INSTITUTO SUPERIOR DE CIÊNCIAS DA EDUCAÇÃO

ISCED - HUÍLA

Diagnóstico das dificuldades no processo de ensino e aprendizagem na disciplina de programação para os estudantes da 10ª classe do Instituto Politécnico nº 131 do Lubango.

Autor: Eloísa do Rosário Baptista Chiquissa

Lubango

(2026)

INSTITUTO SUPERIOR DE CIÊNCIAS DA EDUCAÇÃO

ISCED – HUÍLA

Diagnóstico das dificuldades no processo de ensino e aprendizagem na disciplina de programação para os estudantes da 10ª classe do Instituto Politécnico nº 131 do Lubango.

Trabalho apresentado para obtenção do Grau de

Licenciatura no Ensino de Informática

Autor: Eloísa do Rosário Baptista Chiquissa

Orientador: Tomás Lucas Selombo, Msc

Lubango

(2026)

Agradecimentos

Em primeiro lugar, agradeço a Deus Pai Todo-Poderoso pela vida, pela saúde e pela força que me permitiram chegar até ao fim deste percurso académico.

Agradeço, de forma muito especial, à minha família, sobretudo à minha mãe e aos meus irmãos, pelo apoio constante, pela paciência e pela confiança que sempre depositaram em mim, mesmo nos momentos mais difíceis desta caminhada.

Deixo também o meu agradecimento aos meus companheiros de percurso, que, de forma directa ou indirecta, contribuíram para a realização deste trabalho, através do incentivo, das partilhas e do apoio nos momentos de maior exigência. Em particular, ao meu companheiro, Abrão Barbosa, pelo apoio incondicional.

Aos meus professores, expresso o meu sincero reconhecimento pelos conhecimentos transmitidos ao longo da minha formação. Em particular, agradeço ao meu orientador, Tomás Selombo, pela orientação, disponibilidade, paciência e acompanhamento prestados durante todo o processo de elaboração deste trabalho.

A todos os que contribuíram, de alguma forma, para que este trabalho fosse possível, o meu muito obrigada.

Dedicatória

Dedico este trabalho, em primeiro lugar, aos estudantes de Informática que, ao longo do seu percurso académico, se depararam com dificuldades na aprendizagem da programação. Que este estudo possa servir como um espelho das dificuldades mais comuns enfrentadas nesta fase inicial da formação, contribuindo para uma melhor compreensão dos desafios que marcam o início do contacto com a programação.

Dedico-o igualmente aos professores da área, com a expectativa de que os resultados aqui apresentados possam apoiar a reflexão pedagógica e incentivar a adopção de estratégias que promovam a superação das dificuldades dos alunos no processo de ensino e aprendizagem da programação.

INSTITUTO SUPERIOR DE CIÊNCIAS DA EDUCAÇÃO

ISCED - HUÍLA

DECLARAÇÃO DE AUTORIA DO TRABALHO DE LICENCIATURA

Tenho consciência que a cópia ou plágio além de poderem gerar responsabilidade civil e disciplinar, bem como a reprovação ou retirada do grau, constituem uma grande violação da ética académica.

Nesta base, eu, Eloísa do Rosário Baptista Chiquissa, estudante finalista do Instituto Superior de Ciências da Educação da Huíla (ISCED-HUILA) do curso de Ensino da Informática do Departamento de Ciências Exactas e Naturais, declaro, por minha honra, ter elaborado este trabalho, só e somente com auxílio da Bibliografia que tive acesso e dos conhecimentos adquiridos durante a minha carreira estudantil e profissional.

Lubango, 28 de Dezembro de 2025

A Autora

_____

Eloísa do Rosário Baptista Chiquissa

Resumo

A aprendizagem da programação no ensino técnico-profissional tem-se revelado particularmente exigente para muitos alunos, sobretudo no momento em que contactam pela primeira vez com a disciplina. Na 10.ª classe, fase em que a programação é formalmente introduzida, observam-se dificuldades recorrentes que tendem a comprometer o rendimento académico e a participação dos alunos no processo de aprendizagem. No contexto do IPOLub, estas dificuldades são frequentemente percebidas pelos docentes, mas nem sempre analisadas de forma sistemática e fundamentada, o que limita a compreensão aprofundada do fenómeno.

Neste sentido, o presente estudo procurou responder à seguinte questão de investigação: quais são as principais dificuldades enfrentadas pelos alunos da 10.ª classe do IPOLub na aprendizagem da programação? O objectivo geral consistiu em investigar essas dificuldades, de modo a compreender como se manifestam e quais os factores que se associam ao seu surgimento.

Metodologicamente, o estudo adoptou uma abordagem mista, de natureza descritiva, desenvolvida no contexto do IPOLub. A investigação envolveu alunos da 10.ª classe e professores da disciplina de programação, tendo a recolha de dados sido realizada através de questionários aplicados aos alunos e de questões abertas dirigidas aos professores. O tratamento dos dados quantitativos baseou-se na utilização de métodos de estatística descritiva e inferencial, recorrendo-se a medidas de tendência central, medidas de dispersão e testes de comparação de médias. Os dados qualitativos foram analisados por meio da técnica de análise de conteúdo, permitindo identificar categorias relacionadas com as dificuldades apontadas.

Os resultados revelam a existência de dificuldades de nível moderado na aprendizagem da programação, associadas a factores de natureza cognitiva, pedagógica e contextual. Entre os principais constrangimentos identificados destacam-se limitações no raciocínio lógico, dificuldades na organização do estudo, na interpretação de erros e na aplicação prática dos conteúdos, bem como a escassez de recursos para o estudo fora da sala de aula. Conclui-se que a aprendizagem da programação no contexto analisado exige uma abordagem pedagógica mais integradora e consciente das dificuldades dos alunos, constituindo este estudo um ponto de partida para investigações futuras de carácter experimental e interventivo.

Palavras-chave: Programação; Dificuldades de aprendizagem; Ensino técnico profissional; Informática Aplicada à Gestão.

Abstract

Learning programming in technical and vocational education has proven to be particularly challenging for many students, especially during their first formal contact with the subject. In the 10th grade, when programming is introduced as part of the curriculum, recurring difficulties are observed, often affecting students' academic performance and engagement in the learning process. In the context of IPOLub, these difficulties are frequently recognised by teachers, yet they are not always systematically analysed, which limits a deeper understanding of the phenomenon.

In this context, the present study sought to answer the following research question: what are the main difficulties faced by 10th grade students at IPOLub in learning programming? The general objective of the study was to investigate these difficulties in order to understand how they manifest and which factors are associated with them.

Methodologically, the study adopted a mixed-methods approach with a descriptive design and was conducted at IPOLub. The research involved 10th grade students and programming teachers, with data collected through questionnaires administered to students and open-ended questions addressed to teachers. Quantitative data were analysed using descriptive and inferential statistical methods, including measures of central tendency, measures of dispersion, and mean comparison tests. Qualitative data were examined through content analysis, allowing for the identification of thematic categories related to the reported difficulties.

The results indicate the presence of moderate difficulties in learning programming, associated with cognitive, pedagogical, and contextual factors. The main constraints identified include limitations in logical reasoning, difficulties in study organisation, challenges in interpreting errors and applying content in practice, as well as limited access to resources outside the classroom. It is concluded that learning programming in the analysed context requires more integrative pedagogical approaches that are attentive to students' difficulties, and that this study provides a foundation for future experimental and intervention-based research.

Keywords: Programming; Learning difficulties; Technical and Vocational Education; Computer Science Applied to Management.

LISTA DE GRÁFICOS

LISTA DE TABELAS

LISTA DE SIGLAS	
IPOLUB	Instituto Politécnico nº 131 do Lubango
CFCP	Computational Thinking Framework for Programming.
IDE	Integrated Development Environment.
SPSS:	Statistical Package for the Social Sciences.
PBL:	Project-Based Learning.
LLM	Large Language Model.
MOOC	Massive Open Online Course.

Sumário

INTRODUÇÃO

INTRODUÇÃO

O ensino das linguagens de programação procura desenvolver, nos alunos, competências que lhes permitam conceber programas e sistemas capazes de responder a problemas reais. Apesar disso, a experiência mostra que muitos estudantes enfrentam sérias dificuldades quando iniciam o estudo da disciplina. Os obstáculos surgem sobretudo na compreensão e, mais ainda, na aplicação de conceitos abstractos, como as estruturas de controlo, fundamentais para a construção de algoritmos operacionais (Gomes, 2008).

Mesmo quando os docentes recorrem a estratégias variadas, parte das dificuldades individuais passa despercebida. Essa falta de diagnóstico acaba por limitar a capacidade de oferecer um acompanhamento adequado. Consequentemente, o rendimento académico baixa e, com ele, a confiança e a motivação dos estudantes que, muitas vezes, se sentem incapazes de progredir na área tecnológica.

O presente trabalho procura analisar as dificuldades mais comuns na aprendizagem de programação entre os alunos do Instituto Politécnico n.º 131 do Lubango (IPOLub). Reúne contributos teóricos já consolidados e procura compreender como essas dificuldades se manifestam na prática, tendo em vista explicar o fraco desempenho que frequentemente se observa na disciplina.

Antecedentes do Tema

A investigação sobre as dificuldades no processo de ensino e aprendizagem da programação tem sido uma constante na literatura científica, dada a sua natureza complexa e o elevado índice de insucesso em disciplinas introdutórias. O presente trabalho, que visa o diagnóstico das dificuldades no processo de ensino e aprendizagem na disciplina de programação para os estudantes da 10ª classe do Instituto Politécnico nº 131 do Lubango, insere-se, assim, numa linha de investigação que busca compreender e mitigar este desafio, alinhando-se a trabalhos precursores que exploraram o tema em diferentes contextos.

A análise de trabalhos académicos anteriores (TCCs, Dissertações e Teses) revela-se fundamental para estabelecer o estado da arte e justificar a relevância

da presente investigação, demonstrando a lacuna de conhecimento que se pretende preencher.

É imperativo reconhecer a importância do diagnóstico como ferramenta essencial para a intervenção pedagógica. O trabalho de Scarpati (2023), intitulado Uma proposta de avaliação diagnóstica multidimensional da aprendizagem de programação na formação profissional, investigou a necessidade de estratégias para conhecer as dificuldades e habilidades dos estudantes em actividades práticas de programação em cursos técnicos. O autor defende que uma avaliação diagnóstica não deve focar-se apenas na compreensão de conceitos, mas também na capacidade de resolução de problemas e na aplicação prática. Este estudo é relevante por sublinhar a importância do diagnóstico como princípio que orienta a nossa pesquisa no contexto do ensino técnico angolano.

Avançando na análise, a dissertação de Nunes (2023), Ensino de Lógica de Programação: dificuldades de aprendizagem na percepção de professores do ensino superior, oferece insights valiosos sobre as dificuldades que se manifestam nos níveis iniciais de aprendizagem. O autor, com experiência em instituições de ensino técnico de nível médio, destaca que a lógica de programação é o principal obstáculo e que a percepção dos professores é crucial para identificar as causas do insucesso. Este trabalho reforça a necessidade de se investigar a percepção dos intervenientes no processo, nomeadamente os estudantes da 10ª classe, que se encontram na fase inicial da aprendizagem.

Não podemos, contudo, ignorar o contexto em que nos inserimos. É aqui que os trabalhos de Victor (2011) e Canhici (2014) se tornam cruciais. O estudo de Victor sobre o uso das TIC no processo de ensino/aprendizagem em Angola e a análise de Canhici sobre as dificuldades de aprendizagem no ISCED-Cabinda estabelecem um quadro de referência sobre a realidade estrutural e pedagógica do nosso país. Estes trabalhos precursores, ao contextualizarem o ensino angolano, justificam a necessidade de um estudo mais focado e actualizado, que se debruce especificamente sobre a disciplina de programação, que é o foco da presente investigação.

Os trabalhos precursores demonstram que o diagnóstico das dificuldades na aprendizagem de programação é um tema de relevância internacional e nacional. Contudo, verifica-se uma lacuna na literatura no que diz respeito a um diagnóstico específico e aprofundado das dificuldades enfrentadas pelos estudantes do ensino técnico de nível médio (10ª classe) em Angola, particularmente no Instituto Politécnico nº 131 do Lubango. Enquanto os estudos citados fornecem modelos de diagnóstico, percepções em contextos técnicos e superiores, e contextualizam o ensino em Angola, nenhum se concentra na intersecção específica do nível de ensino, da disciplina e da instituição em foco. A presente investigação, ao realizar um diagnóstico in loco, propõe-se a preencher esta lacuna, fornecendo dados empíricos que possam contribuir directamente com a elaboração de estratégias pedagógicas para a melhoria do processo de ensino e aprendizagem da programação no IPOLub e, por extensão, em instituições de ensino técnico com realidades semelhantes em Angola.

Justificativa da Investigação

O processo de ensino e aprendizagem, sobretudo em áreas técnicas, tem revelado desafios que afectam tanto os docentes como os alunos. No caso particular da programação, essa dificuldade é amplamente reconhecida na literatura: trata-se de um domínio que exige a compreensão rigorosa de conceitos abstractos e uma capacidade contínua de resolução de problemas. Como observa Hannu-Matti (2005, pp. 14-18), “Programação não é um assunto fácil de ser estudado. Requer entendimento correcto de conceitos abstractos. Muitos alunos têm problemas de aprendizagem devido à natureza do assunto.”

A disciplina, por si só, já é exigente. Porém, no contexto angolano, surgem ainda constrangimentos adicionais relacionados com limitações estruturais: carência de recursos, infra-estruturas inadequadas, falta de materiais pedagógicos e um processo de ensino que nem sempre acompanha as necessidades actuais. Mussaque (2024) destaca a fragilidade destes elementos, que acabam por amplificar as dificuldades dos estudantes.

Apesar disso, Hannu-Matti (2005) chama a atenção para um ponto essencial: muitas vezes, o problema não reside apenas na incompreensão dos alunos, mas na ausência de metodologias capazes de os guiar de forma eficaz através desse

processo. Ou seja, a dificuldade é real, mas pode ser mitigada com abordagens pedagógicas mais ajustadas ao perfil dos aprendentes.

Neste cenário, torna-se indispensável compreender, de forma objectiva, quais são os principais obstáculos enfrentados pelos alunos do curso de Informática do IPOLub no estudo da programação. Só conhecendo estes factores é possível delinear estratégias que melhorem o ensino e permitam aos estudantes desenvolver as competências necessárias para prosseguir na área tecnológica com confiança e sustentabilidade académica.

Desenho Teórico da Investigação

Questão de Investigação

Considerando a natureza descritiva da presente investigação, não foram formuladas hipóteses de investigação. Em alternativa, o estudo orienta-se por uma questão de investigação principal e por um conjunto de subquestões, que permitem estruturar a análise dos dados recolhidos e assegurar a coerência entre os objectivos definidos e os resultados obtidos.

- Quais são as dificuldades frequentes enfrentadas pelos alunos da 10.ª classe do curso de Informática Aplicada à Gestão do IPOLub na aprendizagem da programação?
    - Qual é o nível de dificuldade geral pelos alunos na aprendizagem da programação?
    - Existem diferenças no nível de dificuldade de aprendizagem entre alunos do sexo masculino e feminino?
    - De que forma o contacto prévio com programação se relaciona com o nível de dificuldade pelos alunos?
    - Qual é o perfil motivacional dos alunos face às dificuldades em programação?

Objetivo Geral

- Investigar as principais dificuldades enfrentadas pelos alunos da 10.ª classe no processo de ensino e aprendizagem da programação.

Objetivos Específicos

- Fazer o levantamento dos dados relativos às dificuldades de aprendizagem enfrentadas pelos alunos da 10.ª classe na disciplina de Programação.
- Fundamentar teoricamente as principais dificuldades identificadas, com base na literatura científica recente sobre o ensino e aprendizagem da programação.
- Interpretar os dados recolhidos, de modo a evidenciar as dificuldades mais recorrentes entre os alunos.

Desenho Metodológico da Investigação

Neste ponto, apresenta-se detalhadamente o percurso metodológico adoptado para a concretização desta investigação. A selecção criteriosa dos métodos e técnicas aqui descritos foi fundamental para assegurar o rigor e a validade dos resultados obtidos, permitindo responder de forma objectiva às questões de investigação formulada.

Paradigma e Abordagem da Investigação

A definição do tipo de investigação a realizar constitui um passo essencial para o sucesso de qualquer trabalho científico, pois orienta toda a metodologia e as decisões tomadas ao longo do estudo. Como refere Gil (2002), é possível classificar a investigação segundo três aspectos principais: os seus objectivos (exploratória, descritiva ou explicativa), a abordagem utilizada (qualitativa ou quantitativa) e os procedimentos técnicos adoptados (pesquisa bibliográfica, documental, estudo de caso, experimental, entre outros).

A presente investigação assume uma natureza descritiva, uma vez que o seu principal objectivo é identificar, registar e analisar as dificuldades mais recorrentes enfrentadas pelos alunos da 10.ª classe do curso de Informática Aplicada à Gestão do Instituto Politécnico n.º 131 do Lubango (IPOLUB), na disciplina de Programação. De acordo com Gil (2008), a pesquisa descritiva tem como finalidade principal descrever as características de determinada população ou fenómeno, estabelecendo relações entre variáveis sem as manipular. Este tipo de investigação é particularmente útil quando se pretende proceder ao

levantamento de dados reais directamente com os sujeitos do estudo, proporcionando uma compreensão mais clara da situação observada. Conforme Lakatos e Marconi (2017), a investigação descritiva permite observar, registar, analisar e correlacionar factos ou fenómenos sem interferi-los, baseando-se em dados concretos recolhidos em campo.

Atendendo à complexidade do fenómeno educativo em análise, optou-se por uma abordagem predominantemente quantitativa, de natureza descritiva, complementada por elementos qualitativos. A componente qualitativa assumiu um carácter exploratório e interpretativo, tendo como finalidade aprofundar e contextualizar os resultados quantitativos obtidos, sem pretensão de generalização, contribuindo para uma compreensão mais abrangente do fenómeno estudado, deste modo, a investigação enquadra-se numa metodologia mista. Segundo Creswell e Creswell (2021), os designs de métodos mistos representam a integração da pesquisa qualitativa e quantitativa, permitindo uma compreensão mais completa do problema de investigação do que apenas uma das abordagens isoladamente. Esta opção metodológica justifica-se pela necessidade de, por um lado, quantificar as dificuldades enfrentadas pelos alunos na aprendizagem da programação e, por outro, incorporar uma perspectiva institucional e pedagógica, através do contributo interpretativo dos docentes envolvidos no processo de ensino.

