ministradas pelo profesor

	Discordo Totalmente	Discordo Parcialmente	Nem discordo Nem Concordo	Concordo	Concordo Totalmente
O Professor foi assíduo	○	○	○	○	○
O Professor foi pontual	○	○	○	○	○
O professor apresentou os conceitos teóricos de forma clara e compreensível	○	○	○	○	○
As Orientações dadas pelo Professor durante a realização dos exercícios foram objetivas?	○	○	○	○	○
O professor utilizou variedade de recursos didáticos (slides, questionários, exemplos práticos) para dinamizar a aprendizagem dos conceitos teóricos?	○	○	○	○	○
O professor deu feedback construtivo sobre o desempenho dos alunos	○	○	○	○	○
Gostou do Desafio da Sequência Magica	○	○	○	○	○
O Professor esclareceu com clareza as suas dúvidas	○	○	○	○	○
O Professor revelou conhecimento sobre os conteúdos abordados	○	○	○	○	○
O Professor mostrou-se disponível para esclarecimentos de dúvidas	○	○	○	○	○
As atividades realizadas foram adequadas para a aprendizagem	○	○	○	○	○
O tempo disponibilizado para o desafio Sequência Mágica foi suficiente	○	○	○	○	○

2. Qual(ais) a(s) fase(s) da abordagem Pedagógica PRIMM que mais gostou?

- [ ] Predict (Prever)
- [ ] Run (Executar)
- [ ] Investigate (Investigar)
- [ ] Modify (Modificar)
- [ ] Make (Criar novo programa)

