Misconceptions em programação introdutória: existe associação com perfil sociodemográfico?

Victor Araújo Passos, Airton Nascimento Silveira Filho, Fabíola Guerra Nakamura, Elaine Harada Teixeira Oliveira, David Fernandes Oliveira, Leandro Silva Galvão Carvalho

[1]Instituto de Computação – Universidade Federal do Amazonas (UFAM)
Av. Gen. Rodrigo Octávio, 6200, Coroado I – 69080-900 – Manaus – AM – Brasil

{victor.passos,airton.filho,fabiola,elaine,david,galvao}@icomp.ufam.edu.br

Resumo. *Compreender os desafios enfrentados por estudantes iniciantes em programação é essencial para aprimorar o ensino na área. Este estudo investiga a relação entre* misconceptions *em programação e perfis sociodemográficos de estudantes universitários matriculados em cursos fora da área de Computação. Utilizando dados de submissões em juízes online, foram analisados padrões de código e suas associações com indicadores sociodemográficos, tais como: curso de origem, experiência prévia e desempenho acadêmico. Os resultados apontam diferenças significativas preliminares na ocorrência de* misconceptions *entre os perfis. Esses achados, ainda que limitados, reforçam a importância de estratégias pedagógicas adaptadas a perfis de estudantes, mediante validação em estudos futuros com amostras maiores e métodos automatizados de análise.*

Abstract. *Understanding the challenges faced by novice programming students is crucial to improving education in the field. This study examines the relationship between programming misconceptions and the sociodemographic profiles of university students whose courses are not directly related to the field of computing. Using data from online judges, code patterns and their associations with variables such as students' academic background, prior experience, and performance were analyzed. The results indicate preliminary significant differences in the occurrence of* misconceptions *among different profiles. These findings, though limited, highlight the importance of tailored pedagogical strategies for student profiles, pending validation in future studies with larger samples and automated analysis methods.*

1. Introdução

A aprendizagem de programação pode ser bastante desafiadora para alunos iniciantes em um curso de graduação, principalmente para aqueles fora da área de Computação (chamados de *non-majors* na literatura) [Watson and Li 2014]. Dentre os obstáculos encontrados durante o aprendizado, os *misconceptions*, ou seja, ideias imperfeitas sobre conceitos de um determinado conteúdo, são um dos mais problemáticos no desenrolar de uma disciplina [Lewis et al. 2019, Sorva 2023].

No contexto de Introdução à Programação, um exemplo de *misconception* é declarar variáveis que nunca são usadas no programa, ou testar condições mutualmente exclusivas em blocos aninhados. Porém, os *misconceptions* se manifestam de diversas formas

não tão fáceis de se notar, como, por exemplo: falta de entendimento da semântica, dificuldades com conceitos básicos, interpretação incorreta do problema e implementação errada de algoritmos [Ettles et al. 2018].

Além disso, é importante não confundir o termo *misconception* com *mistake*, ou seja, pequenos erros ou deslizes causados por desatenção durante a escrita do código, como a falta de um ponto e vírgula ou o não fechamento de parênteses [Bosse et al. 2021]. Se não forem corrigidos, os *misconceptions* podem levar a erros (*mistakes*) frequentes, frustração e desmotivação no aprendizado [Henley et al. 2021]. Ou seja, nem todo *misconception* leva necessariamente a um *mistake*: lapsos como esquecer de fechar um parêntese indicam apenas uma distração, mas não necessariamente uma ideia imperfeita de como a sintaxe de uma linguagem funciona. Além disso, para fins de clareza e em consonância com a literatura especializada, optamos por manter o termo misconceptions em inglês ao longo do texto. Por exemplo, em vez de traduzir como "conceitos errôneos" ou "ideias equivocadas" — expressões que podem não capturar com precisão seu significado técnico — preferimos preservar o termo original, amplamente utilizado em pesquisas internacionais sobre ensino de programação.

Trabalhos como o de [Sorva 2023] catalogaram *misconceptions* gerais e específicos na linguagem Java, enquanto [Caceffo et al. 2017] e [Gama et al. 2018] os exploraram em C e Python. Em outra vertente, [Araújo et al. 2021] utilizaram dados de juízes online para ampliar essas análises de códigos em Python e [Silva et al. 2023a] aplicaram aprendizado de máquina para identificar compreensões imperfeitas a partir de códigos corretos, criando o catálogo mais amplo até o momento. Entretanto, esses estudos de catalogação representam um passo inicial na compreensão dos *misconceptions*.