Neste estudo, os dados quantitativos são recolhidos através dos questionários aplicados aos alunos, permitindo medir a frequência e a intensidade das dificuldades sentidas. Paralelamente, os dados qualitativos são obtidos através de um questionário de perguntas abertas direcionado a 2 professores da disciplina. O uso de perguntas abertas é fundamental para permitir que os docentes se expressem livremente, oferecendo mais detalhes e contexto sobre as dificuldades observadas na turma, esta recolha assume um caráter qualitativo ao procurar captar a percepção e a experiência profissional dos docentes sobre o desempenho da turma, servindo como um elemento de triangulação de dados. Como afirmam Leite et al. (2021), a pesquisa de métodos mistos engloba a associação entre procedimentos de recolha e análise que combinam técnicas quantitativas e qualitativas para enriquecer a interpretação dos resultados.

Por fim, os procedimentos técnicos adotados incluíram uma pesquisa bibliográfica, que serviu de base à construção da fundamentação teórica, e uma pesquisa de campo, configurando-se como um levantamento descritivo centrado na realidade concreta desta instituição.

Participantes da Investigação

A presente investigação foi desenvolvida no Instituto Politécnico n.º 131 do Lubango (IPOLUB), uma instituição de ensino técnico-profissional de referência na província da Huíla, em Angola.

A população do estudo é constituída por todos os estudantes do curso de Informática Aplicada à Gestão do IPOLUB, estimando-se um total de aproximadamente 450 alunos e pelo corpo docente da disciplina de programação. Segundo Lakatos e Marconi (2017), a população corresponde ao conjunto completo de elementos que partilham determinadas características específicas.

A amostra seleccionada para este estudo é composta por uma turma da 10.ª classe do referido curso e por 2 professores que lecionam a disciplina de Programação na referida instituição. De acordo com Coelho (2021), a amostra constitui uma ferramenta que permite simplificar a realidade, tornando possível a análise dos dados de forma mais precisa, sem a necessidade de envolver toda a população.

A escolha desta turma justifica-se pela relevância de compreender as principais dificuldades enfrentadas pelos alunos no início do seu percurso na disciplina de Programação, uma vez que é neste momento que ocorre o primeiro contacto com os conceitos de lógica e desenvolvimento de algoritmos. A turma em análise é composta aproximadamente por cerca de 40 alunos, tendo todos sido convidados a participar na investigação. A selecção da amostra foi intencional (não probabilística), pois pretendeu-se trabalhar com uma turma específica que apresentava as características necessárias para o alcance dos objectivos propostos.

No que respeita às questões éticas, foram tomadas todas as precauções para garantir a confidencialidade e o anonimato. Todos os participantes (alunos e

professores) foram devidamente informados sobre os objectivos da investigação, e a sua participação foi totalmente voluntária. Foi-lhes assegurado que os dados recolhidos seriam utilizados unicamente para fins académicos, e que a sua identidade não seria revelada em nenhuma fase do estudo.

Critérios de selecção da amostra

A selecção da amostra obedeceu a critérios previamente definidos, de modo a garantir a pertinência dos participantes em relação aos objectivos do estudo. Consideraram-se como critérios de inclusão os estudantes matriculados no curso de Informática e Gestão do Instituto Politécnico do Lubango (IPOLUB) que se encontravam a frequentar a disciplina de Programação, por se tratar da unidade curricular directamente relacionada com o objecto de investigação. Foram igualmente incluídos apenas os estudantes que já tinham tido contacto com os conteúdos da disciplina, assegurando que os participantes possuíam experiência suficiente para reflectir sobre as dificuldades associadas ao processo de aprendizagem da programação. A participação no estudo foi voluntária, tendo sido considerados apenas os questionários devidamente preenchidos. Como critérios de exclusão, não foram considerados estudantes de outros cursos ou áreas de formação, bem como alunos que ainda não tivessem frequentado a disciplina de Programação. Foram igualmente excluídos questionários incompletos ou que não apresentassem respostas válidas para as variáveis em análise.

Relativamente à amostra dos professores, a seleção obedeceu igualmente a critérios previamente definidos, de modo a assegurar a relevância dos participantes face aos objetivos do estudo. Foram considerados como critérios de inclusão os docentes que lecionam no Instituto Politécnico nº 131 do Lubango (IPOLUB), especificamente no curso de Informática aplica a Gestão, na disciplina de Programação, por estarem diretamente envolvidos no processo de ensino da unidade curricular em análise. Foram ainda incluídos apenas os professores que já tiveram contacto pedagógico com alunos da 10ª classe, garantindo que os participantes possuíam experiência concreta no acompanhamento e na avaliação das dificuldades enfrentadas pelos estudantes no processo de aprendizagem da programação. Como critérios de exclusão, não

foram considerados professores que não lecionam no Instituto Politécnico nº 131 do Lubango, que não ministram a disciplina de Programação ou que nunca tiveram contacto com alunos da 10ª classe do curso de Informática aplicada a Gestão. Foram igualmente excluídos docentes cuja experiência profissional não estivesse relacionada com o ensino da programação, por não apresentarem contributos relevantes para as variáveis em estudo.

A definição destes critérios permitiu delimitar claramente a amostra do estudo, garantindo que os dados recolhidos fossem adequados e consistentes com os objectivos da investigação.

Instrumentos e Procedimentos de Recolha de Dados

Para a recolha de dados, utilizou-se dois instrumentos distintos, alinhados com a abordagem metodológica mista adotada no estudo, de modo a captar tanto dados quantitativos junto dos alunos como dados qualitativos junto dos professores.

Relativamente aos alunos, utilizou-se um questionário impresso e estruturado, composto por perguntas fechadas, como referem Costa, Soares e Florêncio (2022), os questionários com perguntas fechadas são ideais para a recolha de informações objectivas em estudos de natureza descritiva, permitindo maior fiabilidade e economia de tempo. A opção pelo formato impresso deveu-se também a razões práticas e contextuais. De acordo com Lorenzini et al. (2024), a aplicação de inquéritos em papel, quando realizada em ambientes controlados como a sala de aula, favorece a concentração dos respondentes, melhora a taxa de resposta e permite recolher dados mais autênticos e fiáveis.

O questionário foi elaborado com base no Computational Thinking Framework for Programming (CFCP), um referencial teórico que organiza as dificuldades de aprendizagem da programação em seis dimensões principais: sintácticas, semânticas, pragmáticas, metacognitivas, de ferramentas/IDE e factores motiacionais. As perguntas foram formuladas segundo uma escala de Likert de 5 pontos (1–Nunca, 2–Raramente, 3–Às vezes, 4–Frequentemente, 5–Sempre), de modo a medir com maior precisão a frequência ou intensidade das dificuldades enfrentadas pelos alunos. Güler e Ayan (2020) salientam que este tipo de instrumento é particularmente útil na investigação educacional, pois

permite obter dados estruturados, interpretáveis estatisticamente, do mesmo modo, Dombi et al. (2021) reforçam que o uso de escalas com múltiplos pontos possibilita uma análise mais rica e diferenciada das avaliações realizadas pelos estudantes.

No que se refere aos professores, utilizou-se um questionário de perguntas abertas, concebido com o objectivo de recolher percepções aprofundadas dos docentes sobre as dificuldades observadas no processo de ensino e aprendizagem da programação, desta forma, foi possível obter dados descritivos e interpretativos essenciais para a compreensão do fenómeno. Este instrumento foi aplicado de forma on-line, através da plataforma Google Forms, garantindo maior flexibilidade, conveniência e confidencialidade para os participantes. A escolha por plataformas digitais é justificada pela sua eficiência, visto que, segundo Lins (2023), a ferramenta Google Forms é largamente adoptada e facilita significativamente o processo de pesquisa académica, optimizando a organização da recolha de dados.

Deste modo, a utilização combinada de questionários estruturados para os alunos e de questionários abertos para os professores permitiu uma análise mais abrangente e integrada do fenómeno em estudo. Esta estratégia de triangulação reforça a consistência metodológica, pois, de acordo com Santos e Baptista (2023), a triangulação entre métodos constitui um processo complexo de cotejamento entre os instrumentos de produção de dados, com o objectivo de ampliar a compreensão acerca de uma questão de pesquisa ou um fenómeno social. Esta abordagem é vista como uma estratégia para combinar rigor, complexidade e profundidade na investigação (Denzin & Lincoln (2006), citado em Santos & Baptista (2023)).

A aplicação do questionário foi realizada directamente em sala de aula, para os alunos, com a presença da investigadora, de forma a esclarecer eventuais dúvidas e garantir o correcto preenchimento. Esta aplicação colectiva e presencial permitiu alcançar uma taxa de resposta elevada e assegurar a qualidade dos dados obtidos.

Procedimentos de Análise de Dados

Após a recolha dos questionários, procedeu-se à análise dos dados através de uma abordagem de métodos mistos, integrando técnicas quantitativas e qualitativas para uma compreensão mais holística do fenómeno em estudo.

No que concerne à vertente quantitativa, os dados foram submetidos a técnicas de estatística descritiva, conforme sustentam Souza e Kerbauy (2017), a análise estatística descritiva é essencial para compreender de forma clara e organizada os fenómenos em estudo. No contexto educativo, este tipo de análise permite descrever, por exemplo, quais os tópicos de programação mais frequentemente apontados como difíceis, bem como a percentagem de alunos que revelam dificuldades específicas.

Os dados foram codificados e inseridos no software IBM SPSS (Statistical Package for the Social Sciences), amplamente utilizado na análise de dados nas ciências sociais e humanas. A utilização deste software possibilitou calcular frequências, percentagens e médias das respostas a cada uma das questões do questionário, permitindo descrever as dificuldades mais recorrentes em cada uma das seis dimensões do CFCP e traçar um perfil geral das dificuldades sentidas pelos alunos da amostra.

Relativamente à componente qualitativa, procedeu-se à análise de conteúdo das respostas abertas dos professores, com o objectivo de aprofundar a compreensão das percepções docentes acerca das dificuldades enfrentadas pelos alunos na aprendizagem da programação. Considerando a natureza exploratória e complementar desta componente, bem como o reduzido número de participantes, a análise foi realizada de forma manual, seguindo os princípios da análise de conteúdo categorial, Ahmed et al. (2025) justificam esta opção metodológica ao defenderem que a análise qualitativa é um processo indutivo que tem como foco a fidelidade ao universo dos significados, exigindo do investigador uma imersão profunda nos dados para identificar padrões e temas emergentes.

As respostas foram organizadas e analisadas a partir de categorias definidas com base nas questões de investigação e nos objectivos do estudo, permitindo a sistematização das ideias expressas pelos docentes e a identificação dos

principais temas abordados. A análise de conteúdo, segundo Dalla Valle (2025), oferece uma análise sistemática e rigorosa dos dados, permitindo uma compreensão mais profunda dos fenómenos estudados, este procedimento possibilitou uma interpretação estruturada dos discursos, respeitando o carácter descritivo da investigação e evitando inferências para além dos dados recolhidos. Desta forma, garante-se que, mesmo em amostras reduzidas, o rigor na categorização assegure a validade das interpretações, conforme sublinha Sampaio (2021).

A apresentação dos resultados será efectuada através de tabelas e gráficos no caso dos dados quantitativos, e quanto à componente qualitativa, os resultados serão apresentados de forma descritiva e interpretativa, com base na análise das respostas dos professores, de forma a proporcionar uma visualização clara e integrada dos resultados, facilitando a sua interpretação e articulação com as conclusões do estudo.

Construção da variável Grau de Dificuldade Geral

O grau de dificuldade geral em programação foi operacionalizado a partir das respostas dos estudantes a um conjunto de itens avaliados numa escala de Likert de cinco pontos, variando de 1 (“Nunca”) a 5 (“Sempre”). Estes itens incidiram sobre diferentes aspectos do processo de aprendizagem da programação, incluindo dificuldades de compreensão, aplicação prática, depuração e utilização do ambiente de programação.

Numa primeira fase, procedeu-se ao cálculo da média aritmética das respostas a estes itens, obtendo-se, para cada participante, um valor médio representativo da dificuldade global percepcionada na escala original de 1 a 5. Esta média permitiu sintetizar a informação proveniente dos vários indicadores num único índice global, facilitando a análise do nível geral de dificuldade sentido pelos alunos.

Com o objectivo de tornar a interpretação dos resultados mais clara e intuitiva, optou-se por transformar esta média para uma escala padronizada de 0 a 100. Esta opção fundamenta-se no facto de escalas percentuais serem de leitura mais imediata em contextos educativos, permitindo uma compreensão mais directa da intensidade das dificuldades reportadas. Importa salientar que esta

transformação corresponde a uma conversão linear da escala original, não alterando a distribuição relativa dos dados nem o significado estatístico das análises realizadas. Sendo assim, o valor mínimo da escala original (1) corresponde a 0 na nova escala, enquanto o valor máximo (5) corresponde a 100, e os valores intermédios mantêm, assim, uma relação proporcional com a escala original, assegurando a coerência e a comparabilidade dos resultados.

Para efeitos de análise e apresentação dos resultados, os valores obtidos foram organizados em três níveis interpretativos de dificuldade: baixa, moderada e elevada. Considerando a escala de 0 a 100, definiram-se os seguintes intervalos: de 0 a 33,33 é baixa dificuldade, de 33,34 a 66,66 é dificuldade moderada e de 66,67 a 100 é elevada dificuldade. A definição destes intervalos teve como base uma divisão equidistante da escala, permitindo uma leitura clara e consistente dos níveis de dificuldade percepcionados pelos estudantes.

CAPÍTULO I - FUNDAMENTAÇÃO TEÓRICA

1. CAPÍTULO 1- FUNDAMENTAÇÃO TEÓRICA

1.1. Introdução ao Ensino da Programação e a sua Relevância

1.1.1. Contexto global e motivação

Nos últimos anos, tem-se verificado um crescente interesse pela inclusão da disciplina de Ciência da Computação nos ensinos básico e secundário. A programação, enquanto parte integrante da ciência da computação, é reconhecida como uma habilidade crucial. Waite e Sentance (2021) no seu relatório "Teaching programming in schools: A review of approaches and strategies", destacam que a programação é fundamental para o desenvolvimento do pensamento lógico, a resolução de problemas e a compreensão de sistemas complexos. O documento enfatiza a importância de uma pedagogia eficaz para o ensino da programação, reconhecendo que a prática é essencial para a compreensão dos conceitos.

Estudos recentes demonstram que o desenvolvimento do pensamento computacional – uma competência que envolve a capacidade de abstracção, decomposição, construção de algoritmos, depuração e resolução de problemas – tem vindo a ser considerado fundamental para o progresso científico, tecnológico e económico do século XXI. Belmar (2022) afirma que "a maioria das políticas educativas actuais reconhece o pensamento computacional como uma competência-chave para a vida no século XXI, equiparada à leitura e à escrita". Este autor sublinha a importância de preparar os alunos não apenas para consumir tecnologia, mas para a criar, um conceito amplamente discutido no seu trabalho.

Ensinar programação no ensino secundário proporciona benefícios que vão muito além do simples domínio técnico. Através da codificação e do desenvolvimento de pequenos projectos, os alunos exercitam competências como o raciocínio lógico, a resolução de problemas complexos, a criatividade e o trabalho colaborativo. Abesadze e Nozadze (2020) sugerem que "a introdução da programação nas escolas é uma via eficaz para o desenvolvimento de

competências fundamentais do século XXI, tais como criatividade, pensamento crítico e resolução de problemas".

A integração da programação nos currículos escolares tem sido objecto de diversas iniciativas e estudos a nível mundial, que fornecem valiosas lições sobre as suas vantagens e desafios. Por exemplo, a reforma curricular no Reino Unido, que introduziu a disciplina de "Computing" em 2014, focou-se em capacitar os alunos como criadores de tecnologia, e não apenas utilizadores (Waite & Sentance, 2021). De forma similar, a Estónia, com o seu programa "ProgeTiger" lançado em 2012, tem sido um exemplo de sucesso na integração da tecnologia e robótica em todos os níveis de ensino, visando fomentar o interesse pelas áreas STEM (Ciência, Tecnologia, Engenharia e Matemática).

Estas experiências internacionais demonstram que a integração bem-sucedida da programação depende não apenas da definição de um currículo, mas também de um forte investimento na formação de professores e no desenvolvimento de recursos pedagógicos adequados. Segundo Araújo (2025), a formação contínua dos docentes é crucial para que se sintam confiantes e competentes no ensino de conceitos de programação. A adopção de metodologias inovadoras, como a aprendizagem baseada em problemas, tem sido explorada, e segundo Kwon et al. (2021), estas abordagens são eficazes porque têm a capacidade de promover o desenvolvimento de competências essenciais para a programação, como o pensamento computacional, a resolução de problemas, a autonomia e o trabalho em equipe, Segundo Coelho e Guedes (2021), a aprendizagem baseada em problemas (PBL) “promove a habilidade de resolução de problemas”, o que é fundamental para a programação, uma vez que a actividade de programar é, em si, um exercício contínuo de resolução de problemas. Além disso, conforme Silva et al. (2015) demonstram, o uso de linguagens de programação visuais, como o Scratch, tem-se revelado estratégias eficazes para introduzir os conceitos fundamentais de forma interactiva, diminuindo a barreira inicial e motivando os alunos antes de transitarem para linguagens baseadas em texto.

Em Portugal, a Sociedade Portuguesa de Ciências da Educação (SPCE) tem vindo a debater a importância da inclusão do pensamento computacional e da programação nos currículos, alinhando-se com as tendências internacionais. A integração da programação e robótica no processo de ensino-aprendizagem tem

demonstrado potencial para renovar o ensino, promovendo um aprendizado alinhado às exigências contemporâneas e futuras, conforme defendem Santos et al. (2024). Deste modo, a experiência prática de programar não só prepara os alunos para futuras carreiras tecnológicas, mas também os capacita com uma mentalidade estruturada para enfrentar desafios em qualquer área do conhecimento.

Num estudo alargado com professores de diversos níveis de ensino, Nouri et al. (2020) concluíram que "a aprendizagem da programação está directamente associada ao desenvolvimento de competências cognitivas, comunicativas, sociais e criativas entre os alunos" . Os autores sublinham ainda que "o pensamento computacional não se limita ao domínio técnico, mas envolve práticas como a depuração, o design de soluções e a comunicação de ideias de forma estruturada". Assim, o ensino da programação é entendido como uma poderosa ferramenta para o desenvolvimento intelectual e pessoal dos estudantes.