3. Indique os aspetos que menos gostou nas aulas?

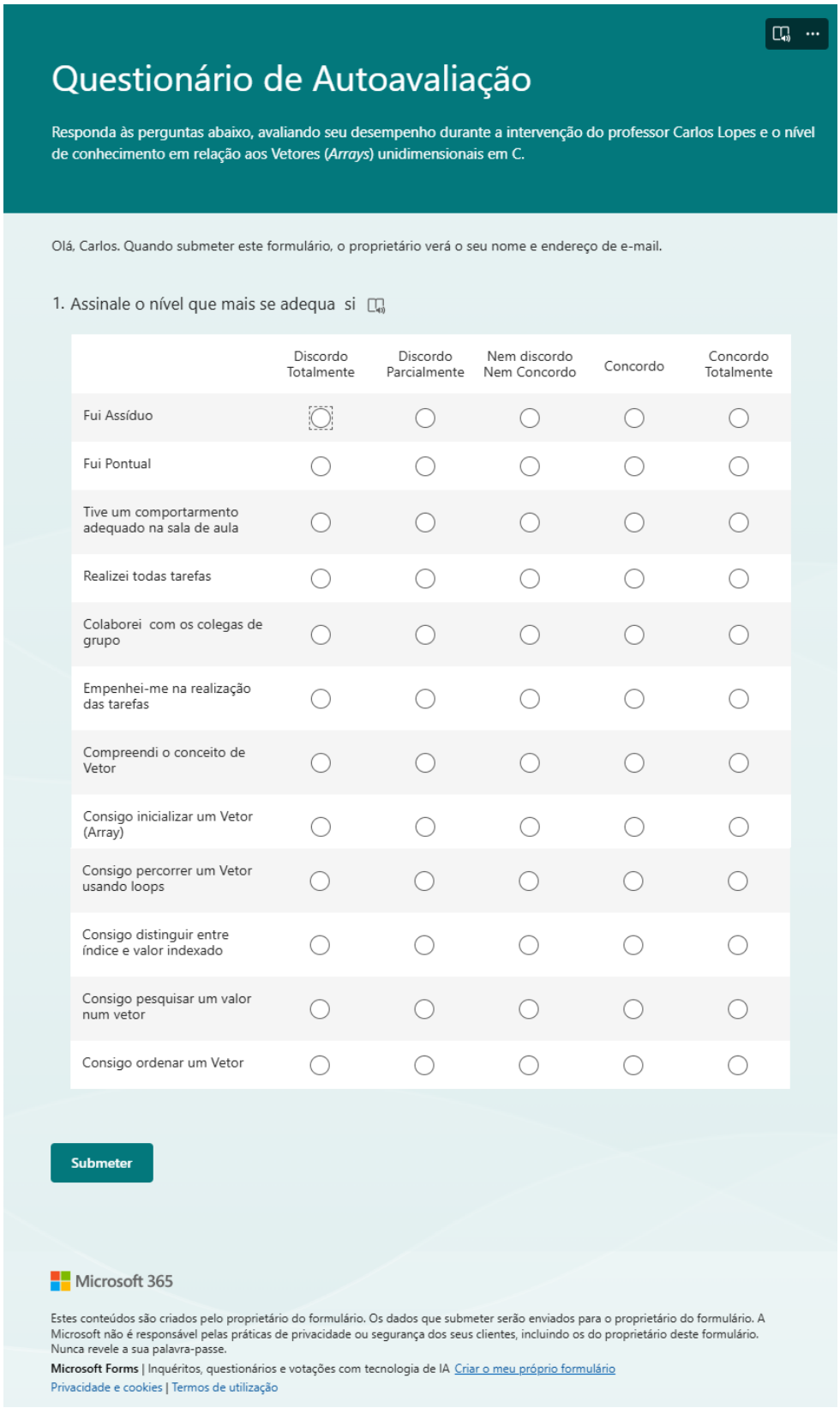
Introduza a sua resposta

4. Indique os aspetos mais gostou nas aulas?

Introduza a sua resposta

Submeter

Z. Questionário de Autoavaliação



Questionário de Autoavaliação

Responda às perguntas abaixo, avaliando seu desempenho durante a intervenção do professor Carlos Lopes e o nível de conhecimento em relação aos Vetores (*Arrays*) unidimensionais em C.

Olá, Carlos. Quando submeter este formulário, o proprietário verá o seu nome e endereço de e-mail.

1. Assinale o nível que mais se adequa si

	Discordo Totalmente	Discordo Parcialmente	Nem discordo Nem Concordo	Concordo	Concordo Totalmente
Fui Assíduo	○	○	○	○	○
Fui Pontual	○	○	○	○	○
Tive um comportarmento adequado na sala de aula	○	○	○	○	○
Realizei todas tarefas	○	○	○	○	○
Colaborei com os colegas de grupo	○	○	○	○	○
Empenhei-me na realização das tarefas	○	○	○	○	○
Compreendi o conceito de Vetor	○	○	○	○	○
Consigo inicializar um Vetor (Array)	○	○	○	○	○
Consigo percorrer um Vetor usando loops	○	○	○	○	○
Consigo distinguir entre índice e valor indexado	○	○	○	○	○
Consigo pesquisar um valor num vetor	○	○	○	○	○
Consigo ordenar um Vetor	○	○	○	○	○