Dada a heterogeneidade de formação dos alunos que ingressam no ensino superior, é preciso identificar se os *misconceptions* encontrados são característicos ou não de determinados perfis de estudantes, a fim de planejamento de estratégias realmente eficazes para uma aprendizagem mais significativa em programação. Neste estudo, procuraremos dar continuidade aos trabalhos de [Silva et al. 2023a] e [Araújo et al. 2021], a partir das seguintes questões de pesquisa:

QP1: Quais são os *misconceptions* mais frequentes nas turmas investigadas?
QP2: Há relação entre os tipos de *misconceptions* mais frequentes e o perfil sociodemográfico dos alunos?

Para responder essas perguntas, foi usado o juiz online *CodeBench* e a disciplina de IPC, ambos da UFAM, como estudo de caso. Além de identificar os *misconceptions* dos alunos, adicionamos um novo elemento a ser investigado: testes de associação da ocorrência de *misconceptions* e o perfil sociodemográfico dos estudantes, que inclui dados autodeclarados como tipo de ensino médio cursado, nível de inglês, experiência prévia em programação, etc., e pode ser extraído do Questionário Demográfico aplicado no início da disciplina. Os códigos e respostas do questionário estão disponíveis de forma anonimizada na base de dados do *CodeBench*[1].

Este artigo está organizado da seguinte forma: a Seção 2 contextualiza a revisão da literatura. A Seção 3 detalha a metodologia adotada, incluindo a coleta de dados por meio do juiz online CodeBench, classificação de *misconceptions* baseada em

[1]`https://codebench.icomp.ufam.edu.br/dataset/`

[Silva et al. 2023a] e os testes estatísticos aplicados para verificar a associação entre variáveis. Na Seção 4, são apresentados os resultados, com ênfase nas relações significativas entre perfis sociodemográficos e padrões de ideias imperfeitas. As limitações metodológicas e potenciais vieses são discutidos na Seção 5, enquanto a Seção 6 conclui o estudo.

2. Trabalhos Relacionados

A identificação de equívocos de aprendizado tem se tornado um tópico relevante em pesquisas conduzidas em diferentes cenários acadêmicos. Trabalhos como o de [Qian and Lehman 2017] buscam definir o que são *misconceptions* na área da programação. Estudos nessa área são importantes, pois permitem a identificação desses equívocos, contribuindo para avanços pedagógicos.

O trabalho de [Souza et al. 2016] apresentou um mapeamento sistemático da literatura, que levantou os principais desafios no ensino e aprendizagem sobre programação. O estudo concluiu que os principais problemas de aprendizagem dos alunos envolviam dificuldade de aplicação dos conhecimentos, falta de motivação e dificuldade de entendimento dos conceitos dados em aula.

[Gama et al. 2018] identificaram *misconceptions* em disciplinas introdutórias de programação ministradas na linguagem Python, por meio da análise de códigos desenvolvidos por alunos, com o objetivo de identificar problemas recorrentes relacionados à manipulação de variáveis, parâmetros de função e estruturas de código. Após a coleta desses dados, eles criaram um catálogo de 28 categorias de *misconceptions*, agrupadas com base nos conteúdos da disciplina, para classificar todos os equívocos identificados.

Complementar a isso, [Araújo et al. 2021] teve como diferencial o uso da base dados do juiz online *CodeBench* como caso de estudo para a análise de *misconceptions*, tendo como referência o catálogo de [Gama et al. 2018]. O estudo foi aplicado a uma amostra composta por dados coletados de 1068 códigos desenvolvidos em Python por 81 alunos. Como resultado, foi criado um catálogo com 27 *misconceptions*, dos quais 19 estavam presentes em ambos os trabalhos.

Por fim, [Silva et al. 2023a] desenvolveram um novo catálogo a partir exclusivamente em códigos Python corretos, isto é, passaram em todos os casos de teste das respectivas questões cadastradas em um juiz online. Para aprimorar a identificação dos *misconceptions*, aplicaram técnicas de aprendizado de máquina, resultando em um conjunto final de 45 categorias. Esse catálogo foi adotado como referência desta pesquisa por ser a abordagem mais atual e abrangente disponível no início do estudo.