1.1.2. Integração da programação em currículos escolares regionais

Na última década, muitos países avançaram na implementação formal da programação nos seus currículos escolares. Belmar (2022) explica que "em países como o Reino Unido, a programação é parte obrigatória do currículo nacional desde 2014, tendo sido equiparada a outras disciplinas basilares como a matemática". Todavia, uma análise comparativa global revela disparidades acentuadas: enquanto países europeus, asiáticos e norte-americanos têm promovido políticas públicas activas nesse domínio, outras regiões, como grande parte da América Latina e do continente africano, enfrentam ainda muitos desafios para integrar efectivamente a programação no ensino.

O mesmo autor adverte que "os países africanos, em particular, continuam atrasados na implementação da programação nos seus currículos escolares, o que acentua as desigualdades digitais e limita as oportunidades de desenvolvimento dos seus jovens".

Em Angola, a implementação do ensino da programação e da informática de forma geral ainda enfrenta limitações significativas, sobretudo ao nível das infra-estruturas e da formação docente. Cangondo et al. (2022) referem que "os

principais desafios incluem a falta de laboratórios de informática, o acesso instável à electricidade e à internet, e a escassez de equipamentos actualizados nas escolas". Para além disso, os autores sublinham que "a maior parte dos docentes não possui formação específica em metodologias activas de ensino da programação, o que compromete a eficácia das práticas pedagógicas".

A mesma investigação conclui que "a mera introdução da tecnologia nas salas de aula é insuficiente se não for acompanhada por políticas institucionais sólidas e formação contínua dos professores". Este cenário evidencia que, embora haja reconhecimento da importância da programação no currículo, a sua aplicação prática em Angola ainda é limitada. Como afirmam os autores, "é imperativo considerar o acesso à tecnologia e à educação digital como um direito básico, e não como um luxo".

Em suma, a literatura recente reforça que o ensino da programação no ensino secundário é de extrema relevância, por fomentar tanto competências técnicas quanto cognitivas e sociais, indispensáveis numa sociedade digital. Abesadze e Nozadze (2020) afirmam que "programar não é apenas escrever código; é ensinar os jovens a pensar e a criar soluções para problemas reais". No entanto, para países em desenvolvimento, como Angola, a concretização dessa visão educativa exige enfrentar desafios estruturais sérios, nomeadamente ao nível dos recursos materiais, da formação de professores e do apoio institucional. O sucesso de quaisquer reformas curriculares dependerá, portanto, da criação de políticas públicas que valorizem a ciência da computação como componente essencial do percurso educativo dos jovens.

1.2. Teorias da Aprendizagem Aplicadas ao Ensino da Programação

Ensinar programação não é apenas uma questão técnica. É também uma questão pedagógica. Para além da linguagem de código, os professores devem compreender como os alunos aprendem, de que forma constroem conhecimento e o que os motiva a persistir diante da complexidade. Neste sentido, teorias da aprendizagem como o construtivismo, o socioconstrutivismo, conectivismo, a aprendizagem significativa e o construcionismo oferecem contributos valiosos e comprovados para transformar a prática educativa.

Compreender estas abordagens teóricas permite aos educadores ir além da simples instrução de sintaxe e lógica, capacitando-os a criar ambientes de aprendizagem que fomentem o pensamento computacional, a criatividade e a capacidade de resolução de problemas complexos. A aplicação consciente destas teorias no planeamento curricular e nas metodologias de ensino é crucial para formar programadores não apenas tecnicamente proficientes, mas também pensadores críticos e inovadores, capazes de se adaptar aos desafios tecnológicos em constante evolução.

1.2.1. Construtivismo e Socioconstrutivismo

De acordo com a teoria construtivista, o conhecimento não é transmitido, mas construído activamente pelo sujeito, em interacção com o meio. Como explica Papert (1985), “a aprendizagem acontece de forma mais eficaz quando o estudante está envolvido na construção activa de artefactos significativos para ele”. Esta ideia alinha-se com o pensamento de Piaget, que defende que para aprender implica acomodar e assimilar novas informações em estruturas mentais já existentes. No contexto da programação, isto traduz-se na necessidade de propor tarefas desafiantes, como desenvolver pequenos jogos, simuladores ou resolver problemas lógicos, nas quais o aluno experimenta, erra, corrige e aprende de forma autónoma.

Por outro lado, o socioconstrutivismo, de matriz vygotskyana, acrescenta a dimensão social ao processo. Vygotsky (1979) defende que a aprendizagem ocorre primeiro no plano social e depois se interioriza no indivíduo, sendo o outro (colega ou professor) peça fundamental neste processo. É o que se observa, por exemplo, quando dois alunos programam em par (pair programming): um com mais domínio orienta o outro, expandindo a chamada Zona de Desenvolvimento Proximal. Este tipo de mediação é central no ensino de programação, especialmente quando os alunos enfrentam obstáculos técnicos e conceptuais.

1.2.2. Conectivismo

O Conectivismo, proposto por George Siemens e aprofundado por Stephen Downes, surge como uma resposta à forma como a internet e as tecnologias digitais alteraram a nossa relação com a informação. Como refere Rostirola

(2020), o Conectivismo "busca que cada indivíduo, em conexão com o mundo construa e produza conhecimento interativo através do conceito de Rede". Ou seja, a aprendizagem já não acontece apenas dentro de nós, mas sim na rede de conexões que estabelecemos com outras pessoas, com fontes de informação e, crucialmente, com a própria tecnologia.

Imagine o conhecimento não como um edifício que construímos tijolo a tijolo na nossa mente, mas como uma teia de aranha, em que cada fio é uma conexão e cada nó é uma fonte de informação. Aprender, nesta perspectiva, é a habilidade de construir e navegar nesta teia, de identificar os nós mais relevantes e de fortalecer as conexões que nos permitem aceder ao conhecimento quando precisamos dele. A aprendizagem, como salienta Paz et. al (2019), "não se restringe a compartimentar ou armazenar conhecimento, na verdade, vai muito além e significa aplicar o conhecimento na prática real de vivências".

1.2.2.1. Os Pilares do Conectivismo

Para compreendermos a fundo esta teoria, é essencial olharmos para os seus princípios basilares, que, embora originalmente delineados por Siemens em 2004, continuam a ser a base do pensamento conectivista:

A aprendizagem e o conhecimento residem na diversidade de opiniões, querendo dizer que o conhecimento não é monolítico, mas sim um mosaico de perspectivas, é enriquecida pela multiplicidade de vozes e pontos de vista que encontramos na rede. A aprendizagem é um processo de conectar nós, estes "nós" podem ser pessoas, ideias, comunidades, fontes de informação ou mesmo dispositivos não-humanos, é a teia que tecemos entre eles. A aprendizagem pode residir em dispositivos não-humanos, o nosso conhecimento não se limita ao que guardamos na nossa memória, ele pode estar armazenado num computador, numa base de dados, numa comunidade online. Saber onde encontrar a informação é tão importante como sabê-la de cor. A capacidade de saber mais é mais importante do que o que se sabe actualmente, em um mundo em constante mudança, a nossa capacidade de aprender continuamente é a nossa maior mais-valia. A manutenção das conexões é fundamental para a aprendizagem contínua, as nossas redes de aprendizagem precisam de ser nutridas e actualizadas constantemente, para que não se tornem obsoletas. A

capacidade de ver conexões entre áreas, ideias e conceitos é uma competência-chave, os seja, a inovação surge, muitas vezes, da intersecção de diferentes campos do saber, a capacidade de estabelecer pontes entre eles é, por isso, fundamental. A actualidade do conhecimento é o objectivo de todas as actividades de aprendizagem conectivistas, o conhecimento é dinâmico e volátil, o que é verdade hoje, pode não o ser amanhã, por isso, a busca por informação actualizada é uma constante. A tomada de decisão é, em si, um processo de aprendizagem, cada escolha que fazemos, cada caminho que seguimos na nossa busca por conhecimento, é uma oportunidade de aprendizagem.

O Conectivismo não depende, exclusivamente, de uma infra-estrutura tecnológica de ponta. Ele valoriza, acima de tudo, as conexões humanas. Numa Angola onde a oralidade e a comunidade têm um papel central, podemos potenciar a criação de redes de aprendizagem locais, em que os alunos partilham conhecimentos, experiências e recursos, cara a cara. Os mais velhos, os mais experientes, podem tornar-se "nós" fundamentais na rede de aprendizagem dos mais novos, transmitindo não só o conhecimento técnico, mas também a sabedoria e a experiência de vida.

A tecnologia, quando disponível, pode ser um catalisador poderoso. O telemóvel, cada vez mais presente na vida dos angolanos, pode tornar-se uma ferramenta de aprendizagem, permitindo o acesso a informação, a comunicação com outros aprendizes e a participação em comunidades de prática online. O professor, neste contexto, assume um papel de facilitador, de curador de informação, que orienta os alunos na construção das suas redes de aprendizagem, que os ajuda a encontrar os recursos mais relevantes e que os desafia a aplicar os seus conhecimentos na resolução de problemas concretos, que façam sentido para a sua realidade.

1.2.3. Aprendizagem Significativa

David Ausubel, criador da teoria da aprendizagem significativa, afirma categoricamente que “o factor mais importante na aprendizagem é aquilo que o aluno já sabe. Averigue-se isso e ensine-se em função disso” (Farias, 2022). Esta abordagem defende que a aprendizagem é mais eficaz quando o novo conteúdo

se liga directamente aos conhecimentos prévios do estudante, gerando compreensão real, e não mera memorização.

No ensino da programação, aplicar Ausubel significa apresentar os conceitos informáticos (como variáveis, estruturas de repetição, ou condicionais) a partir de exemplos do quotidiano do aluno: usar analogias com decisões do dia-a-dia, linguagem natural e resolução de problemas com sentido para o estudante. Como enfatiza Farias (2022), “a teoria ausubeliana apresenta fundamentos sólidos sobre a construção do conhecimento a partir de estruturas cognitivas pré-existentes”. Zanetti, Borges e Ricarte (2022) analisaram 31 artigos publicados entre 2011 e 2021 e constataram que a aplicação da teoria de Ausubel no ensino de programação ainda é tímida, mas muito promissora. Os autores concluem que “existe um reconhecimento crescente do potencial da aprendizagem significativa para tornar o ensino de programação mais acessível, motivador e eficaz”.

1.2.4. Construcionismo

O construcionismo, desenvolvido por Seymour Papert, é uma extensão prática do construtivismo. Mais do que aprender por absorção, defende-se que o aluno aprende construindo algo que tenha significado pessoal, seja um jogo, uma animação, um robô ou um pequeno programa. Papert (1985) explica: “o construcionismo reconhece que a aprendizagem acontece de forma mais eficaz quando as pessoas estão activamente envolvidas na construção de um produto externo”.

Autores como Massa, Oliveira e Santos (2022) destacam que “o uso de computadores na educação, defendido por Papert, está directamente associado à ideia de que as crianças aprendem melhor quando estão envolvidas em actividades criativas e significativas”. Os ambientes de programação visual, como Scratch ou Arduino, são exemplos disso, o aluno não está a decorar comandos, mas a criar com eles.

O estudo de Downey et al. (2022), ao investigarem espaços pedagógicos baseados no construcionismo em cursos de engenharia, afirmam que “o envolvimento prático e criativo dos estudantes na construção de soluções leva à internalização mais profunda dos conceitos técnicos”. Isto valida a ideia de que

a aprendizagem acontece melhor quando o aluno participa activamente e sente que está a construir algo com utilidade.

As teorias de aprendizagem abordadas – Construtivismo, Socioconstrutivismo, Conectivismo, Aprendizagem Significativa e Construcionismo – embora com focos distintos, são altamente complementares no contexto do ensino-aprendizagem de programação. Todas elas convergem na ideia de que a aprendizagem é um processo activo e que o aluno deve ser o protagonista. O Construtivismo estabelece a base para a construção individual do conhecimento através da interacção com o ambiente de programação. O Socioconstrutivismo adiciona a dimensão crucial da colaboração e da mediação social, essencial para o desenvolvimento de projectos em equipa e para a superação de desafios. O Conectivismo, por sua vez, sublinha a importância de saber navegar e manter as conexões na vasta rede de informação e conhecimento, uma competência vital na era digital para o programador moderno. A Aprendizagem Significativa garante que os novos conceitos de programação se ancoram nos conhecimentos prévios dos alunos, tornando a aprendizagem mais profunda e menos memorística. Finalmente, o Construcionismo oferece a metodologia prática para aplicar estas ideias, incentivando a criação de artefactos significativos como o motor da aprendizagem.

Esta integração de perspectivas teóricas visa capacitar os alunos não só com competências técnicas de programação, mas também com a capacidade de pensar criticamente, resolver problemas de forma criativa e colaborar eficazmente, preparando-os para os desafios do século XXI.

Em Angola, o ensino da programação ainda enfrenta inúmeros desafios estruturais: escassez de laboratórios, falhas de energia, acesso limitado à internet e formação insuficiente de professores. Cangondo et al. (2022) são claros: “as instituições em Angola ainda enfrentam desafios como falta de laboratórios para aulas práticas, electricidade, internet e computadores para os estudantes”.

A nível pedagógico, predomina ainda uma abordagem tradicional, baseada na memorização. Contudo, os mesmos autores defendem que “uma formação adequada dos professores e o desenvolvimento de currículos com base em

competências podem transformar este cenário". A aplicação das teorias da aprendizagem, como o construtivismo, o construcionismo, o conectivismo e a aprendizagem significativa, pode, nesse sentido, servir como base sólida para reestruturar o ensino da programação, mesmo em contextos com poucos recursos. É possível, por exemplo, aplicar a aprendizagem significativa utilizando exemplos do contexto local, como resolução de problemas práticos ligados à agricultura, comércio ou gestão comunitária. Também é viável, com criatividade, trabalhar construcionismo com materiais de baixo custo ou simulações digitais em ambiente offline. O construtivismo e o construcionismo recordam-nos que os alunos aprendem melhor quando experimentam, erram e constroem. O Conectivismo lembra-nos que, na era da informação, a capacidade de localizar, avaliar e conectar-se a fontes de conhecimento é tão crucial quanto o conhecimento em si. O socioconstrutivismo reforça o poder do outro (do colega, do professor, do grupo) no processo de construção do saber. E Ausubel mostra-nos que, sem ligação com o que o aluno já conhece, o conteúdo simplesmente não se fixa.

Mesmo em Angola, onde os desafios são reais e urgentes, é possível aplicar essas teorias com criatividade, bom senso e compromisso. Como defende Papert (1985), "quando os estudantes falam a linguagem do código para criar algo que os entusiasma, estão a aprender muito mais do que apenas programação". Estão a aprender a pensar, a resolver problemas, a trabalhar em equipa, competências fundamentais para o século XXI.

1.3. Dificuldades comuns no ensino e aprendizagem de programação

A aprendizagem de programação constitui uma tarefa de notável complexidade que exige dos estudantes um conjunto diversificado de competências que transcendem a mera memorização de sintaxes. Diversos estudos têm investigado as dificuldades enfrentadas por alunos em disciplinas introdutórias de programação, procurando identificar as causas subjacentes a estes desafios. A compreensão destas causas revela-se fundamental para o desenvolvimento de estratégias pedagógicas mais eficazes e para a redução das taxas de reprovação e evasão em cursos da área tecnológica.

Um estudo realizado por Silva e Moreira (2021) com estudantes do curso de Análise e Desenvolvimento de Sistemas do IFRN/Pau dos Ferros identificou algumas das principais dificuldades e suas causas. Os investigadores destacam que: "entre as dificuldades citadas pelos estudantes, destacam-se a dificuldade em colocar em prática o que estudam na parte teórica, a abstracção dos problemas e o raciocínio lógico" (p. 8). Esta constatação revela-se particularmente significativa, pois aponta para uma desconexão entre o conhecimento teórico adquirido e a sua aplicação prática. Os autores observam ainda que estas dificuldades podem conduzir à desmotivação, reprovação e, em casos mais extremos, ao abandono do curso.

No que concerne ao tempo de dedicação aos estudos, Silva e Moreira (2021) descobriram dados preocupantes: "Quando questionados sobre o tempo de dedicação aos estudos de programação, 32% dos estudantes afirmaram dedicar menos de 1 hora diária, 28% dedicam entre 1 a 2 horas, 16% dedicam entre 2 a 3 horas e apenas 20% dedicam mais de 3 horas diárias" (p. 4). Esta distribuição temporal sugere que uma parcela significativa dos estudantes não dedica tempo suficiente para uma disciplina que exige prática constante e reflexão contínua.

Relativamente aos métodos de resolução de problemas, os investigadores identificaram práticas que comprometem o desenvolvimento da autonomia intelectual: "Quando os estudantes encontram dificuldades na resolução de problemas, 21 estudantes responderam que pesquisam por soluções na internet e/ou livros, 20 pedem ajuda para outros colegas, 16 pedem ajuda para professores e/ou monitores, e 2 abandonam o problema" (p. 5). Esta tendência para procurar soluções externas, em detrimento do exercício do próprio raciocínio, constitui um obstáculo ao desenvolvimento das competências fundamentais de programação.

Um estudo mais recente, conduzido por Ngadengon et al. (2025) com estudantes de politécnicos na Malásia, corrobora e expande as causas das dificuldades no aprendizado de programação. Este estudo procurou analisar a compreensão dos alunos sobre Resolução de Problemas e Design de Programas (PSPD), os factores que contribuem para o baixo desempenho e a influência dos ambientes de aprendizagem.

Os investigadores identificaram que as maiores dificuldades dos estudantes residem em:" Os resultados indicam que o valor médio mais elevado, 3.34 (DP = 0.916), mostra que os estudantes têm mais dificuldade em conceber programas para resolver tarefas específicas. Isso sugere que eles enfrentam dificuldades na criação de algoritmos, no reconhecimento de problemas, na formulação de soluções e na implementação das suas ideias por meio da codificação." (p. 1056).

Esta descoberta é particularmente reveladora, pois aponta para uma lacuna fundamental na capacidade de transformar um problema conceptual numa solução programável. Os autores observam ainda que: “Além disso, os estudantes enfrentam dificuldades com a depuração (debugging), que apresenta um valor médio de 3.31 (DP = 0.960), e com a compreensão da sintaxe, cujo valor médio é de 3.30 (DP = 0.864). " (p. 1056).

No que se refere aos factores curriculares, os autores identificaram problemas estruturais significativos: "Os dois factores com classificação mais elevada, com valores médios de 3.32 e 3.17, são: a ênfase do plano curricular na teoria em detrimento da aplicação prática e o número excessivo de tópicos abordados em cada semestre, sendo ambos classificados como de "média-alta" importância " (p. 1057). Esta constatação sublinha a importância de equilibrar adequadamente os componentes teóricos e práticos do currículo, bem como de gerir cuidadosamente a densidade do conteúdo programático.