Submeter

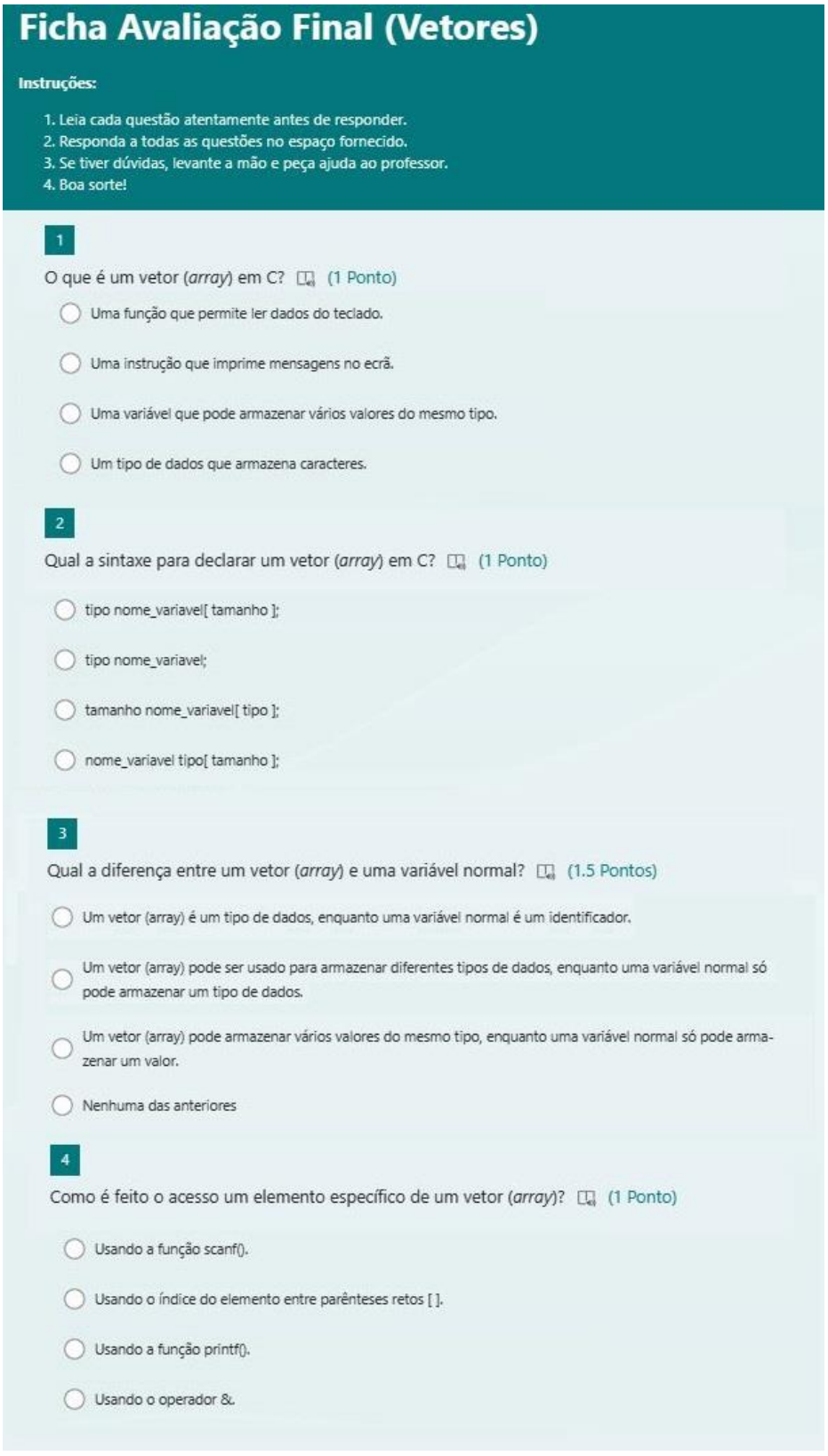
Microsoft 365

Estes conteúdos são criados pelo proprietário do formulário. Os dados que submeter serão enviados para o proprietário do formulário. A Microsoft não é responsável pelas práticas de privacidade ou segurança dos seus clientes, incluindo os do proprietário deste formulário. Nunca revele a sua palavra-passe.

Microsoft Forms | Inquéritos, questionários e votações com tecnologia de IA Criar o meu próprio formulário

Privacidade e cookies | Termos de utilização

AA. Questionário de Avaliação Final



Ficha Avaliação Final (Vetores)

Instruções:

1. Leia cada questão atentamente antes de responder.
2. Responda a todas as questões no espaço fornecido.
3. Se tiver dúvidas, levante a mão e peça ajuda ao professor.
4. Boa sorte!

1

O que é um vetor (*array*) em C? (1 Ponto)

- ○ Uma função que permite ler dados do teclado.
- ○ Uma instrução que imprime mensagens no ecrã.
- ○ Uma variável que pode armazenar vários valores do mesmo tipo.
- ○ Um tipo de dados que armazena caracteres.

2

Qual a sintaxe para declarar um vetor (*array*) em C? (1 Ponto)

- ○ tipo nome_variavel[tamanho];
- ○ tipo nome_variavel;
- ○ tamanho nome_variavel[tipo];
- ○ nome_variavel tipo[tamanho];

3

Qual a diferença entre um vetor (*array*) e uma variável normal? (1.5 Pontos)

- ○ Um vetor (array) é um tipo de dados, enquanto uma variável normal é um identificador.
- ○ Um vetor (array) pode ser usado para armazenar diferentes tipos de dados, enquanto uma variável normal só pode armazenar um tipo de dados.
- ○ Um vetor (array) pode armazenar vários valores do mesmo tipo, enquanto uma variável normal só pode armazenar um valor.
- ○ Nenhuma das anteriores

4

Como é feito o acesso um elemento específico de um vetor (*array*)? (1 Ponto)

- ○ Usando a função scanf().
- ○ Usando o índice do elemento entre parênteses retos [ ].
- ○ Usando a função printf().
- ○ Usando o operador &.