No contexto de classes sociodemográficas, a pesquisa de [Lopes 2017] buscou compreender em quais áreas de estudo estudantes que ingressaram por meio de ações afirmativas se graduaram em universidades públicas brasileiras nos anos de 2009 e 2010. Para isso ela analisou o banco de dados do Exame Nacional de Desempenho do Estudante (Enade). A conclusão do artigo é que estudantes de ações afirmativas de universidades públicas tendem a se graduar em áreas de estudo menos prestigiadas. Essa entrada em áreas com menor prestígio pode levar a oportunidades ocupacionais menos relevantes e menores rendimentos no mercado de trabalho, indicando que a educação ainda reproduz desigualdades sociais no Brasil, apesar das políticas de ação afirmativa.

Por último, o estudo de [Casiraghi et al. 2020] avaliou os fatores relacionados ao rendimento acadêmico (RA) de estudantes do Ensino Superior em uma instituição privada do interior do Rio de Janeiro e relacionou a autopercepção do RA com o coeficiente de rendimento acadêmico. A metodologia consistiu em um estudo censitário realizado em 2019 com 264 estudantes de 19 cursos, utilizando um formulário eletrônico para coleta de dados e os registros de RA da instituição. A análise estatística envolveu a Análise de Variância (ANOVA). A principal conclusão aponta que o RA variou significativamente em relação às áreas de conhecimento, sexo, faixa etária e autopercepção do rendimento. Embora a escolaridade dos pais não tenha apresentado relação significativa de forma geral, a escolaridade materna a partir de oito anos de estudo mostrou um efeito positivo no RA. A renda familiar, no contexto dessa instituição privada, não se mostrou um fator significativo para o RA.

Dessa forma, ampliando a contribuição dos trabalhos citados, o estudo analisou os misconceptions em programação introdutória, relacionando-os com o perfil sociodemográfico dos alunos. Salienta-se que o levantamento realizado neste trabalho deriva da pesquisa de [Silveira Filho et al. 2025], que realizou um levantamento longitudinal do surgimento e evolução de *misconceptions*.

3. Metodologia

Dando continuidade ao levantamento longitudinal de [Silveira Filho et al. 2025], este estudo analisou códigos produzidos por alunos na resolução de avaliações em uma disciplina de introdução à programação da UFAM, ofertada em 2023. Os códigos e os dados demográficos estão armazenados de forma anonimizada no juiz online CodeBench[2], cuja escolha se justifica por sua adoção sistemática e central no cotidiano da disciplina desde 2013, permitindo a coleta estruturada e automatizada de dados relevantes sobre o desempenho e os padrões de tentativa dos estudantes. A disciplina foi estruturada em seis módulos quinzenais, cada um composto por uma aula conceitual, duas aulas práticas para resolução de listas de exercícios e uma avaliação parcial. Todas as atividades ocorreram em laboratório de informática, com uso intensivo do CodeBench como ambiente principal para experimentação de conceitos, resolução de problemas e realização de avaliações. Os temas abordados em cada módulo estão descritos na Tabela 1.

Tabela 1. Conteúdos ministrados ao longo dos seis módulos da disciplina

Módulo	Conteúdo
M1	Programação Sequencial
M2	Decisão Simples
M3	Decisão aninhada
M4	Repetição por condição
M5	Vetores e Strings
M6	Repetição por contagem

Para este estudo, foram selecionadas duas turmas com desempenhos contrastantes: a de melhor desempenho (Engenharia Mecânica) e a de pior desempenho (Licenciatura em Matemática). A escolha se baseou na expressiva diferença entre o desempenho médio

[2]`https://codebench.icomp.ufam.edu.br/dataset/`

dos alunos de cada turma. A análise concentrou-se apenas nas questões das avaliações presenciais, realizadas em ambiente monitorado por um professor e um monitor, para limitar comportamentos desonestos. Tal recorte possibilita maior probabilidade de que cada amostra de código-solução seja única.