1.3.1. Factores Cognitivos: A Complexidade do Pensamento Abstracto

As dificuldades no aprendizado de programação encontram-se frequentemente enraizadas em desafios cognitivos profundos. A programação exige um nível elevado de pensamento abstracto e raciocínio lógico, competências que nem todos os estudantes desenvolvem no mesmo ritmo ou com a mesma facilidade. Esta transição do pensamento concreto para o abstracto constitui um obstáculo significativo para muitos principiantes.

Como observam Ngadengon et al. (2025), a compreensão das estruturas de controlo representa um desafio particular: "As estruturas de controlo na resolução de problemas, que envolvem sequência, selecção e repetição, apresentaram um valor médio de 2.89 (DP = 0.706). Isso implica que as

estruturas de controlo — que incluem os tópicos de sequência, selecção e repetição — são complexas para os estudantes aprender" (p. 1056).

A depuração de erros (debugging) constitui outra área de dificuldade significativa. Esta tarefa exige não apenas conhecimento técnico, mas também uma capacidade analítica apurada para identificar a origem de falhas no código. A compreensão da sintaxe de uma linguagem de programação, embora possa parecer uma questão de memorização, envolve na realidade a internalização de regras e estruturas que formam a base para a construção de lógica complexa.

1.3.2. Factores Pedagógicos e Curriculares: O Impacto das Metodologias de Ensino

A forma como a programação é ensinada e a estrutura dos currículos desempenham um papel crucial nas dificuldades de aprendizado. Uma ênfase excessiva na teoria em detrimento da prática constitui um problema recorrente. Muitos cursos concentram-se na apresentação de conceitos teóricos sem oferecer oportunidades suficientes para que os alunos apliquem esse conhecimento em cenários práticos e reais.

Ngadengon et al. (2025) descobriram que os estudantes preferem claramente abordagens mais práticas:" As discussões interactivas com os docentes obtiveram o valor médio mais elevado, 4.23 (DP = 0.805), indicando um nível "Alto" de eficácia na facilitação da aprendizagem. O trabalho prático em sessões de laboratório surge logo a seguir, com uma média de 4.19 (DP = 0.800), também classificado como de eficácia "Alta" " (p. 1058). Esta preferência por metodologias interactivas e práticas contrasta com a realidade de muitos currículos que privilegiam abordagens expositivas tradicionais.

O volume de conteúdo abordado num único semestre também se revela um factor limitante. Currículos sobrecarregados com muitos tópicos impedem que os estudantes aprofundem o seu entendimento em conceitos fundamentais, conduzindo a uma compreensão superficial e à dificuldade em conectar diferentes temas.

1.3.3. Factores Motivacionais e Experienciais: O Papel da Motivação e da Experiência Prévia

A motivação do estudante constitui um componente vital no processo de aprendizagem. A baixa motivação pode ser simultaneamente uma causa e uma consequência das dificuldades em programação. A percepção de que a disciplina é excessivamente difícil, a frustração com a depuração de erros ou a falta de progresso podem conduzir à desmotivação, que por sua vez impacta negativamente o envolvimento e a persistência do aluno.

Ngadengon et al. (2025) identificaram a falta de experiência prévia como um factor significativo:" Relativamente à experiência em programação na secção B, 89 estudantes (37.7%) declararam ter experiência em programação. Por outro lado, 147 estudantes (62.3%) afirmaram não ter qualquer experiência em programação " (p. 1055). Esta disparidade na experiência prévia cria desafios pedagógicos consideráveis, pois os estudantes sem qualquer contacto anterior com conceitos de lógica de programação ou pensamento algorítmico necessitam de um apoio adicional para construir essa base fundamental.

As investigações de Silva e Moreira (2021) e dos autores citados à cima convergem em pontos cruciais, nomeadamente a dificuldade na aplicação prática da teoria, a importância do raciocínio lógico e da abstracção, e a necessidade de metodologias de ensino mais eficazes e centradas no aluno. A identificação destas causas constitui o primeiro passo para o desenvolvimento de intervenções pedagógicas que possam mitigar estes desafios e promover uma aprendizagem mais efectiva em programação.

Como observam Ngadengon et al. (2025) nas suas conclusões: "Com base nesses resultados, podem ser feitas várias sugestões para melhorar o ensino de programação. Em primeiro lugar, os docentes devem focar-se em métodos práticos no processo de ensino, garantindo que os estudantes consigam aplicar os conceitos teóricos em cenários do mundo real" (p. 1060). Esta recomendação sublinha a necessidade urgente de repensar as abordagens pedagógicas tradicionais, privilegiando metodologias que integrem teoria e prática de forma equilibrada e que considerem as necessidades individuais dos estudantes.

1.3.4. O Contexto Angolano: Desafios e Implicações para o Aprendizado de Programação

Ao debruçarmo-nos sobre as dificuldades no aprendizado de programação, é imperativo considerar o contexto específico de cada região. Em Angola, o cenário educacional e tecnológico apresenta desafios particulares que, embora não directamente focados na programação, podem ter um impacto significativo na forma como os estudantes adquirem estas competências cruciais. A ausência de estudos empíricos específicos sobre as dificuldades no aprendizado de programação em Angola torna necessária uma abordagem inferencial, baseada nos desafios mais amplos que o país enfrenta na educação e no desenvolvimento tecnológico.

Um estudo fundamental para a compreensão das dificuldades no aprendizado de programação no contexto angolano é o de Cangondo et al. (2022), intitulado "Computer Science Education in Angola: The Key Challenges". Este artigo, publicado nas atas da 2022 IEEE Global Engineering Education Conference (EDUCON), investigou a situação do ensino de informática em Angola através de um questionário aplicado a professores de escolas secundárias, colégios e universidades do país.

Os resultados deste estudo corroboram e aprofundam as inferências feitas anteriormente sobre as limitações infra-estruturais e pedagógicas. Os autores identificaram que, apesar da revolução digital global, as instituições angolanas ainda enfrentam "vários desafios, tais como, falta de laboratórios para aulas práticas, falta de electricidade, falta de internet, e falta de computadores para estudantes". Estas carências básicas criam um ambiente desfavorável para o ensino e aprendizagem de programação, que exige acesso constante a recursos tecnológicos e práticos.

Além das limitações de infra-estrutura, o estudo aponta para a ineficácia da adopção de tecnologias em sala de aula, sugerindo que esta pode ser devida à "ausência de factores determinantes para a inclusão tecnológica pelas instituições, ou mesmo por causa da formação de professores na área, seja fraca, ineficiente ou incompleta". Esta observação é crucial, pois destaca que a mera disponibilidade de tecnologia não garante a sua utilização eficaz; é

necessário um planeamento institucional adequado e, acima de tudo, professores bem preparados para integrar essas ferramentas no processo de ensino-aprendizagem.

Os autores concluem que, embora o uso de software em sala de aula facilite a aquisição de conhecimento, "ainda existem várias barreiras a serem superadas para que isso aconteça amplamente". Esta análise reforça a ideia de que as dificuldades no aprendizado de programação em Angola não são apenas cognitivas ou motivacionais, mas também sistémicas, enraizadas em desafios estruturais e na formação de recursos humanos.

1.4. Impactos das Dificuldades no Desempenho Académico e Profissional

A incapacidade de superar as barreiras iniciais na programação pode desencadear um ciclo vicioso de desmotivação, baixo rendimento e, em última análise, o abandono da área, com implicações directas para o desenvolvimento tecnológico e a qualificação de mão-de-obra, especialmente em contextos como o de Angola.

As dificuldades na compreensão de conceitos de programação têm um impacto directo e mensurável no desempenho académico dos estudantes. A literatura recente sublinha que a incapacidade de assimilar os fundamentos da programação pode levar a resultados académicos insatisfatórios, manifestados em notas baixas em trabalhos e exames, e, em casos mais graves, a taxas elevadas de reprovação e abandono dos cursos. Ngadengon et al. (2025) realçam que "problemas na compreensão de conceitos de programação podem ter um impacto no desempenho académico geral dos estudantes, resultando em baixo desempenho em trabalhos e exames".

Um dos principais desafios reside na transição do conhecimento teórico para a aplicação prática. Silva e Moreira (2021), num estudo com estudantes brasileiros, identificaram que "entre as dificuldades citadas pelos estudantes, destacam-se a dificuldade em colocar em prática o que estudam na parte teórica, a abstracção dos problemas e o raciocínio lógico". Esta desconexão entre a teoria e a prática é um factor crítico que compromete a capacidade do aluno de

resolver problemas reais e de desenvolver projectos funcionais. A falta de raciocínio lógico e a dificuldade em abstrair problemas complexos em componentes programáveis são barreiras que, se não forem superadas, impedem o progresso do estudante na disciplina.

Além disso, a dedicação insuficiente ao estudo é um factor agravante. O mesmo estudo de Silva e Moreira (2021)revelou que uma percentagem significativa de estudantes dedica menos de uma hora diária ao estudo da programação, uma disciplina que exige prática constante e reflexão contínua. Esta falta de empenho, aliada às dificuldades intrínsecas da matéria, contribui para o insucesso académico. A tendência para procurar soluções prontas na internet ou pedir ajuda a colegas, em vez de exercitar o próprio raciocínio, também limita o desenvolvimento da autonomia intelectual e das competências essenciais de programação.

As consequências do baixo desempenho académico são vastas. Para além da reprovação, a desmotivação é um efeito colateral comum. Robins et al. (2003), citados por Ngadengon et al. (2025), já alertavam que "os desafios iniciais na programação resultam em estudantes a desistir e a não ter uma auto-imagem forte, o que é crítico para a sua prossecução de futuros na área". A percepção de incapacidade pode levar os estudantes a questionar a sua aptidão para a área tecnológica, culminando no abandono dos estudos ou na mudança para outras áreas, o que representa uma perda de talento para o sector.

As dificuldades persistentes na aprendizagem da programação não afectam apenas o percurso académico, mas também têm um impacto profundo e duradouro na trajectória profissional dos indivíduos. Num mercado de trabalho cada vez mais digitalizado, a proficiência em programação é uma competência altamente valorizada e, em muitos sectores, indispensável. A incapacidade de dominar esta área pode limitar significativamente as oportunidades de emprego e o desenvolvimento de uma carreira bem-sucedida.

Ngadengon et al. (2025) sublinham que a proficiência em programação é um requisito essencial para estudantes que procuram disciplinas nas áreas de Tecnologia da Informação e Ciência da Computação. A falta de uma base sólida em programação pode levar a dificuldades na entrada no mercado de trabalho,

uma vez que as empresas procuram profissionais com capacidade de resolver problemas complexos e de desenvolver soluções eficazes. A dificuldade em conceber programas para tarefas específicas, em criar algoritmos e em depurar código, conforme identificado por Ngadengon et al. (2025), são lacunas que se traduzem directamente em menor empregabilidade e progressão na carreira.

No contexto angolano, estes desafios são ainda mais acentuados devido a problemas estruturais no ensino e à escassez de recursos. O documento "Cenários do Futuro de Angola 2050" e outros estudos sobre a educação em Angola apontam para a falta de infra-estruturas adequadas, acesso limitado à electricidade e à internet, e a carência de formação docente específica em metodologias activas, estes factores contribuem para que os estudantes angolanos enfrentem barreiras adicionais na aquisição de competências de programação, o que os coloca em desvantagem no mercado de trabalho global e, por vezes, até no mercado local.

Apesar do crescimento do mercado de trabalho em Angola, muitos dos novos empregos são de baixa qualidade e contribuem modestamente para o crescimento económico (Bank). A falta de mão-de-obra qualificada em áreas tecnológicas, como a programação, é um desafio fundamental para o desenvolvimento do país (INQ, 2021). Assim, as dificuldades na aprendizagem da programação não só prejudicam o indivíduo, mas também limitam o potencial de inovação e crescimento económico de Angola, ao não conseguir formar profissionais aptos a responder às exigências de um mundo cada vez mais dependente da tecnologia.

Em suma, as dificuldades na aprendizagem da programação têm um efeito cascata, que se inicia no desempenho académico e se estende ao sucesso profissional. A superação destes desafios exige não só um maior empenho individual, mas também a implementação de políticas educativas robustas e o investimento em infra-estruturas e formação de professores, especialmente em países em desenvolvimento como Angola, para garantir que os jovens estejam preparados para as exigências do século XXI.

1.5. Estratégias Comprovadas para Superação das Dificuldades na Aprendizagem de Programação

A aprendizagem de programação, embora crucial na era digital, apresenta desafios significativos que podem levar à frustração e ao desânimo. Contudo, diversas estratégias pedagógicas e de auto-regulação têm-se mostrado eficazes na mitigação destas dificuldades, promovendo um percurso de aprendizagem mais robusto e bem-sucedido. A aplicação destas estratégias é fundamental tanto a nível global como, em particular, no contexto angolano, onde o acesso a recursos e metodologias inovadoras pode ser um factor diferenciador.

As metodologias de ensino tradicionais, muitas vezes focadas na transmissão passiva de conhecimento, revelam-se insuficientes para as complexidades da programação. Em contraste, abordagens pedagógicas inovadoras promovem um ambiente de aprendizagem mais dinâmico e eficaz, Segundo Vidal-Silva et al. (2025), a educação em programação vai além da mera transmissão de conhecimento; ela fomenta uma mentalidade computacional, aprimora o raciocínio lógico e equipa os estudantes com estratégias de resolução de problemas.

Entre as abordagens mais eficazes, destacam-se a aprendizagem baseada em projectos (Project-Based Learning - PBL) e a programação em pares (pair programming). Estas metodologias incentivam os alunos a aplicar conceitos teóricos em cenários práticos, o que solidifica a compreensão e desenvolve competências essenciais para o mercado de trabalho. Segundo Maia et al. (2025), a gamificação, outra metodologia activa, tem demonstrado ser eficaz no ensino de programação, melhorando a motivação e o engajamento dos alunos.

No contexto angolano, a implementação de tais metodologias pode ser um desafio devido às limitações de infra-estrutura e formação de professores. No entanto, a adaptação e a promoção de projectos práticos, mesmo com recursos limitados, podem estimular a criatividade e a capacidade de resolução de problemas dos estudantes. A colaboração entre instituições de ensino e empresas locais pode facilitar a criação de projectos relevantes e a disponibilização de mentores, enriquecendo a experiência de aprendizagem. Segundo Blaszko et al. (2021), a formação contínua de professores

universitários em metodologias activas é fundamental para a transformação da prática pedagógica e para a promoção de uma aprendizagem significativa.

Para além das abordagens pedagógicas, as estratégias de aprendizagem autorregulada desempenham um papel vital na superação das dificuldades em programação. Segundo Ramírez-Echeverry et al. (2025), as estratégias de aprendizagem autorregulada empregadas por estudantes em cursos de programação revelam uma variedade de estratégias eficazes, incluindo a resolução de problemas, a aquisição de conhecimento e a gestão do ambiente de estudo. A metacognição, ou seja, a capacidade de monitorizar e regular o próprio processo de aprendizagem, é destacada como um factor crucial, em conjunto com a prática, a resolução de problemas e a gestão do tempo. A pesquisa também aponta para a importância de um suporte aprimorado na estruturação e documentação dos processos de pensamento e código, sugerindo que os educadores devem desenvolver intervenções direccionadas para melhorar as habilidades de auto-regulação dos estudantes.

Em Angola, onde os recursos educacionais podem ser escassos, a promoção da auto-regulação torna-se ainda mais pertinente. Incentivar os estudantes a desenvolverem a sua autonomia na aprendizagem, a gerirem o seu tempo de estudo e a procurarem activamente soluções para os seus problemas pode compensar algumas das lacunas existentes. A criação de grupos de estudo e a promoção de plataformas de partilha de conhecimento, mesmo que informais, podem fortalecer estas competências.

1.5.1. Ferramentas e Recursos de Apoio

A utilização de ferramentas e recursos de apoio é uma estratégia comprovada para facilitar a aprendizagem de programação, tornando-a mais acessível e eficaz. Ferramentas interactivas, plataformas de e-learning e ambientes de desenvolvimento integrados (IDEs) com funcionalidades de depuração e feedback em tempo real podem melhorar significativamente a experiência de aprendizagem. A literatura actual enfatiza a importância de ambientes de aprendizagem que forneçam feedback imediato e oportunidades para experimentação prática, elementos cruciais para a consolidação do conhecimento em programação.

Modelos de Linguagem Grandes (LLMs), como o ChatGPT, emergem como ferramentas de apoio promissoras na educação em programação. Yan et al. (2025) investigaram o impacto da programação colaborativa baseada em LLMs no pensamento computacional e na auto-eficácia dos estudantes. Os resultados sugerem que a integração de LLMs pode reduzir significativamente a carga cognitiva dos estudantes e melhorar as suas habilidades de pensamento computacional. Embora o estudo se foque no ensino fundamental e médio, as implicações para o ensino superior e profissional são claras: LLMs podem actuar como assistentes inteligentes, fornecendo feedback instantâneo, sugestões de código e explicações, superando limitações de tempo e localização e atrasos no feedback em modelos colaborativos convencionais. Esta acessibilidade e adaptabilidade tornam os LLMs particularmente relevantes em contextos onde o acesso a instrutores especializados é limitado.

Além disso, plataformas de e-learning de código aberto (Open Source Learning Management Systems - LMS) oferecem soluções flexíveis e personalizáveis para a gestão e entrega de conteúdo educacional. Koblyakov (2025) destaca que plataformas como Open edX e Moodle são altamente escaláveis e permitem uma personalização superior, eliminando custos de licenciamento elevados. Estas plataformas suportam cursos online massivos (MOOCs), aprendizagem mista e cursos auto-ritmados, com ferramentas avançadas de autoria e análises integradas para monitorizar o envolvimento e o desempenho dos alunos. A natureza de código aberto garante actualizações constantes e uma comunidade global de desenvolvedores que contribuem para a sua inovação e segurança. A implementação de tais plataformas pode democratizar o acesso à educação em programação, permitindo que instituições em Angola adaptem os recursos às suas necessidades específicas e promovam a colaboração entre estudantes e educadores.

No contexto angolano, a expansão do acesso à internet e a dispositivos computacionais é crucial para que os estudantes possam beneficiar plenamente destas ferramentas. Iniciativas que promovam a criação de laboratórios de informática bem equipados e o acesso a plataformas de e-learning, mesmo que offline, podem fazer uma diferença substancial. A integração de ferramentas como o Arduino, conforme revisado por Marín-Marín et al. (2024), pode também

ser uma estratégia valiosa para o desenvolvimento do pensamento computacional e habilidades de programação através de projectos práticos e interactivos, mesmo em ambientes com recursos limitados.