5

O que é um índice de um vetor (*array*)? (1 Ponto)

- ○ O tipo de dado armazenado no vetor (array).
- ○ O tamanho do vetor (array).
- ○ O nome do vetor (array).
- ○ A posição de um elemento no vetor (array).

6

Considerando a sintaxe da linguagem C, como faria para atribuir o valor 10 ao primeiro elemento do vetor **numeros**. (1 Ponto)

Introduza a sua resposta

7

Considere que o vetor **notas** de 5 posições armazena os seguintes valores: {2, 3, 4, 5, 6}. Qual será o valor de **notas**[2] após a seguinte instrução ser executada:

notas[2] = **notas**[0] + **notas**[1]? (1.5 Pontos)

- ○ 2
- ○ 3
- ○ 4
- ○ 5

8

Analise o código a seguir e selecione qual a opção que apresenta o output correto: (2 Pontos)

```c
int main() {
  int array[5] = {1, 3, 5, 7, 9};
  int i;
  for (i = 0; i < 5; i++) {
    array[i] = array[i] * 2;
  }
  for (i = 0; i < 5; i++) {
    printf("%d ", array[i]);
  }
  return 0;
}
```

- ○ 1 3 5 7 9
- ○ 2 4 6 8 10
- ○ 2 6 10 14 18
- ○ 5 10 15 20 25

9

Qual o erro presente no seguinte bloco de código? (2 Pontos)

```c
int main() {

  int array[5];

  array[6] = 10;

}
```

Introduza a sua resposta

10

A função **soma_vetor()** recebe um vetor de números inteiros e seu tamanho como parâmetros e retorna a soma de todos os elementos do array.
Responda/Indique qual é a assinatura/cabeçaho correto da função **soma_vetor()**?
(3 Pontos)

Introduza a sua resposta

11

Considere o seguinte vetor: **int notas[10] = {10, 9, 15, 17, 9, 11, 19, 12, 7, 11};**

Crie um programa em C no ambiente de desenvolvimento DevC ou OnlineGDB que declare o vetor de notas acimas apresentado e que encontre :
- A **maior nota** do vetor e imprima a **maior nota** e a sua **posição** no vetor (*array*).

Output pretendido: "A maior nota do vetor é 19, e está na posição 7 do vetor".

Cole o código do programa na caixa de texto.

(5 Pontos)

Introduza a sua resposta

Submeter

BB. Primeira Sessão PRIMM – Proposta de Resolução

Etapa 1: Predict

1

Analise o seguinte bloco de código, sem o executar, preveja o que ele faz, e indique na caixa de resposta qual o output do programa? *

```c
#include <stdio.h>

int main() {

  int i;
  int numeros[5];

  numeros[0] = 10;
  numeros[1] = 20;
  numeros[2] = 30;
  numeros[3] = 40;
  numeros[4] = 50;

  for (i = 0; i < 3; i++) {
    printf("Elemento %d do vetor: %d\n", i, numeros[i]);
  }

  return 0;
}
```

Elemento 0 do vetor : 10
Elemento 1 do vetor : 20
Elemento 2 do vetor : 30

Etapa 2: Run

Copie o seguinte bloco de código para ambiente de desenvolvimento **DevC** ou **Online GDB**, compile e execute o programa.

```c
#include <stdio.h>

int main() {

  int i;
  int numeros[5];

  numeros[0] = 10;
  numeros[1] = 20;
  numeros[2] = 30;
  numeros[3] = 40;
  numeros[4] = 50;

  for (i = 0; i < 3; i++) {
    printf("Elemento %d do vetor: %d\n", i, numeros[i]);
  }

  return 0;
}
```