3.1. Coleta e tratamento de dados

Para coletar os códigos dos alunos, utilizamos um *script* em Python adaptado de [Lima et al. 2021][3]. Os dados extraídos continham apenas os identificadores (IDs) dos estudantes, assegurando o anonimato. A filtragem por curso foi feita manualmente: cada arquivo foi verificado individualmente e, se pertencesse a um dos cursos de interesse, seu ID era registrado para análise. Esse processo resultou em uma amostra composta por 83 estudantes: 41 da Engenharia Mecânica e 42 da Licenciatura em Matemática.

3.2. Extração e análise dos códigos dos alunos

O *script* de extração organizou os *logs* do período letivo 2023/1 armazenados no Codebench. Os dados foram estruturados em pastas separadas por turmas, usuários, dados socio-demográficos e resultados de avaliações (incluindo notas e códigos *Python*). Os 83 estudantes analisados produziram, ao todo, 498 códigos distintos em resposta às questões de avaliação. A análise considerou a última versão submetida ao juiz online. Quando não havia submissão formal, considerou-se a última versão salva no sistema.

Para garantir maior confiabilidade na identificação de *misconceptions*, os dois primeiros autores deste estudo realizaram inicialmente uma etapa de calibração: analisaram separadamente 10 códigos em comum e, em seguida, discutiram os resultados para alinhar critérios e metodologia. Após essa etapa, cada pesquisador ficou responsável pela análise dos códigos de um dos cursos.

Tanto códigos aceitos quanto rejeitados pelo CodeBench foram incluídos na análise, com o objetivo de capturar misconceptions conceituais que podem estar presentes inclusive em soluções corretas do ponto de vista sintático e funcional, mas que revelam falhas estruturais ou lógicas. Essa abordagem ampliou a capacidade de detectar equívocos que poderiam passar despercebidos em análises restritas apenas a erros evidentes.

3.3. Classificação dos *misconceptions*

Os *misconceptions* identificados foram classificados com base na tipologia proposta por [Silva et al. 2023a], que os organiza conforme os principais temas da disciplina, como estruturas de decisão e repetição. Para respostas incorretas, foi avaliado se o erro refletia de fato um *misconception* ou se era fruto de distração ou erro acidental. Já nas respostas corretas, analisou-se a lógica interna dos códigos para identificar possíveis equívocos conceituais que, embora não resultassem em falhas nos testes, comprometiam a qualidade da solução. Todas as ocorrências foram então categorizadas conforme os grupos definidos por [Silva et al. 2023a].

3.4. Coleta e análise dos dados sociodemográficos

As informações sociodemográficas foram obtidas a partir de um questionário aplicado no início do semestre por meio do próprio CodeBench. A Tabela 2 apresenta as variáveis

[3] `https://anonymous.4open.science/r/questions-irt-calculator-D6DE/README.md`

selecionadas para análise, como tipo de ensino médio cursado, modalidade de ingresso na universidade, escolaridade dos pais, entre outros.

Tabela 2. Indicadores sociodemográficos retirados do questionário

Pergunta	**Opções de Resposta**
Tipo de Ensino Médio cursado	Público, Técnico, Particular
Modalidade de ingresso na universidade	SISU/ENEM, Processo Seletivo Contínuo (PSC), Processo Seletivo Extramacro (PSE)
Tipo de cota	Ampla concorrência, Cota por renda, Cota independente de renda
Motivo da escolha do curso	Afinidade/Vocação, Mercado de Trabalho, Influência Familiar, Prestígio Social, Baixa concorrência
Curso era a primeira opção do aluno	Sim, Não
Escolaridade dos pais	Nenhum possui graduação, Apenas mãe, Apenas pai, Ambos possuem, Não se sabe
Proficiência em leitura em inglês	Fluente, Razoável, Pouco, Não lê
Possui computador pessoal em casa	Sim, Não
Uso exclusivo do computador	Exclusivo, Não Exclusivo
Experiência prévia com programação	Sim, Não
Experiência prévia de estágio/trabalho	Sim, Não
Formação acadêmica anterior	Sim, Não

Para investigar possíveis associações entre os indicadores sociodemográficos e a ocorrência de *misconceptions*, foram seguidas as etapas abaixo:

1. Para cada indicador sociodemográfico, identificamos as classes disponíveis (opções de resposta).
2. Para cada ID de aluno pertencente a cada classe do indicador, contamos o número de *misconceptions* cometidos para cada uma das seis categorias mais recorrentes de *misconceptions* em toda a amostra das duas turmas juntas (Tabela 3).
3. Para verificar se havia diferença significativa no número de *misconceptions* cometidos entre as classe de um indicador sociodemográfico, utilizamos os testes estatísticos não-paramétricos de Mann-Whitney U (perguntas com duas opções de resposta) e Kruskal-Wallis (perguntas com três ou mais opções de resposta), com nível de significância de $\alpha = 95\%$.