CAPÍTULO II - APRESENTAÇÃO E ANÁLISE DOS DADOS

2. CAPÍTULO 2- APRESENTAÇÃO E ANÁLISE DOS DADOS

2.1. Caracterização do Curso de Informática e Gestão do IPOLub

O curso de Informática Aplicada à Gestão ministrado no Instituto Politécnico do Lubango (IPOLub) integra-se no subsistema de ensino técnico-profissional e tem a duração de três anos, correspondentes à 10.ª, 11.ª e 12.ª classes, com uma carga horária global aproximada de 4.080 horas. O curso tem como objectivo fundamental a formação técnica e profissional de jovens em idade escolar, preparando-os para o exercício de uma profissão, de modo a responder às necessidades do país e à constante evolução tecnológica (RETEP, 2011).

De acordo com a grelha curricular em anexo (Tabela 24), o plano de estudos encontra-se estruturado em três grandes componentes: sociocultural, científica e técnica, tecnológica e prática. A componente sociocultural, predominante nos dois primeiros anos, abrange disciplinas como Português, Língua Estrangeira, Formação de Atitudes Integradoras e Educação Física, visando o desenvolvimento de competências comunicativas, sociais e comportamentais essenciais para a inserção profissional. A componente científica contempla disciplinas fundamentais como Matemática, Organização e Administração de Empresas e Direito, desempenhando um papel estruturante no desenvolvimento do raciocínio lógico, da capacidade analítica e da compreensão dos processos organizacionais. A Matemática, em particular, assume uma presença constante ao longo do curso, o que evidencia a sua relevância como base para a compreensão de conteúdos técnicos, nomeadamente na área da programação. A componente técnica, tecnológica e prática constitui o eixo central do curso, integrando disciplinas directamente relacionadas com a informática, tais como Técnica e Linguagem de Programação, Tecnologias de Informação e Comunicação, Bases de Dados e Redes de Computadores, Sistemas de Informação e Comunicação, Informática Aplicada à Gestão e Projectos Tecnológicos. Esta componente apresenta uma carga horária significativa, reflectindo a natureza prática e aplicada do curso, bem como a exigência de competências técnicas específicas por parte dos alunos.

No que respeita ao estado actual do plano curricular, este prevê a articulação entre saberes teóricos de base, como Matemática, Língua Portuguesa e Língua Estrangeira, e saberes práticos e relacionais, incluindo o trabalho em equipa, a responsabilidade, a flexibilidade e a segurança no contexto laboral (RETEP, 2011).

Esta organização curricular demonstra uma intenção clara de formar técnicos capazes de actuar tanto ao nível técnico como ao nível organizacional e social. Relativamente ao perfil de saída, o curso visa formar técnicos aptos a montar e manter sistemas informáticos, programar e gerir aplicações informáticas, instalar software e compreender a estrutura e os processos de uma empresa. Entre as competências esperadas destacam-se a capacidade de programar em diferentes linguagens, desenvolver aplicações integráveis na internet, operar sistemas informáticos em rede, automatizar sistemas de informação e elaborar documentação técnica sobre o software desenvolvido.

Não obstante a coerência e abrangência do plano curricular, a elevada carga horária dedicada à programação e às disciplinas técnicas exige dos alunos competências sólidas ao nível do raciocínio lógico, da abstração, da resolução de problemas e da autonomia no estudo. Neste contexto, a disciplina de Programação, introduzida já na 10.ª classe, representa um desafio significativo para muitos alunos, sobretudo aqueles que ingressam no curso sem contacto prévio com conteúdos informáticos ou com bases matemáticas consolidadas.

Deste modo, a caracterização do curso de Informática Aplicada à Gestão no IPOLub permite compreender o grau de exigência associado à aprendizagem da programação e justifica a pertinência do presente estudo, ao procurar diagnosticar as principais dificuldades enfrentadas pelos alunos da 10.ª classe no processo de ensino e aprendizagem desta disciplina fundamental para o perfil profissional pretendido.

2.2. Caracterização e Descrição Detalhada do Instrumento Utilizado.

O instrumento de recolha de dados aplicado para os alunos, consistiu num questionário estruturado elaborado com o objectivo de identificar e analisar as

dificuldades enfrentadas pelos alunos na aprendizagem da disciplina de Programação, no contexto do curso de Informática e Gestão do Instituto Politécnico do Lubango (IPOLUB).

De forma mais específica, o questionário procurou recolher a percepção dos estudantes relativamente aos principais obstáculos sentidos ao longo do processo de aprendizagem, incidindo sobre aspectos relacionados com a compreensão dos conceitos, a aplicação prática dos conteúdos, a utilização correcta da sintaxe, a resolução de erros e o uso de ferramentas de programação. Importa salientar que o instrumento não teve como finalidade avaliar o desempenho objectivo dos alunos, mas sim compreender a forma como estes percepcionam as suas próprias dificuldades, estando, por isso, alinhado com a natureza descritiva do presente estudo.

O questionário foi construído com base no Computational Thinking Framework for Programming (CFCP), sendo organizado em seis dimensões principais, de modo a abranger diferentes níveis e tipos de dificuldades inerentes à aprendizagem da programação. A primeira dimensão diz respeito às dificuldades sintácticas, relacionadas com a memorização e aplicação das regras da linguagem de programação e com a ocorrência de erros de escrita do código. A segunda dimensão corresponde às dificuldades semânticas, associadas à compreensão do funcionamento do código, à previsão da saída de um programa e à interpretação das estruturas de controlo. A terceira dimensão incide sobre as dificuldades pragmáticas, ligadas à aplicação prática dos conceitos aprendidos, à selecção adequada das estruturas de controlo e à transformação de problemas reais em soluções programáveis. A quarta dimensão contempla as dificuldades metacognitivas, relacionadas com a capacidade de identificar erros, planear estratégias de estudo e monitorizar o próprio processo de aprendizagem. A quinta dimensão refere-se às dificuldades associadas ao uso de ferramentas e ambientes de programação, nomeadamente à utilização do ambiente de desenvolvimento, à interpretação das mensagens de erro e ao recurso a ferramentas alternativas. E por fim, a dimensão motivacional que tem como objetivo analisar os factores motivacionais e como eles interferem durante o processo de ensino e aprendizagem da programação.

No que respeita à sua composição, o questionário integra um total de 24 questões. Destas, três correspondem a dados pessoais e académicos dos participantes, nomeadamente idade, sexo e contacto prévio com programação. As restantes questões distribuem-se pelas cinco dimensões referidas, assegurando uma cobertura equilibrada dos diferentes factores que influenciam o processo de aprendizagem da programação.

As respostas aos itens foram recolhidas através de uma escala de Likert de cinco pontos, variando entre 1 (“Nunca”) e 5 (“Sempre”). Esta escala permitiu aferir a frequência com que os estudantes experienciam determinadas dificuldades ou comportamentos relacionados com a aprendizagem da programação, possibilitando a conversão das respostas em dados numéricos passíveis de tratamento estatístico. A opção por esta escala revelou-se adequada aos objectivos do estudo, uma vez que permite captar diferentes graus de intensidade na percepção dos alunos, indo além de respostas dicotómicas.

Para efeitos de análise e interpretação dos resultados, as respostas referentes às dificuldades foram agregadas através do cálculo da média dos itens correspondentes, tendo posteriormente sido transformadas para uma escala de 0 a 100, por meio de uma conversão linear. Esta transformação teve como finalidade facilitar a leitura e a compreensão dos resultados, sem alterar o significado estatístico dos dados. Com base nesta escala, os níveis de dificuldade foram posteriormente classificados em baixa, moderada e elevada dificuldade.

2.3. Análise e Interpretação dos Resultados

O presente capítulo apresenta e discute os resultados obtidos através do questionário aplicado aos estudantes, com o propósito de diagnosticar as principais dificuldades sentidas no processo de ensino e aprendizagem da programação. Os dados foram tratados e analisados com recurso ao programa SPSS, sendo apresentados sob a forma de tabelas e gráficos, acompanhados de uma interpretação analítica e descritiva dos resultados.

2.3.1. Distribuição por gênero

O primeiro aspecto analisado refere-se à distribuição dos participantes segundo o gênero. De acordo com o gráfico de frequência (Gráfico 1), observa-se a proporção de estudantes do gênero masculino e feminino que responderam ao questionário.

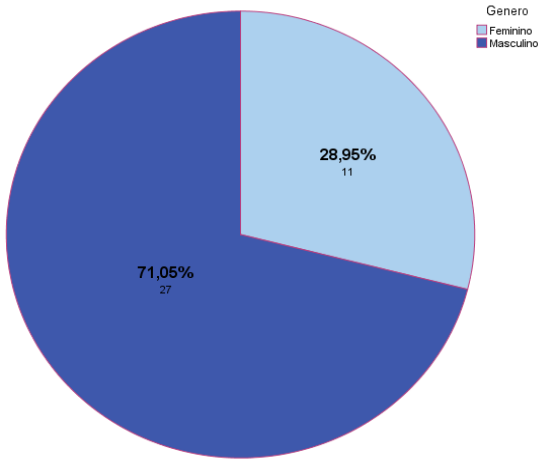

Gráfico 1 - Distribuição por gênero

Verifica-se que maioria dos participantes é do gênero masculino, representando 71,05% do total da amostra, enquanto o outro grupo corresponde a 28,95%. Essa informação evidencia a predominância de um gênero na composição da turma, o que pode refletir a tendência observada em curso de tecnologia.

2.3.2. Distribuição por faixa etária

No que respeita à faixa etária, o Gráfico 2 mostra a distribuição das idades dos respondentes. Constata-se que a maior parte dos estudantes se encontra na faixa etária compreendida entre 15 a 17 anos, seguindo-se a faixa entre 18 a 20 anos.

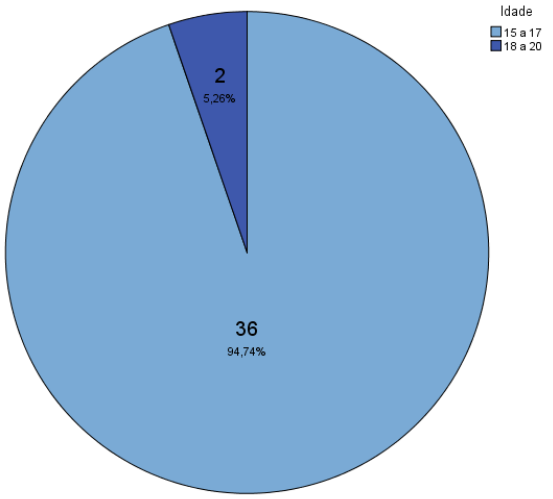

Gráfico 2 - Distribuição por faixa etária

A concentração de estudantes em faixas etárias mais jovens sugere que a maioria se encontra em fases iniciais da formação académica, o que poderá ter implicações na experiência prévia com conceitos de lógica e programação.

2.3.3. Contacto prévio com programação

Outro aspecto relevante para a compreensão do contexto dos participantes é o contacto prévio com programação. Conforme demonstra o Gráfico 3, 34,21% dos estudantes afirmaram já ter tido algum tipo de contacto anterior com programação, enquanto 65,79% indicaram nunca ter tido qualquer experiência anterior.

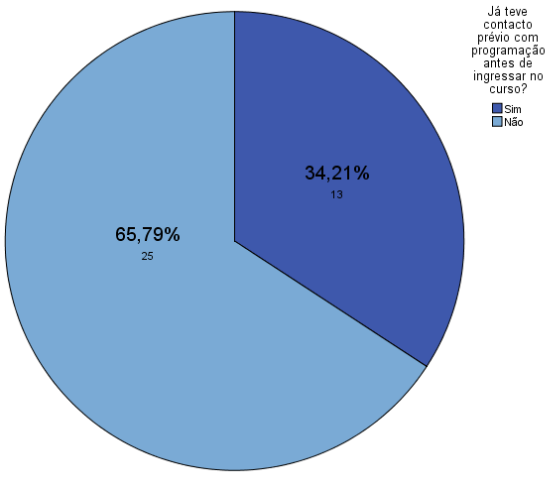

Gráfico 3 - Contacto prévio com programção

Este dado é particularmente importante, uma vez que o contacto prévio tende a facilitar a assimilação dos conceitos básicos, enquanto a ausência dessa experiência pode constituir um factor de dificuldade acrescida no processo de aprendizagem.

2.3.4. Diagnóstico das Dificuldades Específicas

Nesta parte do estudo, procurou-se identificar as dificuldades mais frequentemente relatadas pelos estudantes em aspectos concretos do processo de aprendizagem. Conforme apresentado nas tabelas à baixo, a análise dos dados recolhidos através da escala de Likert de 5 pontos, permite delinear o perfil das dificuldades percebidas pelos estudantes da 10.ª classe no domínio da Programação, focando-se na proporção de respostas que indicam a ocorrência da dificuldade ("Às vezes", "Frequentemente" e "Sempre").

2.3.4.1. Dimensão 1- Dificuldades Sintáticas

Tabela 1- Diagnóstico de dificuldade(1)

Tenho dificuldade em lembrar a sintaxe correta dos comandos da linguagem de programação.				
		Frequency	Percent	Valid Percent
Valid	Nunca	3	7,3	7,9
	Raramente	11	26,8	28,9
	Às vezes	20	48,8	52,6
	Frequentemente	3	7,3	7,9
	Sempre	1	2,4	2,6
	Total	38	92,7	100,0

A dificuldade em reter a sintaxe correcta dos comandos da linguagem de programação é reportada por 63,1% dos inquiridos (52,6% "Às vezes", 7,9% "Frequentemente" e 2,6% "Sempre"). Este dado sugere que, embora a sintaxe seja um aspecto fundamental, a sua memorização constitui um obstáculo significativo para a maioria dos estudantes, podendo comprometer a fluidez na escrita do código.

Tabela 2- Diagnóstico de dificuldade(2)

Esqueço com frequência símbolos como ponto e vírgula, parênteses ou chaves.				
		Frequency	Percent	Valid Percent
Valid	Nunca	14	34,1	36,8
	Raramente	10	24,4	26,3
	Às vezes	12	29,3	31,6
	Frequentemente	1	2,4	2,6
	Sempre	1	2,4	2,6
	Total	38	92,7	100,0

Relativamente ao esquecimento de símbolos básicos (ponto e vírgula, parênteses), 36,8% dos estudantes reportam esta dificuldade (31,6% "Às vezes", 2,6% "Frequentemente" e 2,6% "Sempre"). Comparativamente à dificuldade geral de memorização da sintaxe, esta dificuldade específica é menos prevalente, indicando que o problema reside mais na estrutura dos comandos do que nos elementos de pontuação.

Tabela 3- Diagnóstico de dificuldade(3)

Perco muito tempo tentando corrigir erros de sintaxe.				
		Frequency	Percent	Valid Percent
Valid	Nunca	5	12,2	13,2
	Raramente	9	22,0	23,7
	Às vezes	13	31,7	34,2
	Frequentemente	8	19,5	21,1
	Sempre	3	7,3	7,9
	Total	38	92,7	100,0

A gestão do tempo na depuração de erros sintácticos é um problema para 63,2% dos estudantes (34,2% "Às vezes", 21,1% "Frequentemente" e 7,9% "Sempre"). Embora a sintaxe seja uma dificuldade menos crítica do que a lógica, o tempo despendido na sua correcção sugere ineficiência no processo de debugging e uma possível falta de familiaridade com as ferramentas de desenvolvimento.

2.3.4.2. Dimensão 2- Dificuldades Semânticas

Tabela 4- Diagnóstico de dificuldade(4)

Mesmo sem erros de sintaxe, muitas vezes o programa não funciona como eu esperava.				
		Frequency	Percent	Valid Percent
Valid	Nunca	3	7,3	7,9
	Raramente	10	24,4	26,3
	Às vezes	14	34,1	36,8
	Frequentemente	11	26,8	28,9
	Total	38	92,7	100,0

Um total de 65,7% dos inquiridos reporta que o programa não funciona conforme o esperado, mesmo na ausência de erros sintácticos (36,8% "Às vezes" e 28,9% "Frequentemente"). Este é um dos problemas mais expressivos, evidenciando que a maior dificuldade reside na lógica e na semântica do código, e não apenas na sua forma.

Tabela 5- Diagnóstico de dificuldade(5)

Tenho dificuldade em entender o que um trecho de código realmente faz.				
		Frequency	Percent	Valid Percent
Valid	Nunca	4	9,8	10,5
	Raramente	16	39,0	42,1
	Às vezes	12	29,3	31,6
	Frequentemente	2	4,9	5,3
	Sempre	4	9,8	10,5
	Total	38	92,7	100,0

A compreensão semântica do código, ou seja, a capacidade de interpretar a função de um segmento de código, é um desafio para 47,4% dos participantes (31,6% "Às vezes", 5,3% "Frequentemente" e 10,5% "Sempre"). Este resultado aponta para uma lacuna na assimilação dos conceitos subjacentes à sintaxe, dificultando a leitura e a manutenção de código.

Tabela 6- Diagnóstico de dificuldade(6)

Sinto dificuldade em prever a saída (resultado) de um programa.				
		Frequency	Percent	Valid Percent
Valid	Nunca	8	19,5	21,1
	Raramente	13	31,7	34,2
	Às vezes	9	22,0	23,7
	Frequentemente	5	12,2	13,2
	Sempre	3	7,3	7,9
	Total	38	92,7	100,0

A capacidade de traçar a execução de um programa e prever o seu resultado final é percebida como difícil por 44,8% dos estudantes (23,7% "Às vezes", 13,2% "Frequentemente" e 7,9% "Sempre"). Esta dificuldade está intrinsecamente ligada à falta de domínio da lógica de programação e à incapacidade de simular mentalmente o fluxo de controlo do código.

Tabela 7- Diagnóstico de dificuldade(7)

Quando penso em estruturas com if, for ou while, sinto dificuldade em imaginar como funcionam na prática				
		Frequency	Percent	Valid Percent
Valid	Nunca	14	34,1	36,8
	Raramente	7	17,1	18,4
	Às vezes	6	14,6	15,8
	Frequentemente	8	19,5	21,1
	Sempre	3	7,3	7,9
	Total	38	92,7	100,0

A dificuldade na visualização mental do funcionamento das estruturas de controlo é reportada por 44,8% dos estudantes (15,8% "Às vezes", 21,1% "Frequentemente" e 7,9% "Sempre"). Este dado reforça a necessidade de abordagens pedagógicas que promovam a visualização e a simulação do fluxo de execução do código.