2

O output do programa foi o mesmo que previu na etapa 1? *

- (●) Sim
- ( ) Não

3

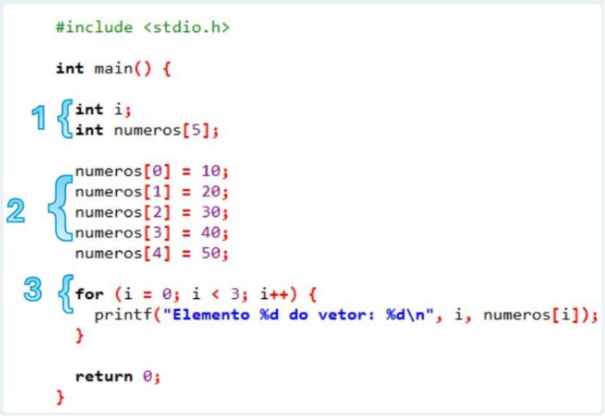
Coloque/Cole aqui o output do programa da etapa Run. *

Elemento 0 do vetor : 10
Elemento 1 do vetor : 20
Elemento 2 do vetor : 30

Etapa 3 : Investigate

4

Analise o seguinte bloco de código, e identifique o conceito de programação associado a cada bloco de código (1; 2; 3)



5

Bloco 1

- [ ] Instrução de atribuição
- [ ] Operação aritmética
- [ ] Estrutura de seleção
- [ ] Estrutura de repetição
- [x] Declaração de variáveis ✓

6

Bloco 2

- [x] Instrução de atribuição ✓
- [ ] Operação aritmética
- [ ] Estrutura de seleção
- [ ] Estrutura de repetição
- [ ] Declaração de variáveis

7

Bloco 3

- [x] Instrução de atribuição ✓
- [x] Operação aritmética ✓
- [ ] Estrutura de seleção
- [x] Estrutura de repetição ✓
- [ ] Declaração de variáveis

Etapa 4 : Modify

8

Altere o seguinte programa, para que apresente no ecrã, todos os elementos do vetor:
Responda apenas a linha de código que irá sofrer alteração.

```
#include <stdio.h>

int main() {

  int i;
  int numeros[5];

  numeros[0] = 10;
  numeros[1] = 20;
  numeros[2] = 30;
  numeros[3] = 40;
  numeros[4] = 50;

  for (i = 0; i < 3; i++) {
    printf("Elemento %d do vetor: %d\n", i, numeros[i]);
  }

  return 0;
}
```

```
for(i=0; i<5; i++)
```

Etapa 5 - Make

9

Crie um novo programa que solicite 5 números ao utilizador, e os armazene num vetor chamado **numeros**, e de seguida escreve o conteúdo do vetor **numeros** no ecrã.

Observações : Crie o programa no editor DevC ou Online GDB e cole o código na caixa de resposta *

```
#include <stdio.h>

int main() {
// Declarar um vetor de inteiros com tamanho 5
int numeros[5];

// Solicitar ao utilizador 5 números
for (int i = 0; i < 5; i++) {
printf("Digite o %dº número: ", i + 1);
scanf("%d", &numeros[i]);
}

// Escrever o conteúdo do vetor no ecrã
for (int i = 0; i < 5; i++) {
printf("%d ", numeros[i]);
}

printf("\n");

return 0;
}
```

CC. Segunda Sessão PRIMM – Proposta de Resolução

Etapa 1: Predict

1

Analise o seguinte bloco de código, sem o executar, preveja o que ele faz, e escreva na caixa de resposta qual o output do programa?

```c
#include <stdio.h>
main()
{
    int numeros[5];
    int i;
    for (i=0;i<=4;i++)
     {
      printf("Introduza um numero inteiro para a posicao %d\n",i);
      scanf("%d",&numeros[i]);
     }
    printf("\n\n");

    for (i=0;i<=4;i++)
      printf("%d\n",numeros[i]);
}
```

O programa solicita ao utilizador para introduzir cinco números. De seguida, cada número é colocado pela ordem em que foi inserido no vetor numeros.
Por fim apresenta no ecrã os números pela ordem de introdução no vetor.

Etapa 2: Run

Copie o seguinte bloco de código para ambiente de desenvolvimento **DevC** ou **Online GDB**, compile e execute o programa.

```c
#include <stdio.h>
main()
{
  int numeros[5];
  int i;

  for (i=0;i<=4;i++)
   {
    printf("Introduza um numero inteiro para a posicao %d\n",i);
    scanf("%d",&numeros[i]);
   }
  printf("\n\n");

  for (i=0;i<=4;i++)
    printf("%d\n",numeros[i]);
}
```

2

O output do programa foi o mesmo que previu na etapa 1 - Predict?