Assim, para uma determinada categoria de *misconception* e um determinado indicador sociodemográfico com k classes ($k \geq 2$), foi testada a seguinte hipótese nula:

$$H_0 : \mu_1 = \mu_2 = \ldots = \mu_k$$

ou seja, não há diferença significativa entre as médias de *misconceptions* cometidos pelos estudantes de diferentes k classes do indicador sociodemográfico.

Os testes de Mann-Whitney U e Kruskal-Wallis podem ser empregados para verificar uma associação entre uma variável contínua (nº de ocorrências de *misconceptions* de um tipo X) e uma categórica (classes de um indicador sociodemográfico Y). Porém, não mensuram a intensidade dessa associação em uma escala padronizada (por exemplo, -1 a $+1$), como fariam coeficientes de correlação (ex.: Pearson ou Spearman).

4. Resultados e discussões

4.1. Categorias de *misconceptions* encontradas

Nesta pesquisa, os *misconceptions* foram identificados nos seguintes contextos:

- **Submissões corretas:** códigos que passaram em todos os casos de teste do juiz online, mas apresentaram falhas na lógica ou estrutura, indicando algum *misconception*.
- **Submissões incorretas:** códigos que não passaram nos casos de teste do juiz online devido a erros conceituais ou de implementação.
- **Códigos não submetidos:** códigos que não foram enviados para avaliação no juiz online, mas apresentaram *misconceptions* durante testes do aluno.

Além disso, alguns códigos foram excluídos da análise por não conterem informações suficientes para o estudo, tais como:

- **Códigos incompletos:** rascunhos com poucas linhas, como declarações isoladas de variáveis, sem estrutura executável.
- **Erros de interpretação do enunciado:** casos em que os erros estavam ligados a falhas na compreensão do problema, em vez de *misconceptions* de programação.

Durante a análise, cada *misconception* identificado foi classificado conforme as 45 categorias levantadas por [Silva et al. 2023a]. As mais frequentes encontradas na amostra estão listadas na Tabela 3. A listagem completa das categorias, bem como sua definição, pode ser encontrada em [Silva et al. 2023b] (p.125).

Tabela 3. Descrição dos *misconceptions* mais frequentes, conforme nomenclatura proposta por [Silva et al. 2023a]

ID	Nome
A1	Variável não utilizada
A2	Variável atribuída a si mesma
A5	Importação não utilizada
B4	Comandos repetidos dentro de blocos *if-elif-else*
B8	Não utilização da declaração *elif/else*
C3	Operações redundantes dentro de laços

A Figura 1 apresenta os *misconceptions* mais frequentes nos dois cursos, com destaque para estruturas de decisão, como repetições desnecessárias em blocos `if-else` (B4) e importações não utilizadas (A5). Este estudo também identificou mais categorias do que o trabalho anterior de [Silveira Filho et al. 2025].