2.3.4.3. Dimensão 3- Dificuldades Pragmáticas

Tabela 8- Diagnóstico de dificuldade(8)

Fico indeciso sobre qual estrutura (if, while, for, etc.) devo usar em determinados problemas				
		Frequency	Percent	Valid Percent
Valid	Nunca	8	19,5	21,1
	Raramente	11	26,8	28,9
	Às vezes	10	24,4	26,3
	Frequentemente	7	17,1	18,4
	Sempre	2	4,9	5,3
	Total	38	92,7	100,0

A incerteza na escolha da estrutura de controlo mais adequada para a resolução de um problema é reportada por exatamente 50,0% dos estudantes (26,3% "Às vezes", 18,4% "Frequentemente" e 5,3% "Sempre"). Este dado sublinha uma dificuldade na aplicação pragmática dos conceitos de controlo de fluxo, um pilar da lógica de programação.

Tabela 9- Diagnóstico de dificuldade(9)

Tenho dificuldade em aplicar conceitos de programação em situações práticas.				
		Frequency	Percent	Valid Percent
Valid	Nunca	6	14,6	15,8
	Raramente	12	29,3	31,6
	Às vezes	11	26,8	28,9
	Frequentemente	5	12,2	13,2
	Sempre	4	9,8	10,5
	Total	38	92,7	100,0

A transposição dos conceitos teóricos de programação para a resolução de problemas práticos é um desafio para 52,6% dos inquiridos (28,9% "Às vezes", 13,2% "Frequentemente" e 10,5% "Sempre"). Este resultado reforça a dificuldade na dimensão pragmática, sugerindo que a aprendizagem é, por vezes, descontextualizada ou que os estudantes não conseguem realizar a ponte entre a teoria e a prática.

Tabela 10- Diagnóstico de dificuldade(10)

Não consigo transformar um problema real em código de forma clara.				
		Frequency	Percent	Valid Percent
Valid	Nunca	4	9,8	10,5
	Raramente	11	26,8	28,9
	Às vezes	12	29,3	31,6
	Frequentemente	6	14,6	15,8
	Sempre	5	12,2	13,2
	Total	38	92,7	100,0

A incapacidade de traduzir um problema do mundo real para uma representação algorítmica clara é reportada por 50,6% dos estudantes (31,6% "Às vezes", 15,8% "Frequentemente" e 13,2% "Sempre"). Esta dificuldade, que se situa na fase inicial do desenvolvimento de software, é crucial e indica uma falha na capacidade de abstração e modelagem de problemas.

2.3.4.4. Dimensão 4- Dificuldades Metacognitivas

Tabela 11- Diagnóstico de dificuldade(11)

Quando meu código não funciona, tenho dificuldade em identificar onde está o erro.				
		Frequency	Percent	Valid Percent
Valid	Nunca	4	9,8	10,5
	Raramente	11	26,8	28,9
	Às vezes	10	24,4	26,3
	Frequentemente	11	26,8	28,9
	Sempre	2	4,9	5,3
	Total	38	92,7	100,0

No processo de depuração, 60,1% dos estudantes (26,3% "Às vezes", 28,9% "Frequentemente" e 5,3% "Sempre") reportam dificuldade em localizar a origem do erro quando o código falha. Este ponto sugere uma ineficácia nas estratégias de debugging, o que pode estar relacionado com a dificuldade em prever o resultado do programa (Tabela 6).

Tabela 12- Diagnóstico de dificuldade(12)

Tenho dificuldade em organizar meu estudo para aprender programação sozinho(a).				
		Frequency	Percent	Valid Percent
Valid	Nunca	3	7,3	7,9
	Raramente	9	22,0	23,7
	Às vezes	12	29,3	31,6
	Frequentemente	6	14,6	15,8
	Sempre	8	19,5	21,1
	Total	38	92,7	100,0

A dificuldade em gerir e organizar o estudo autónomo é a dificuldade mais expressiva, sendo reportada por 72,0% dos estudantes (31,6% "Às vezes", 15,8% "Frequentemente" e 21,1% "Sempre"). Este resultado aponta para uma fragilidade metacognitiva e de auto-regulação da aprendizagem, sugerindo que os estudantes necessitam de maior apoio na gestão do seu processo de estudo.

2.3.4.5. Dimensão 5- Dificuldades com Ferramentas/ IDE

Tabela 13- Diagnóstico de dificuldade(13)

Tenho dificuldade em utilizar o ambiente de programação (IDE).				
		Frequency	Percent	Valid Percent
Valid	Nunca	9	22,0	23,7
	Raramente	11	26,8	28,9
	Às vezes	13	31,7	34,2
	Frequentemente	3	7,3	7,9
	Sempre	2	4,9	5,3
	Total	38	92,7	100,0

A utilização do Ambiente de Desenvolvimento Integrado (IDE) é um obstáculo para 47,4% dos estudantes (34,2% "Às vezes", 7,9% "Frequentemente" e 5,3% "Sempre"). Embora não seja uma dificuldade maioritária, a percentagem é significativa e indica que a ferramenta de trabalho, em si, pode estar a adicionar uma camada de complexidade ao processo de aprendizagem.

Tabela 14- Diagnóstico de dificuldade(14)

Não compreendo bem as mensagens de erro que aparecem no compilador.				
		Frequency	Percent	Valid Percent
Valid	Nunca	2	4,9	5,3
	Raramente	12	29,3	31,6
	Às vezes	11	26,8	28,9
	Frequentemente	9	22,0	23,7
	Sempre	4	9,8	10,5
	Total	38	92,7	100,0

A interpretação das mensagens de erro do compilador é difícil para 62,9% dos inquiridos (28,9% "Às vezes", 23,7% "Frequentemente" e 10,5% "Sempre"). Este é um dos problemas mais críticos, pois a incapacidade de entender o feedback do sistema impede a correção autónoma dos erros, tornando o processo de aprendizagem dependente da intervenção externa.

Tabela 15- Diagnóstico de dificuldade(15)

Perco muito tempo configurando ou ajustando o ambiente de programação.				
		Frequency	Percent	Valid Percent
Valid	Nunca	8	19,5	21,1
	Raramente	13	31,7	34,2
	Às vezes	13	31,7	34,2
	Frequentemente	4	9,8	10,5
	Total	38	92,7	100,0

A perda de tempo na configuração do ambiente de programação é reportada por 44,7% dos estudantes (34,2% "Às vezes" e 10,5% "Frequentemente"). Esta dificuldade, embora técnica, consome tempo de aprendizagem e pode gerar frustração inicial.

De forma geral, a dificuldade mais expressiva é de natureza metacognitiva, com 72,0% dos estudantes a reportarem dificuldade em organizar o estudo autónomo. No domínio das dificuldades de Natureza Pragmática e de Resolução

de Problemas, os dados indicam que a maioria dos estudantes enfrenta desafios significativos, nomeadamente: 65,7% reportam que o programa não funciona como esperado, mesmo na ausência de erros sintácticos; 63,2% reportam perda de tempo na correcção de erros de sintaxe; e 60,1% reportam dificuldade em identificar a origem do erro quando o código falha. Adicionalmente, 52,6% dos estudantes reportam dificuldade em aplicar conceitos de programação em situações práticas, e 50,6% reportam dificuldade em transformar um problema real em código de forma clara. As dificuldades de Natureza Sintáctica e Semântica também apresentam alta prevalência, com 63,1% dos estudantes a reportarem dificuldade em reter a sintaxe correcta dos comandos. Por fim, no domínio das Ferramentas, 62,9% dos estudantes reportam dificuldade em interpretar as mensagens de erro do compilador, o que constitui um obstáculo significativo à depuração autónoma.

Em suma, as dificuldades mais críticas, que afectam mais de 60% dos inquiridos, concentram-se na organização do estudo autónomo, na sintaxe, na depuração de erros lógicos e na interpretação do feedback do compilador.

2.3.5. Dificuldade Geral em Programação

Com vista a obter uma visão global do nível de dificuldade percepcionado, calculou-se a média geral da dificuldade atribuída pelos estudantes. Os resultados indicam que o valor médio do grau de dificuldade geral é de 49,9, numa escala em que zero (0) corresponde à ausência de dificuldade e cem (100) à dificuldade máxima. Este valor situa-se no intervalo definido como dificuldade moderada, sugerindo que, de forma global, os estudantes percepcionam a aprendizagem da programação como um processo que apresenta desafios relevantes, mas não extremos. O desvio-padrão de 13,04 demonstra uma média dispersão das respostas indicando que, embora exista uma tendência central em torno de um nível intermédio de dificuldade, há variações individuais significativas na forma como os alunos experienciam o processo de aprendizagem. Este resultado sugere a coexistência de estudantes que sentem dificuldades mais acentuadas com outros que revelam maior facilidade, o que reflecte a heterogeneidade da turma analisada.

Tabela 16- Nível de dificuldade

	N	Mínimo	Maximo	Média	Desvio padrão
Grau de Dificuldade Geral (0 = Nenhuma, 100 = Máxima)	38	13,1	79,8	49,906	13,0421

O Gráfico 4 apresenta a distribuição dos níveis de dificuldade, evidenciando a predominância de percepções (moderadas/altas/baixas) entre os inquiridos.

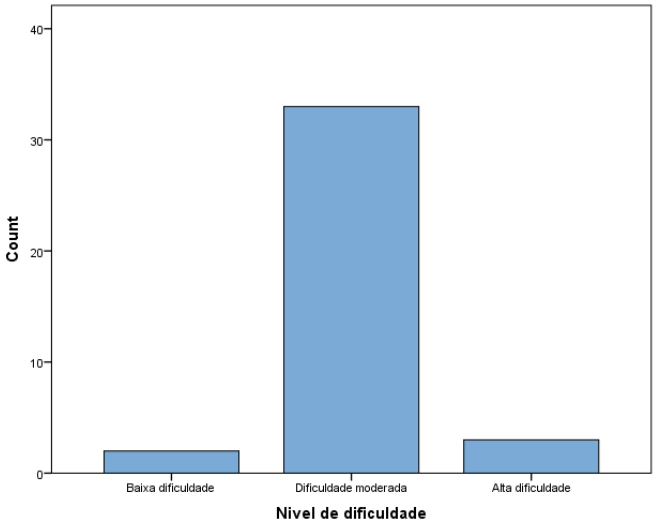

Gráfico 4- Distribuição dos níveis

2.3.6. Comparações entre Grupos

Nesta secção, apresentam-se as comparações entre grupos, com o objectivo de verificar se variáveis como o género e contacto prévio com programação, exercem influência sobre o nível de dificuldade geral relatado pelos estudantes.

2.3.6.1. Género e dificuldade geral

Recorreu-se ao teste-t de amostras independentes, a fim de comparar as médias de dificuldade entre estudantes do género masculino e feminino. Os resultados obtidos indicam que não se verificam diferenças estatisticamente significativas (Sig. = 0,165).

Tabela 17- Teste-t para amostras independentes (1)

Teste-t para amostras independentes				
		Teste de Levene para igualdade das variações		Teste-t para igualdade das médias
		F	Sig.	t
Grau de Dificuldade Geral (0 = Nenhuma, 100 = Máxima)	Variação iguais assumidas	0,063	0,803	1,419
	Variação iguais não assumidas			1,441

Tabela 18- Teste-t para amostras independentes (2)

Teste-t para amostras independentes				
		Teste-t para igualdade das médias		
		df	Sig. (2-tailed)	Diferença das médias
Grau de Dificuldade Geral (0 = Nenhuma, 100 = Máxima)	Variação iguais assumidas	36	0,165	6,5296
	Variação iguais não assumidas	19,263	0,166	6,5296

Tabela 19- Teste-t para amostras independentes (3)

Teste-t para amostras independentes	
	Teste-t para igualdade das médias

		Erro padrão de diferença de médias	Intervalo de confiança a 95% da diferença
			Limite inferior
Grau de Dificuldade Geral (0 = Nenhuma, 100 = Máxima)	Variação iguais assumidas	4,6026	-2,8048
	Variação iguais não assumidas	4,5305	-2,9440

Tabela 20- Teste-t para amostras independentes (4)

Teste-t para amostras independentes		
		Teste-t para igualdade das médias
		Intervalo de confiança a 95% da diferença
		Limite Superior
Grau de Dificuldade Geral (0 = Nenhuma, 100 = Máxima)	Variação iguais assumidas	15,8640
	Variação iguais não assumidas	16,0032

O teste de Levene (F = 0,063, p = 0,803) indicou que as variâncias dos dois grupos são homogéneas, assim, foi considerada a linha “Variação iguais assumidas” para interpretação do teste-t.

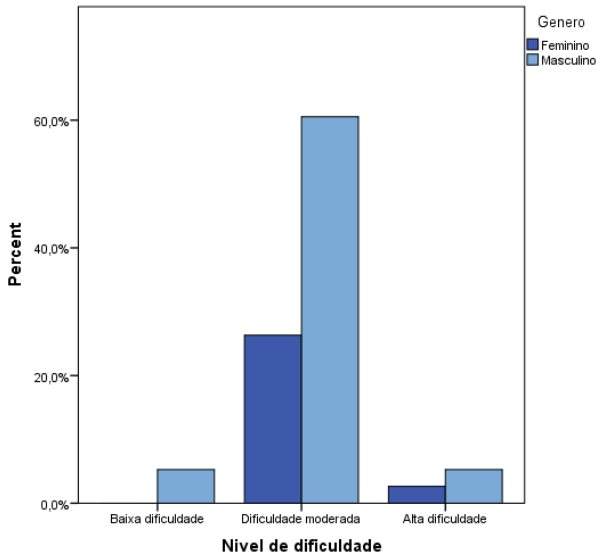

Gráfico 5 - Nível de dificuldade geral por gênero

O teste-t (t(36) =1,419, p = 0,165) e o Gráfico 5, revelaram que não existe diferença estatisticamente significativas entre médias dos dois grupos quanto ao grau de dificuldade geral em programação (diferença média = 6,53).

2.3.6.2. Contacto prévio e dificuldade geral

Analisou-se, igualmente, a relação entre o contacto prévio com programação e o nível de dificuldade geral. Para analisar esta relação, recorreu-se ao procedimento Compare Means no SPSS, permitindo calcular as médias do grau de dificuldade geral em função do contacto prévio com programação. Tabela 21 apresenta a comparação descritiva do grau de dificuldade geral em programação entre estudantes que afirmaram ter tido contacto prévio com programação antes de ingressarem no curso e aqueles que não tiveram qualquer contacto anterior.

Tabela 21- Contacto prévio e dificuldade geral

	Já teve contacto prévio com programação antes de ingressar no curso?	N	Média	Desvio Padrão
Grau de Dificuldade Geral	Sim	13	43,590	14,4545
	Não	25	53,190	11,1767

Observa-se que os estudantes com contacto prévio (N = 13) apresentam uma média de dificuldade geral de 43,59, enquanto os estudantes sem contacto prévio (N = 25) registam uma média superior, de 53,19, numa escala de 0 a 100. Esta diferença descritiva sugere que os estudantes que já tinham alguma experiência prévia com programação tendem a percepcionar menores níveis de dificuldade na aprendizagem da disciplina, quando comparados com aqueles que iniciaram o curso sem qualquer contacto anterior. No que diz respeito à dispersão dos dados, o desvio-padrão é mais elevado no grupo com contacto prévio (DP = 14,45) do que no grupo sem contacto (DP = 11,18), este resultado indica uma maior variabilidade nas percepções de dificuldade entre os estudantes que tiveram contacto prévio, sugerindo que, embora em média apresentem menores dificuldades, as experiências individuais dentro deste grupo são mais heterogéneas, por outro lado, o grupo sem contacto prévio revela uma distribuição mais homogénea das dificuldades percebidas, concentrando-se em níveis consistentemente mais elevados.

De acordo com o Gráfico 6, apresenta-se de forma visual a distribuição dos níveis de dificuldade geral em função do contacto prévio dos estudantes com a programação antes do ingresso no curso. O gráfico ilustra as tendências já evidenciadas na tabela anterior, permitindo uma leitura mais clara da proporção de estudantes com dificuldades baixas, moderadas e elevadas nos dois grupos considerados.

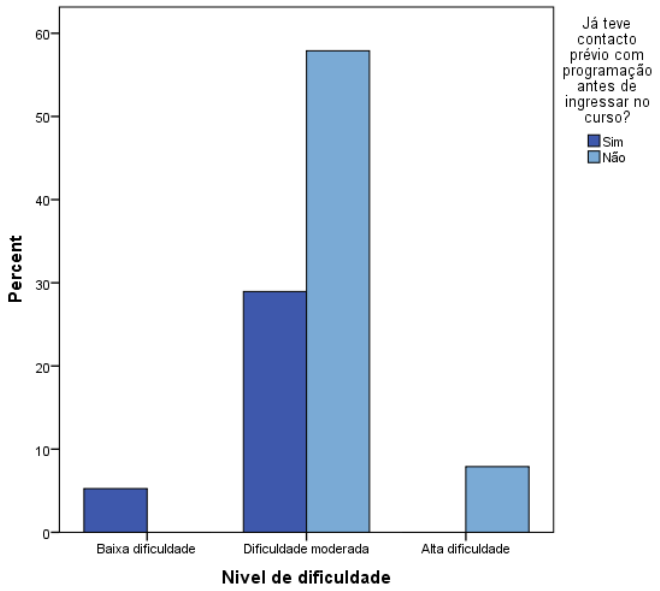

Gráfico 6 - Contacto prévio e dificuldade geral

Os resultados sugerem que os estudantes que referiram ter tido contacto prévio com programação apresentam, em média, níveis mais baixos de dificuldade geral. No entanto, tendo em conta a dimensão reduzida da amostra e o seu carácter não probabilístico, estes dados devem ser interpretados com prudência, não permitindo estabelecer relações de influência ou generalização para além do contexto específico da turma analisada.

2.3.7. Factores motivacionais

Com vista a complementar o diagnóstico das dificuldades enfrentadas pelos alunos no processo de ensino e aprendizagem de programação, procedeu-se igualmente à análise descritiva dos factores motivacionais associados a esta disciplina. Esta análise teve como objectivo compreender as percepções, atitudes e crenças dos alunos relativamente à sua motivação para aprender programação, bem como a forma como reagem perante desafios e dificuldades ao longo do processo de aprendizagem.

Os factores motivacionais foram analisados individualmente, recorrendo-se a medidas de tendência central e dispersão (mínimo, máximo, média e desvio-padrão), de modo a caracterizar o comportamento global das respostas dos alunos em cada item (Tabela 22).