- (●) Sim
- ( ) Não

3

Coloque/Cole aqui o output do programa da etapa 2 - Run

```
Introduza um numero inteiro para a posicao 0
5
Introduza um numero inteiro para a posicao 1
2
Introduza um numero inteiro para a posicao 2
3
Introduza um numero inteiro para a posicao 3
4
Introduza um numero inteiro para a posicao 4
2
```

Etapa 3 : Investigate

4

Analise o seguinte bloco de código, indique na caixa de resposta o que faz o bloco de código entre a linha 4 e 10.

```
1  #include <stdio.h>
2  main()
3  {
4     int numeros[5];
5     int i;
6     for (i=0;i<=4;i++)
7      {
8       printf("Introduza um numero inteiro para a posicao %d\n",i);
9       scanf("%d",&numeros[i]);
10     }
11    printf("\n\n");
12
13    for (i=0;i<=4;i++)
14      printf("%d\n",numeros[i]);
15 }
```

Linha 4: declarar vetor numeros com elementos do tipo inteiro
Linha 5: declarar variável i
Linha 6: ciclo for com 5 iterações (desde iteração 0 até 4)
linha 7: chaveta inicia o bloco código do ciclo for
Linha 8: Pede ao utilizador para inserir um numero inteiro apresentado no ecrã a seguinte mensagem "Introduza um numero inteiro para a posição [ ] "
Linha 9: Guarda cada numero inserido pela ordem de inserção, na posição respetiva do vetor.
linha 10: chaveta termina o bloco código do ciclo for

Etapa 4 : Modify

5

Altere o seguinte programa, para que, após utilizador inserir 5 números, os apresente no ecrã, **pela ordem inversa de inserção**, e **todos na mesma linha**, separados pela caracter **Tab**

```
#include <stdio.h>
main()
{
  int numeros[5];
  int i;

  for (i=0;i<=4;i++)
   {
    printf("Introduza um numero inteiro para a posicao %d\n",i);
    scanf("%d",&numeros[i]);
   }
  printf("\n\n");

  for (i=0;i<=4;i++)
    printf("%d\n",numeros[i]);
}
```

```
for (i=4; i >=0 ; i-- ) printf("%t\n",numeros[i]);
```

Etapa 5 - Make

6

Crie um novo programa que solicite 8 notas (0 a 20) ao utilizador, e os armazene num vetor chamado **notas**, e de seguida apresente no ecrã :

- A **Soma** das notas do vetor
- A **Menor** nota do vetor
- A **Maior** nota do vetor
- A **Quantidade** de notas **pares** do vetor
- A **Média** das notas.
- A **Quantidade de notas "20"** do vetor

Observações : O programa **só deverá aceitar notas entre 1 e 20**.
Crie o programa no editor DevC ou Online GDB e cole o código na caixa de resposta

```
#include <stdio.h>

int main() {
// Declaração do vetor de notas
int notas[8];

// Variáveis para armazenar as estatísticas
int soma = 0, menor = 0, maior = 0, pares = 0, notas_20 = 0;
float media = 0.0;

// Leitura das notas
for (int i = 0; i < 8; i++) {
printf("Digite a nota %d (entre 1 e 20): ", i + 1);
scanf("%d", &notas[i]);

// Validação da nota
while (notas[i] < 1 || notas[i] > 20) {
printf("Nota inválida! Digite novamente: ");
scanf("%d", &notas[i]);
}

// Atualização dos Apresentação
soma += notas[i];
if (notas[i] < menor) {
menor = notas[i];
}
if (notas[i] > maior) {
maior = notas[i];
}
if (notas[i] % 2 == 0) {
pares++;
}
if (notas[i] == 20) {
notas_20++;
}
}

// Cálculo da média
media = (float) soma / 8;

// Apresentação dos resultados
printf("\n**Apresentação das notas:**\n");
printf("Soma: %d\n", soma);
printf("Menor nota: %d\n", menor);
printf("Maior nota: %d\n", maior);
printf("Quantidade de notas pares: %d\n", pares);
printf("Média das notas: %.2f\n", media);
printf("Quantidade de notas 20: %d\n", notas_20);

return 0;
}
```