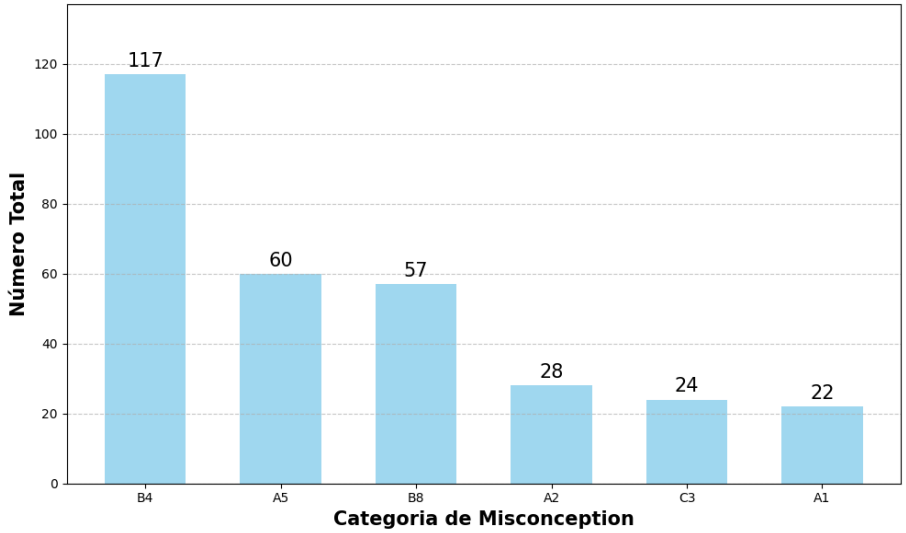

Figura 1. Número de ocorrências dos seis *misconceptions* mais frequentes em avaliações de 83 alunos de programação introdutória

4.2. Relação dos *misconceptions* com os indicadores sociodemográficos

A análise estatística realizada por meio dos testes de Mann-Whitney U e Kruskal-Wallis revelou relações pouco significativas entre a maioria dos indicadores sociodemográficos e a ocorrência dos seis *misconceptions* mais recorrentes. Esses resultados, sintetizados na Tabela 4, permitiram identificar padrões que vinculam características dos estudantes a compreensões imperfeitas de programação.

Tabela 4. Resultados dos testes estatísticos aplicados para avaliar a associação entre categorias de misconceptions e variáveis sociodemográficas

Misconception	Indicador sociodemográfico	Valor p	Conclusão
A5	Experiência prévia de estágio/trabalho	0,0303	Diferença Significativa
B8	Curso era a primeira opção do aluno	0,0319	Diferença Significativa
A5	Possui computador pessoal em casa	0,0726	Inconclusivo (Tendência)
B4	Possui computador pessoal em casa	0,0907	Inconclusivo (Tendência)
A1	Todos os indicadores testados	$>$ 0,05	Sem diferença significativa
A2	Todos os indicadores testados	$>$ 0,05	Sem diferença significativa
C3	Todos os indicadores testados	$>$ 0,05	Sem diferença significativa

A partir dos resultados da Tabela 4, observa-se que apenas duas relações foram verificadas como significativas a 95% (valor $p > 0{,}05$):

- "Experiência prévia de estágio/trabalho" e o *misconception* **A5**
- "Curso era a primeira opção do aluno" e o *misconception* **B8**

Como mostra a Figura 2, estudantes com experiência prévia em estágio ou trabalho apresentaram menos *misconceptions* do tipo A5 (16,6% no grupo "SIM" $\times$ 83,3% no grupo "NÃO"). A diferença acentuada sugere que a vivência profissional pode incentivar práticas de código mais enxutas, evitando importações redundantes. Além disso, alunos que escolheram seus respectivos cursos como primeira opção apresentaram maior incidência de B8 (55,5% no grupo "SIM" $\times$ 44,4% no grupo "NÃO"); embora haja uma