Tabela 22- Factores Motivacionais

	Mínimo	Máximo	Média	Desvio Padrão
Acredito que a minha motivação influencia no meu desempenho em programação.	2	5	4,66	0,771
Quando consigo resolver um problema de programação, sinto-me mais motivado a continuar aprendendo.	2	5	4,68	0,765
Sinto que programação é uma disciplina muito difícil para mim.	1	5	2,53	1,095
Quando encontro dificuldades, tenho vontade de desistir facilmente.	1	5	1,95	1,076
Acredito que sou capaz de aprender programação com esforço e prática.	3	5	4,66	0,699

De um modo geral, os resultados apontam para elevados níveis de concordância quanto ao papel da motivação no desempenho académico, bem como para uma forte crença na capacidade individual de aprender programação através do esforço e da prática.

Com efeito, os itens relacionados com a motivação intrínseca apresentaram médias elevadas: a afirmação “Acredito que a minha motivação influencia no meu desempenho em programação” registou uma média de 4,66, enquanto o item “Quando consigo resolver um problema de programação, sinto-me mais motivado a continuar aprendendo” apresentou a média mais elevada (4,68). Estes valores, próximos do máximo da escala, indicam que os alunos reconhecem claramente a motivação como um elemento central no seu processo de aprendizagem, sendo a resolução bem-sucedida de problemas um importante reforço motivacional.

Por outro lado, os itens associados à percepção de dificuldade e à desistência apresentaram médias consideravelmente mais baixas: a afirmação “Sinto que programação é uma disciplina muito difícil para mim” obteve uma média de 2,53, situada abaixo do ponto intermédio da escala, sugerindo que, de forma geral, os alunos não percepcionam a programação como excessivamente difícil, apesar de existir alguma variabilidade nas respostas. De igual modo, o item “Quando encontro dificuldades, tenho vontade de desistir facilmente” registou uma média de 1,95, indicando que a maioria dos alunos não demonstra uma tendência acentuada para desistir perante os desafios encontrados.

Finalmente, a forte crença na auto-eficácia académica é evidenciada pelo item “Acredito que sou capaz de aprender programação com esforço e prática”, que apresentou uma média de 4,66, com valores mínimos relativamente elevados. Este resultado reforça a ideia de que os alunos possuem uma percepção positiva das suas capacidades, o que constitui um factor motivacional relevante no contexto da aprendizagem da programação.

Em síntese, os dados sugerem que os alunos apresentam níveis elevados de motivação, persistência e confiança nas suas capacidades, coexistindo com uma percepção moderada da dificuldade da disciplina. Estes resultados permitem caracterizar o perfil motivacional dos alunos e fornecem uma base consistente para a análise da relação entre factores motivacionais e a dificuldade percebida na aprendizagem da programação.

2.3.8. Parecer pedagógico – Análise qualitativa

Com base nas respostas obtidas no questionário aplicado aos docentes, procedeu-se à organização e sistematização da informação por meio de categorias de análise, definidas em função das questões colocadas e do conteúdo recorrente nos discursos dos professores. As categorias permitiram agrupar as perceções dos docentes relativamente às dificuldades na aprendizagem da programação, aos fatores que contribuem para essas dificuldades, à influência do contacto prévio com a disciplina e às dificuldades observadas nas aulas práticas que nem sempre são verbalizadas pelos alunos. Esta categorização possibilita uma leitura comparativa e estruturada dos

pareceres, facilitando a interpretação dos dados qualitativos no contexto do estudo.

Tabela 23- Parecer Docente

Categorias de análise	Professor A	Professor B
Dificuldades na aprendizagem da programação	Falta de continuidade de estudos, escassez de recursos computacionais e dificuldades na língua inglesa	Programação como disciplina nova e dificuldades em matemática
Fatores determinantes das dificuldades	Ausência de meios informáticos individuais, fraco estudo pós-aula e necessidade de formação em língua inglesa	Fragilidades em matemática e no raciocínio lógico
Contacto prévio com a programação	Defende a introdução da programação em classes anteriores ao II ciclo	Defende a introdução da programação em classes anteriores ao II ciclo
Dificuldades observadas nas aulas práticas	Pouco domínio teórico e dificuldade na aplicação prática dos conteúdos	Dificuldades na compreensão e aplicação dos conteúdos durante as aulas práticas

A análise dos pareceres dos docentes, apresentada na Tabela 23, permite compreender de forma mais aprofundada as dificuldades enfrentadas pelos alunos na aprendizagem da programação, a partir das respostas às questões constantes do questionário aplicado aos professores (*Apêndice 2*). No que se refere à primeira questão, relativa às principais dificuldades observadas na 10.ª classe, os docentes apontam, de forma convergente, a novidade da disciplina para a maioria dos alunos, bem como limitações associadas aos conhecimentos

matemáticos, ao raciocínio lógico e às condições de estudo, aspetos que condicionam o acompanhamento da sequência dos conteúdos lecionados.

Relativamente à segunda questão, que procurou identificar os fatores mais determinantes para essas dificuldades, os professores destacam elementos de natureza contextual e pedagógica, como a escassez de meios informáticos para a realização de práticas e investigações pós-aula, a ausência de hábitos de estudo autónomo e as dificuldades associadas à língua inglesa, frequentemente utilizada em ambientes e materiais de programação. Estes fatores contribuem para a fragilização do processo de aprendizagem e limitam a consolidação dos conteúdos fora do contexto da sala de aula.

No que diz respeito à terceira questão, referente à influência do contacto prévio com a programação, ambos os docentes reconhecem que a sua ausência afeta negativamente o desempenho dos alunos, defendendo a introdução da programação em níveis de ensino anteriores como uma estratégia que poderia facilitar a adaptação dos alunos à disciplina na 10.ª classe e melhorar o seu rendimento académico.

Por fim, em resposta à quarta questão, relacionada com as dificuldades que os alunos tendem a não verbalizar, mas que são observadas durante as aulas práticas, os professores referem dificuldades na aplicação dos conteúdos teóricos, falta de domínio de conceitos básicos e desorientação na resolução dos exercícios propostos. Estas dificuldades evidenciam fragilidades na articulação entre teoria e prática, sugerindo a pertinência de estratégias pedagógicas que promovam uma maior integração entre ambos os momentos do processo de ensino e aprendizagem.

2.3.9. Síntese dos Resultados

A análise dos dados permitiu identificar um conjunto de dificuldades enfrentadas pelos alunos da 10.ª classe na aprendizagem da programação, evidenciando que estas dificuldades se manifestam em diferentes dimensões do processo de ensino e aprendizagem. Os resultados quantitativos indicam que os alunos enfrentam obstáculos de natureza cognitiva, prática e motivacional, os quais condicionam o seu desempenho na disciplina.

No que se refere às dificuldades cognitivas, verificou-se que muitos alunos apresentam limitações ao nível da compreensão dos conceitos fundamentais da programação, bem como dificuldades relacionadas com o raciocínio lógico e a interpretação de problemas. Estes resultados encontram respaldo nos pareceres dos professores, que destacam a exigência de conhecimentos matemáticos e lógicos como um dos principais entraves à aprendizagem da programação. As dificuldades de ordem prática revelam-se sobretudo na aplicação dos conteúdos teóricos em actividades práticas, sendo frequente a desarticulação entre teoria e prática. Esta situação é corroborada pelos docentes, que observam, durante as aulas práticas, que a falta de domínio dos conceitos básicos compromete a capacidade dos alunos em resolver exercícios de forma autónoma. Relativamente ao contacto prévio com a programação, os dados quantitativos demonstram que a maioria dos alunos não teve experiências anteriores com esta área, o que se reflecte num maior nível de dificuldade na aprendizagem. Esta constatação é igualmente partilhada pelos professores, que consideram que a ausência de uma introdução precoce à programação dificulta o acompanhamento da sequência dos conteúdos leccionados. Para além dos aspectos cognitivos e pedagógicos, os resultados evidenciam a influência de factores contextuais e motivacionais, como a falta de meios informáticos para a prática fora do contexto escolar, a ausência de hábitos de estudo autónomo e as dificuldades relacionadas com a língua inglesa. Estes factores, referidos tanto pelos alunos como pelos professores, contribuem para agravar as dificuldades no processo de aprendizagem da programação.

De forma geral, os resultados obtidos permitem concluir que as dificuldades enfrentadas pelos alunos na aprendizagem da programação resultam da conjugação de factores individuais, pedagógicos e contextuais. A identificação destes factores constitui uma base sólida para a análise crítica dos resultados e para a formulação de propostas de intervenção, a desenvolver no capítulo seguinte.

CAPÍTULO III - CONCLUSÃO

CONCLUSÕES

O presente estudo teve como objectivo geral investigar as principais dificuldades enfrentadas pelos alunos da 10.ª classe do curso de Informática Aplicada à Gestão do IPOLub no processo de ensino e aprendizagem da programação. Para o efeito, formularam-se questões de investigação orientadas para a identificação das dificuldades específicas, a análise do nível de dificuldade geral percepcionado, a verificação de possíveis diferenças associadas ao género, a influência do contacto prévio com programação e a caracterização do perfil motivacional dos alunos.

A análise dos dados quantitativos recolhidos junto dos alunos permitiu concluir que as dificuldades na aprendizagem da programação assumem um carácter multifacetado, não se limitando a aspectos pontuais da sintaxe, com efeito, verificou-se uma elevada incidência de dificuldades de natureza metacognitiva, nomeadamente na organização do estudo autónomo, bem como dificuldades associadas à depuração de erros, à interpretação das mensagens do compilador e à aplicação prática dos conceitos de programação, estes resultados indicam que os principais obstáculos residem na compreensão do funcionamento do código, na resolução de problemas e na gestão do próprio processo de aprendizagem.

Relativamente ao nível de dificuldade geral percepcionado, os resultados evidenciaram que a programação é encarada pelos alunos como uma disciplina de dificuldade moderada, no entanto, o desvio-padrão registado demonstra a existência de diferenças individuais significativas, revelando a coexistência de alunos que experienciam dificuldades mais acentuadas com outros que apresentam maior facilidade, esta heterogeneidade reforça a necessidade de estratégias pedagógicas diferenciadas, capazes de responder a perfis de aprendizagem distintos.

No que diz respeito à influência do género no nível de dificuldade geral, os resultados estatísticos não evidenciaram diferenças significativas entre alunos do sexo masculino e feminino. Este achado sugere que, no contexto específico da turma analisada, o género não constitui um factor determinante na

aprendizagem da programação, contrariando algumas percepções comuns associadas às áreas tecnológicas.

Por outro lado, a análise da relação entre o contacto prévio com programação e o nível de dificuldade revelou que os alunos com experiência anterior tendem a percepcionar menores dificuldades na aprendizagem da disciplina, embora esta relação tenha sido analisada de forma descritiva, os resultados apontam para a importância da exposição antecipada à programação como elemento facilitador do processo de aprendizagem, reforçando a relevância de abordagens introdutórias nos níveis de ensino anteriores.

No que se refere ao perfil motivacional dos alunos, os resultados indicaram níveis elevados de motivação intrínseca, persistência e crença na capacidade de aprender programação através do esforço e da prática, paralelamente, a percepção da programação como excessivamente difícil e a tendência para desistir perante obstáculos apresentaram valores reduzidos. Estes dados permitem concluir que as dificuldades identificadas não estão associadas à desmotivação, mas antes a limitações cognitivas, metodológicas e contextuais.

A análise qualitativa das respostas dos professores veio reforçar e complementar os resultados obtidos junto dos alunos. Os docentes identificaram dificuldades relacionadas com o raciocínio lógico, a insuficiente base matemática, a falta de continuidade do estudo fora da sala de aula e as limitações de acesso a recursos tecnológicos. Esta convergência entre as percepções dos alunos e dos professores confere maior robustez aos resultados, evidenciando que as dificuldades observadas são estruturais e não meramente circunstanciais.

Importa ainda referir que algumas questões do questionário, nomeadamente as relacionadas com o contacto com ferramentas introdutórias como o Scratch, não foram exploradas em profundidade ao longo do estudo. Esta opção metodológica resultou da delimitação do objecto de investigação, centrado nas dificuldades, no nível de dificuldade geral e na motivação dos alunos, no entanto, estas questões revelam-se pertinentes enquanto indicadores de exposição inicial à programação, constituindo uma linha relevante para investigações futuras.

Em síntese, os resultados obtidos permitiram responder de forma clara às questões de investigação formuladas e atingir o objectivo geral do estudo, evidenciando que as dificuldades na aprendizagem da programação resultam de uma combinação de factores cognitivos, metacognitivos, pedagógicos e contextuais, mais do que de falta de motivação ou de características individuais como o género.

Sugestões

Com base nos resultados obtidos no presente estudo, sugere-se que investigações futuras possam aprofundar a análise das dificuldades na aprendizagem da programação por meio de abordagens de natureza experimental ou quase experimental. Em particular, recomenda-se a realização de estudos que avaliem a eficácia de estratégias pedagógicas alternativas capazes de facilitar a introdução dos conceitos fundamentais de programação junto dos alunos da 10.ª classe.

Uma possibilidade consiste na utilização do Scratch como ferramenta introdutória à programação, numa fase inicial do processo de ensino e aprendizagem. Por se tratar de uma linguagem visual e baseada em blocos, o Scratch poderá contribuir para a redução das dificuldades associadas ao raciocínio lógico, à compreensão de problemas e à articulação entre teoria e prática, identificadas no presente estudo. A utilização desta ferramenta poderá ainda atenuar a exigência imediata de conhecimentos matemáticos mais complexos e da língua inglesa, permitindo que os alunos se concentrem, numa fase inicial, na lógica dos algoritmos e na resolução de problemas de forma progressiva.

Sugere-se que futuros estudos adoptem um desenho metodológico experimental, envolvendo grupos de controlo e grupos experimentais, nos quais o Scratch seja utilizado numa primeira fase como estratégia de introdução à programação, sendo posteriormente efectuada a transição para linguagens de programação textuais leccionadas no currículo. Esta abordagem permitiria avaliar, de forma mais objectiva, o impacto da utilização do Scratch no desempenho, na motivação e na autonomia dos alunos, bem como na superação das dificuldades identificadas neste estudo.

Adicionalmente, recomenda-se que essas investigações considerem a integração de actividades práticas orientadas e de estratégias que promovam o estudo autónomo, de modo a reforçar a consolidação dos conteúdos fora do contexto da sala de aula. Estudos com esta abordagem poderão contribuir para o aperfeiçoamento das práticas pedagógicas no ensino da programação, particularmente no contexto do ensino técnico-profissional.

REFERÊNCIAS BIBLIOGRÁFICA

Bibliografia

Abesadze, S., & Nozadze, D. (Janeiro de 2020). Make 21st Century Education: The Importance of Teaching Programming in Schools. *International Journal of Learning and Teaching, 6*. doi: 10.18178/ijlt.6.3.158-163

Ahmed, S. K., Mohammed, R. A., Nashwan, A. J., Ibrahim, R. H., Abdalla, A. Q., Ameen, B. M., & Khdhir, R. M. (Agosto de 2025). Using thematic analysis in qualitative research. *ScienceDirect, 6*. doi:https://doi.org/10.1016/j.glmedi.2025.100198

Araújo, R. B. (2025). Integração das tecnologias digitais, o currículo escolar e a formação de professores: necessidades emergentes no cenário educacional. *Cuadernos De Educación Y Desarrollo, 17*, pp. 01-25. doi:10.55905/cuadv17n1-001

Bank, T. W. (s.d.). *Bons empregos para a juventude angolana: Oportunidades, Desafios e Orientações de Políticas.* World Bank Document. Fonte: https://documents1.worldbank.org/curated/en/099210502232360806/pdf/P174737020b1d101709def0fd1e971879f0.pdf

Belmar, H. (19 de Outubro de 2022). Review on the teaching of programming and computational thinking in the world. *Frontiers in Computer Science, 4*. doi:10.3389/fcomp.2022.997222

Blaszko, C. E., Claro, A. d., & Ujiie, N. T. (20 de Fevereiro de 2021). A contribuição das metodologias ativas para a prática pedagógica dos professores universitários. *Revista Eletrônica de Educação, 6*. doi:https://doi.org/10.25053/redufor.v6i2.3908

Cangondo, G., Pombo, N., Souza-Pereira, L., Ouhbi, S., & Silva, B. (2022). Computer Science Education in Angola: The Key Challenges. *Proceedings of the 2022 IEEE Global Engineering Education Conference (EDUCON).* Jemni, M.; Kallel, I.; Akkari, A. doi:10.1109/EDUCON52537.2022.9766392

Canhici, M. H. (2014). *Estudo sistemático de monografias dos finalistas do ISCED-Cabinda sobre dificuldades de aprendizagem (2006-2011).* Dissertação, Belo Horizonte.

Coelho, B. (21 de Abril de 2021). Amostragem: o que é e como fazer cálculo amostral. *Mettzer*.

Coelho, M. d., & Guedes, A. M. (04 de Abril de 2021). Aprendizagem Baseada em Problemas aplicada à Programação de Computadores: Um Mapeamento Sistemático. *18*. doi:https://doi.org/10.22456/1679-1916.110298

Costa, M. A., Soares, S. L., & Florêncio, T. S. (2022). Abordagem quantitativa em pesquisas educacionais: perspectivas no programa de pós-graduação da UFMG (2017-2019) . *DOXA: Revista Brasileira de Psicologia e Educação*, p. e022019.

Creswell, J. W. (2021). *Projeto de pesquisa: métodos qualitativo, quantitativo e misto (5ª ed.).* Porto Alegre: Penso.

da Silva , T. R., Medeiros , T. J., Medeiros , H., Lopes , R., & Aranha , E. (13 de Março de 2015). Ensino-aprendizagem de programação: uma revisão sistemática da literatura. *Revista Brasileira de Informática na Educação, 23*. doi:10.5753/RBIE.2015.23.01.182

Denzin, N. K., & Lincoln, Y. S. (2006). *O planejamento da pesquisa qualitativa: Teorias e abordagens* (2.ª ed.). Porto Alegre: Artmed.

Dombi, J., & Jónas, T. (2021). Likert scale-based evalutions with flexible fuzzy numbers. Em J. Dombi, *Advances in the Theory of Probabilistic and Fuzzy Data* (pp. 167-187). Cham: Springer.

Downey, R. J., Youmans, K., Villanueva Alarcón, I., Nadelson, L., & Bouwma-Gearhart, J. (22 de Outubro de 2022). Building Knowledge Structures in Context: An Exploration of How Constructionism Principles Influence Engineering Student Learning Experiences in Academic Making Spaces. *Education Sciences, 12*. doi: https://doi.org/10.3390/educsci12110733

Farias, G. B. (Abril-junho de 2022). Contributos da aprendizagem significativa de David Ausubel para o desenvolvimento da competência em informação. *Perspectivas em Ciência da Informação, 27*. doi:https://doi.org/10.1590/1981-5344/39999

Gil, A. C. (2002). *Métodos e técnicas de Pesquisa Social.* São Paulo: Altas.