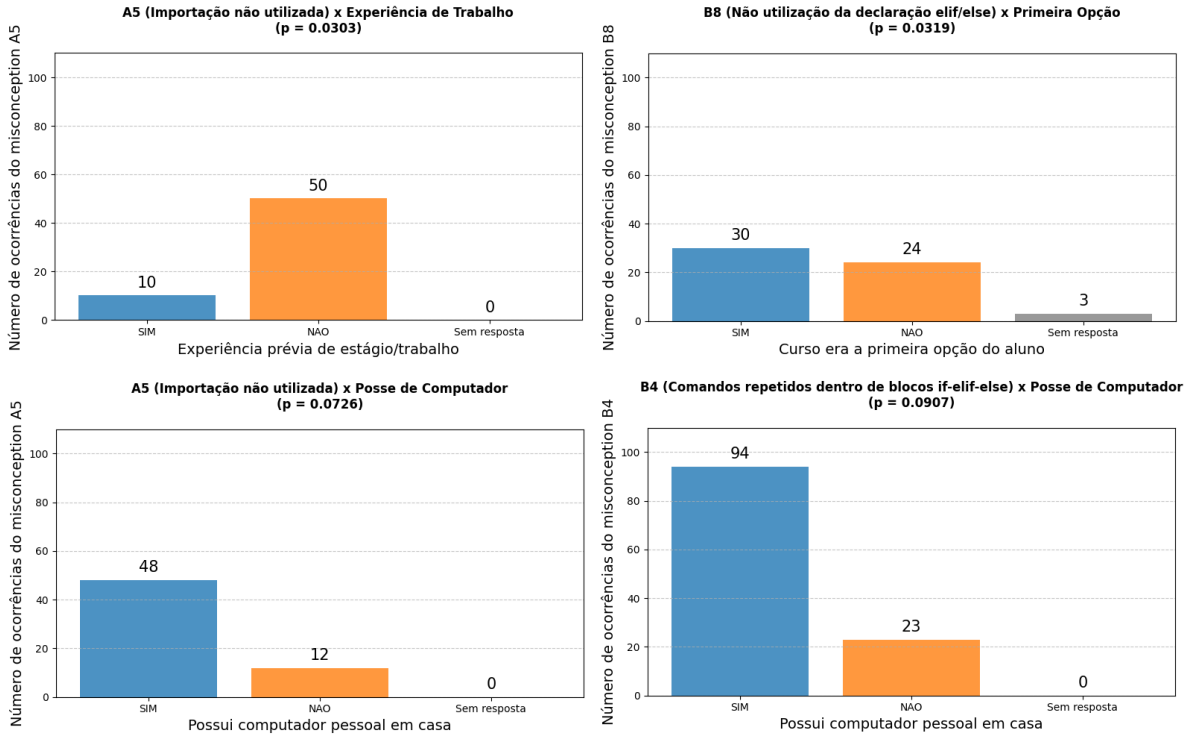

Figura 2. Frequência de misconceptions por perfil sociodemográfico com relações significativas.

associação entre as variáveis, não é possível afirmar que exista uma relação causal entre afinidade com o curso e a adoção de estruturas condicionais menos eficientes.

Adicionalmente, duas relações foram consideradas como inconclusivas, pois os resultados dos testes estatísticos foram próximos do limiar (valor $p \approx 0{,}05$):

- "Possui computador pessoal em casa" e a categoria **A5**
- "Possui computador pessoal em casa" e a categoria **B4**

Como mostra a Figura 2, estudantes com computador pessoal tenderam a apresentar mais vezes o *misconception* A5 (80% no grupo "SIM" $\times$ 20% no grupo "NÃO"). Esses números sugerem que o acesso à tecnologia não necessariamente implica uma melhor compreensão dos conceitos fundamentais de programação. Além disso, estudantes com computador pessoal também apresentaram mais *misconceptions* da categoria B4 (80% no grupo "SIM" $\times$ 20% no grupo "NÃO"), corroborando com a hipótese anterior. Por fim, os *misconceptions* **A1** (variável não utilizada), **A2** (variável atribuída a si mesma) e **C3** (operações redundantes em laços) não mostraram associações significativas com nenhum dos indicadores sociodemográficos observados.

5. Limitações

Durante o estudo, foram identificadas algumas limitações que podem ter impactado os resultados. Uma delas diz respeito às questões de avaliação deixadas em branco ou preenchidas parcialmente, o que dificultou identificar um número maior de *misconceptions*. Essa situação foi mais frequente entre os alunos da Licenciatura em Matemática, grupo

em que houve maior desistência nas avaliações. Como consequência, houve uma subrepresentação nesse curso, criando uma impressão equivocada de que essa turma apresentou menos *misconceptions* em comparação com a de Engenharia Mecânica.

Outra limitação importante foi a baixa taxa de resposta a algumas questões do questionário sociodemográfico, especialmente aquelas relacionadas à modalidade de ingresso na universidade e ao tipo de cota. Essas variáveis, bem como outras com baixa adesão, não puderam ser incluídas nas análises estatísticas (conforme Figura 2), o que restringiu o escopo da investigação. A incorporação desses dados teria potencial para enriquecer o estudo, ampliando a identificação de possíveis relações entre fatores sociodemográficos e a ocorrência de *misconceptions*.

Por fim, a principal limitação foi o tamanho reduzido da amostra: apenas 83 estudantes, o que pode ter comprometido o poder estatístico dos testes aplicados, dificultando a detecção de diferenças significativas entre os grupos analisados. Como solução, futuros estudos devem ampliar a amostra, aplicando identificação automatizada de misconceptions e usando dados de diferentes instituições. Isso não apenas aumentaria a robustez estatística, como também permitiria melhor generalização dos achados.