Gil, A. C. (2008). *Como elaborar projetos de pesquisa.* São Paulo: Atlas.

Gomes, e. a. (Julho de 2008). Aprendizagem de programação de computadores: Dificuldades e ferramentas de suporte. *Revista Portuguesa de Pedagogia*, pp. 161-179.

Guler, G., & Ayan, C. (2020). To be direct or not: Reversing Likert response format items. *The Spanish Journal of Psychology*, p. e24.

INQ, I. N. (2021). *Estudo Especializado sobre o Mercado de Trabalho e Actividades Económicas.* Instituto Nacional de Qualificações - Angola. Fonte: https://inq.gov.ao/pt/servicos/estudo-mercado-de-trabalho-relactorio-final-vf.pdf

Koblyakov, P. (22 de Abril de 2025). *What is an Open Source LMS?* Fonte: Raccoon Gang: https://raccoongang.com/blog/open-source-lms-everything-you-need-know/

Kwon, K., Ottenbreit-Leftwich,, A. T., Brush, T. A., Jeon, M., & Yan, G. (05 de Agosto de 2021). Integration of problem-based learning in elementary computer science education: effects on computational thinking and attitudes. *Educational Technology Research and Development, 69*, pp. 2761-2787. doi: 10.1007/s11423-021-10034-3

Lahtinen , E., Ala-Mutka, K., & Jarvinen, H.-M. (2005). A Study of the Difficulties of Novice Programmers. *ACM SIGCSE*, pp. 14-18.

Lakatos, E. M. (2017). *Metodologia do trabalho Cientifico.* São Paulo.

Leite, L. R., Verde, A. P., Oliveira, F. d., & Nunes, J. B. (2021). Abordagem mista em teses de um programa de pós-graduação em educação: análise à luz de Creswell. *Scielo Brasil, 47*. doi:https://doi.org/10.1590/S1678-4634202147243789

Lins, A. S., & Mitzko, T. (2023). A UTILIZAÇÃO DA FERRAMENTA GOOGLE FORMS COMO INSTRUMENTO DE MELHORIA NAS ANÁLISES CRÍTICAS DOS KPI'S NA SANEPAR. *Anais 32.º Congresso Brasileiro de Engenharia Sanitária e Ambiental.* Fonte: https//abes-dn.org.br/anaiseletronicos/32cbesa/183_tema_v.pdf

Lorenzini, F., Costa, S., & Almeida, P. (2024). Investigação mista em educação: estratégias e aplicações em contextos escolares. *Revista Portuguesa de Eduacação*, pp. 763-779.

Maia, P. C., Matias, D. G., Cândido, E. F., Bastos, J. L., Couto, M. B., de Oliveira, M. M., . . . Mesquita, S. G. (14 de Abril de 2025). Ensino de programação com abordagens ativas: estratégias e resultados. *Cuadernos De Educación Y Desarrollo, 17*. doi:DOI: 10.55905/cuadv17n4-072

Marín-Marín, J.-A., Gárcia-Tudela, P. A., & Duo-Terrón, P. (30 de Abril de 2024). Computational thinking and programming with Arduino in education: A systematic review for secondary education. *Heliyon, 10*. doi:https://doi.org/10.1016/j.heliyon.2024.e29177

Massa, N. P., Oliveira, G. S., & Santos, J. A. (21 de Setembro de 2022). O construcionismo de Seymour Papert e os computadores na educação. *Cadernos da FUCAMP*. Fonte: https://revistas.fucamp.edu.br/index.php/cadernos/article/view/2820

MESCTI (Ministério do Ensino Superior, Ciências, Tecnologias e Inovação). (2023). *Ensino Superior e Tecnologia em Angola: Um cenário de desafios e oportunidades.* Fonte: https://www.researchgate.net/publication/370462734_ENSINO_SUPERIOR_E_TECNOLOGIA_EM_ANGOLA_UM_CENARIO_DE_DESAFIOS_E_OPORTUNIDADES

Mussaque, A. J. (Setembro de 2024). Atenção escolar às dificuldades de aprendizagem em Angola: Estratégias de interveção. *Recima21-Revista Científica Multidisciplinar ISSN 2675-6218*. doi:https://doi.org/10.47820/recima21.v5i9.5680

Ngadengon, Z., Subramaniam, T. S., Yasak, Z., Ideris, N. A., & Ramly, Z. M. (07 de Fevereiro de 2025). Evaluating the Difficulties in Programming. *International Journal of Academic Research in Progressive Education and Development, 14*, pp. 1051-1065. doi: http://dx.doi.org/10.6007/IJARPED/v14-i1/24518

Nouri, J., Zhang, L., Mannila, L., & Norén, E. (2020). Development of computational thinking, digital competence and 21st century skills when learning programming in K-9. *Education Inquiry, 11*. doi:10.1080/20004508.2019.1627844

Nunes, C. M. (2023). *ENSINO DE LÓGICA DE PROGRAMAÇÃO: dificuldades de aprendizagem na percepção de professores do ensino superior.* Dissertação, Belo Horizonte.

Papert, S. (1985). *Mindstorms: Crianças, computadores e ideias poderosas* (1.ª (Tradução brasileira da obra original de 1980) ed.). Porto Alegre: Artmed.

Pilonetto, I. A., Paz, D. P., & Rodrigues, L. (03 de Março de 2019). CONECTIVISMO: APRENDENDO A PARTIR DAS CONEXÕES. *Revista Mundi, 4*. doi:https://doi.org/10.21575/25254782rmetg2019vol4n1734

(2017). *PLANO NACIONAL DE DESENVOLVIMENTO DA - "EDUCAR - ANGOLA 2030".* Fonte: https://planipolis.iiep.unesco.org/sites/default/files/ressources/angola-educar_2030.pdf

Ramalho, I. d., Araújo, C. L., & Resende, V. d. (19 de 12 de 2021). Contribuições do uso do software NVivo em pesquisa discursiva crítica. *Cadernos de Linguagem e Sociedade, 22*. doi:https://doi.org/10.26512/les.v22i2.34038

Ramírez-Echeverry, J. J., Restrepo-Calle, F., & Jiménez, S. T. (21 de Fevereiro de 2025). Self-Regulated Learning Strategies in Computer Programming. *European Journal of Education and Pedagogy, 60*. doi:https://doi.org/10.1111/ejed.70052

RETEP. (2011). *Dossiês de curso Formação Técnica, Tecnológica e Prática.*

Rostirola, S. C. (14 de Janeiro de 2020). Conectivismo Pedagógico: novas formas de ensinar e aprender no século XXI. *Revista Thema, 16*. doi:https://doi.org/10.15536/thema.V16.2019.1012-1025.1583

Sampaio, R. C., & Lycarião, D. (2021). *Análise de conteúdo categorial: Manual de Aplicação.* Brasília: Enap.

Sampaio, R. C., & Sampaio, R. C. (2021). *Análise de conteúdo categorial: manual de aplicação.* Brasília: Escola Nacional de Administração Pública (Enap).

Santos, S. M., Souza, D. C., Cruz, E. C., Melo Júnior, H. G., Venturim, I. d., Silva, L. J., . . . Silva Neto, R. C. (2024). TRANSFORMAÇÃO DO CURRÍCULO ESCOLAR: A INTEGRAÇÃO DE PROGRAMAÇÃO E ROBÓTICA NO PROCESSO DE ENSINO-APRENDIZAGEM. *Revista Contemporânea, 4*. doi:https://doi.org/10.56083/RCV4N4-080

Santos, S. V., & Baptista, M. C. (21 de 03 de 2023). Triangulação metodológica em pesquisas etnográficas com e sobre crianças. *Revista Diálogo Educacional, 23*, pp. 201-229. doi:https://doi.org/10.7213/1981-416X.23.076.DS08

Scarpati, E. D. (2023). *UMA PROPOSTA DE AVALIAÇÃO DIAGNÓSTICA MULTIDIMENSIONAL DA APRENDIZAGEM DE PROGRAMAÇÃO NA FORMAÇÃO PROFISSIONAL.* Dissertação, Vitória.

Silva, F. L., & Moreira, I. A. (2021). Análise das dificuldades na aprendizagem de programação no curso de análise e desenvolvimento de sistemas do IFRN/Pau dos Ferros. *8º Encontro Nacional de Computação dos Institutos Federais (ENCOMPIF)* (pp. 1-8). Sociedade Brasileira de Computação. doi:https://doi.org/10.5753/enucompi.2021.17752

Souza, L. H., & Kerbauy, M. L. (2017). Técnicas de análise de dados qualitativos na educação. *Revista Brasileira de Estudos Pedagógicos*, pp. 217-235.

VALLE, P. R., & FERREIRA, J. D. (31 de Janeiro de 2025). ANÁLISE DE CONTEÚDO NA PERSPECTIVA DE BARDIN: CONTRIBUIÇÕES E LIMITAÇÕES PARA A PESQUISA QUALITATIVA EM EDUCAÇÃO. *Scielo Brasil*. doi:https://doi.org/10.1590/0102-469849377

Victor, E. L. (2011). *Impacto do Uso Das Tecnologias de Informação e Comunicação Tic Face ao Processo de Ensino/Aprendizagem Em Angola.* Dissertação, Évora.

Vidal-Silva, C., Rodas-Silva, J., Vinueza-Morales, M., Córdova-Morán, j., & Cevallos-Ayón, E. (28 de Março de 2025). Teaching programming in higher education: a bibliometric analysis of trends, technologies, and pedagogical approaches. *Frontiers in Education, 10*. doi:https://doi.org/10.3389/feduc.2025.1525917

Vygotsky, L. S. (1979). *A formação social da mente: O desenvolvimento dos processos psicológicos superiores* (4.ª ed.). São Paulo: Martins Fontes.

Waite, J., & Sentance, S. (2021). *Teaching programming in schools: A review of approaches and strategies.* Cambridge, Reino Unido: Instituição/Editora:. Fonte: https://www.raspberrypi.org/app/uploads/2021/11/Teaching-programming-in-schools-pedagogy-review-Raspberry-Pi-Foundation.pdf

Welborn, L., Cilliers, J., & Kwasi, S. (2020). *Cenários do Futuro de Angola 2050.* University of Denver - Korbel School of International Studies. Fonte: https://korbel.du.edu/wp-content/uploads/2025/04/Cenarios-do-Futuro-de-Angola-2050.pdf

Yan, Y.-M., Chen, C.-Q., Hu, Y.-B., & Ye, X.-D. (07 de Fevereiro de 2025). LLM-based collaborative programming: impact on students' computational thinking and self-efficacy. *Humanities and Social Sciences Communications, 12*. doi:https://doi.org/10.1057/s41599-025-04471-1

Zanetti, H. A., Borges, M. A., & Ricarte, I. L. (2022). A teoria de aprendizagem significativa no ensino de programação: um mapeamento sistemático da literatura. *Anais do XXXIII Simpósio Brasileiro de Informática na Educação (SBIE)* (pp. 1-14). Sociedade Brasileira de Computação (SBC). doi:https://doi.org/10.5753/sbie.2022.224579

APÊNDICE

APÊNDICIE

Apêndice 1-*Questionário: Dificuldades na Aprendizagem de Programação (Baseado no CFCP)*

Este questionário tem como objetivo identificar as dificuldades enfrentadas pelos alunos na aprendizagem de programação. As respostas devem ser dadas de acordo com a escala abaixo: (1) Nunca (2) Raramente (3) Às vezes (4) Frequentemente (5) Sempre

OBS: Assinale uma única opção para cada questão.

Dados pessoais

Idade _____

Sexo: Feminino ( ) Masculino ( )

Já teve contacto prévio com programação antes de ingressar no curso?

Sim ( ) Não ( )

Dimensão 1 – Dificuldades Sintáticas

Tenho dificuldade em lembrar a sintaxe correta dos comandos da linguagem de programação.

Nunca ( ) Raramente ( ) Às vezes ( ) Frequentemente ( ) Sempre ( )

Esqueço com frequência símbolos como ponto e vírgula, parênteses ou chaves.

Nunca ( ) Raramente ( ) Às vezes ( ) Frequentemente ( ) Sempre ( )

Perco muito tempo tentando corrigir erros de sintaxe.

Nunca ( ) Raramente ( ) Às vezes ( ) Frequentemente ( ) Sempre ( )

Dimensão 2 – Dificuldades Semânticas

Mesmo sem erros de sintaxe, muitas vezes o programa não funciona como eu esperava.

Nunca ( ) Raramente ( ) Às vezes ( ) Frequentemente ( ) Sempre ( )

Tenho dificuldade em entender o que um trecho de código realmente faz.

Nunca ( ) Raramente ( ) Às vezes ( ) Frequentemente ( ) Sempre ( )

Sinto dificuldade em prever a saída (resultado) de um programa.

Nunca ( ) Raramente ( ) Às vezes ( ) Frequentemente ( ) Sempre ( )

Quando penso em estruturas como if, for ou while, sinto dificuldade em imaginar como funcionam na prática.

Nunca ( ) Raramente ( ) Às vezes ( ) Frequentemente ( ) Sempre ( )

Dimensão 3 – Dificuldades Pragmáticas

Fico indeciso sobre qual estrutura (if, while, for, etc.) devo usar em determinados problemas.

Nunca ( ) Raramente ( ) Às vezes ( ) Frequentemente ( ) Sempre ( )

Tenho dificuldade em aplicar conceitos de programação em situações práticas.

Nunca ( ) Raramente ( ) Às vezes ( ) Frequentemente ( ) Sempre ( )

Não consigo transformar um problema real em código de forma clara.

Nunca ( ) Raramente ( ) Às vezes ( ) Frequentemente ( ) Sempre ( )

Gostaria que fossem utilizadas mais atividades práticas e inovadoras no ensino de programação (ex.: jogos, projetos, exercícios criativos).

Nunca ( ) Raramente ( ) Às vezes ( ) Frequentemente ( ) Sempre ( )

Atividades inovadoras (como desafios em grupo ou competições de código) ajudam-me a aprender melhor.

Nunca ( ) Raramente ( ) Às vezes ( ) Frequentemente ( ) Sempre ( )

Gostaria que fossem dadas mais oportunidades de prática livre em programação (fora de testes e avaliações).

Nunca ( ) Raramente ( ) Às vezes ( ) Frequentemente ( ) Sempre ( )

Dimensão 4 – Dificuldades Metacognitivas

Quando meu código não funciona, tenho dificuldade em identificar onde está o erro.

Nunca ( ) Raramente ( ) Às vezes ( ) Frequentemente ( ) Sempre ( )

Tenho dificuldade em organizar meu estudo para aprender programação sozinho(a).

Nunca ( ) Raramente ( ) Às vezes ( ) Frequentemente ( ) Sempre ( )

Estratégias de aprendizagem autorregulada (ex.: fazer resumos, rever códigos sozinho, autoexplicação) ajudam na minha compreensão em programação.

Nunca ( ) Raramente ( ) Às vezes ( ) Frequentemente ( ) Sempre ( )

Dimensão 5 – Dificuldades com Ferramentas/IDE

Tenho dificuldade em utilizar o ambiente de programação (IDE).

Nunca ( ) Raramente ( ) Às vezes ( ) Frequentemente ( ) Sempre ( )

Não compreendo bem as mensagens de erro que aparecem no compilador.

Nunca ( ) Raramente ( ) Às vezes ( ) Frequentemente ( ) Sempre ( )

Perco muito tempo configurando ou ajustando o ambiente de programação.

Nunca ( ) Raramente ( ) Às vezes ( ) Frequentemente ( ) Sempre ( )

Gostaria que o professor ensinasse programação com ferramentas visuais, como o Scratch.

Nunca ( ) Raramente ( ) Às vezes ( ) Frequentemente ( ) Sempre ( )

Já ouvi falar ou utilizei o Scratch em alguma atividade de programação.

Nunca ( ) Raramente ( ) Às vezes ( ) Frequentemente ( ) Sempre ( )

Dimensão 6 – Fatores Motivacionais

1. Acredito que a minha motivação influencia no meu desempenho em programação.

Nunca ( ) Raramente ( ) Às vezes ( ) Frequentemente ( ) Sempre ( )

2. Quando consigo resolver um problema de programação, sinto-me mais motivado a continuar aprendendo.

Nunca ( ) Raramente ( ) Às vezes ( ) Frequentemente ( ) Sempre ( )

3. Sinto que programação é uma disciplina muito difícil para mim.

Nunca ( ) Raramente ( ) Às vezes ( ) Frequentemente ( ) Sempre ( )

4. Quando encontro dificuldades, tenho vontade de desistir facilmente.

Nunca ( ) Raramente ( ) Às vezes ( ) Frequentemente ( ) Sempre ( )

5. Acredito que sou capaz de aprender programação com esforço e prática.

Nunca ( ) Raramente ( ) Às vezes ( ) Frequentemente ( ) Sempre ( )

Apêndice 2-*Questionário para os docentes*

1. Na sua experiência, quais são as principais dificuldades que os alunos da 10ª classe enfrentam na aprendizagem da programação?

2. Que factores considera mais determinantes para essas dificuldades?

3. De que forma a ausência ou presença de contacto prévio com programação influencia o desempenho dos alunos nas aulas?

4. Quais são as dificuldades que os alunos não conseguem verbalizar, mas que o professor observa claramente durante as aulas práticas?

ANEXO

ANEXO

Tabela 24- Grelha Curricular

Disciplinas	**Horas curriculares semenais**		
	10ª classe	**11ª classe**	**12ª classe**
Componentes Sóciocultural			
Português	3	3	-
Inglês/Francês	3	3	-
Formação de Atitudes Integradoras	2	2	-
Educação Física	2	2	-
Subtotal	**10**	**10**	-
Componente Científica			
Matemática	5	4	5
Organização e Administração de Empresas	3	4	5
Direito	-	-	3
Subtotal	**8**	**8**	13
Componente Técnica, Tecnológica e Pratica			
Técnica e Linguagem de Programação	6	4	-
Sistema de Informação e Comunicação	-	-	4
Base de Dados e Redes de Computadores	4	3	-
Tecnologias de Informação e Comunicação	4	4	3
Informática Aplicada à Gestão	-	3	6
Projectos Tecnológicos	-	-	6P(+6P)
Subtotal	**14**	**14**	**19(+6)**
TOTAL	**32**	**32**	**32(+6)**