6. Considerações Finais e Trabalhos Futuros

Este estudo apresentou uma análise preliminar da relação entre *misconceptions* em programação e perfis sociodemográficos. Dos 72 testes estatísticos realizados (6 categorias mais recorrentes de *misconceptions* $\times$ 12 perguntas do questionário), apenas dois apresentaram diferenças significativas entre os grupos analisados: A5 $\times$ experiência prévia de estágio/trabalho e B8 $\times$ escolha do curso como primeira opção. Essa baixa taxa de significância estatística sugere que, com o número atual de participantes, foi difícil identificar associação clara entre os perfis sociodemográficos e as categorias de *misconceptions*. Mesmo sutis, essas relações sinalizam que fatores motivacionais e de vivência são importantes e devem ser considerados durante o processo de aprendizagem, o que sugere a necessidade de estratégias pedagógicas mais personalizadas para com os alunos.

As relações encontradas sugerem caminhos para investigações futuras, como a influência de trabalho prévio ou identificação com a carreira acadêmica, mas precisam ser interpretadas com cautela. Assim, para estudos futuros, recomenda-se uma ampliação do tamanho da amostra, o que aumentaria o poder estatístico das análises e permitiria identificar relações mais consistentes entre *misconceptions* e indicadores sociodemográficos. Para isso, é necessário desenvolver métodos automatizados para a detecção de *misconceptions*, a fim de tornar o processo mais ágil e preciso.

Agradecimentos

Os autores agradecem ao Conselho Nacional de Desenvolvimento Científico e Tecnológico (CNPq) e à Universidade Federal do Amazonas (UFAM) pelo apoio financeiro por meio do Edital 12/2024 – PROPESP/UFAM, vinculado ao Programa Institucional de Bolsas de Iniciação Científica (PIBIC), que viabilizou a participação de dois dos pesquisadores neste estudo.

Referências

Araújo, A., Filho, D. L. Z., Harada, E., Oliveira, T., Carvalho, L. S. G., Pereira, F. D., and Oliveira, D. B. F. (2021). Mapeamento e análise empírica de misconceptions

comuns em avaliações de introdução à programação. In *Anais do Simpósio Brasileiro de Educação em Computação*, pages 123–131, Porto Alegre, RS, Brasil. SBC.

Bosse, Y., Wiese, I. S., Silva, M. A. G., Lago, N., Brandão, L. O., Redmiles, D., Kon, F., and Gerosa, M. A. (2021). Catalogs of C and Python Antipatterns by CS1 Students. Technical report, Department of Computer Science (IME), University of São Paulo (USP).

Caceffo, R., França, B., Gama, G., Benatti, R., Aparecida, T., Caldas, T., and Azevedo, R. (2017). An antipattern documentation about misconceptions related to an introductory programming course in C. In *Technical Report 17-15*, page 42. Institute of Computing, University of Campinas.

Casiraghi, B., Almeida, L. S., Boruchovitch, E., and Aragão, J. C. S. (2020). Rendimento acadêmico no ensino superior: variáveis pessoais e socioculturais do estudante. *Revista Práxis*, 12(24):95–104.

Ettles, A., Luxton-Reilly, A., and Denny, P. (2018). Common logic errors made by novice programmers. In *Proceedings of the 20th Australasian Computing Education Conference*, ACE '18, page 83–89, New York, NY, USA. Association for Computing Machinery.

Gama, G., Caceffo, R., Souza, R., Bennati, R., Aparecida, T., Garcia, I., and Azevedo, R. (2018). An antipattern documentation about misconceptions related to an introductory programming course in Python. *Institute of Computing, University of Campinas, Tech. Rep. IC-18-19*, page 106.

Henley, A., Ball, J., Klein, B., Rutter, A., and Lee, D. (2021). An Inquisitive Code Editor for Addressing Novice Programmers' Misconceptions of Program Behavior. In *International Conference on Software Engineering*, pages 165–170. IEEE Computer Society.

Lewis, C. M., Clancy, M. J., and Vahrenhold, J. (2019). Student knowledge and misconceptions. In Fincher, S. A. and Robins, A. V., editors, *The Cambridge Handbook of Computing Education Research*, The Cambridge Handbook of Computing Education Research, pages 773–800. Cambridge University Press.

Lima, M. A., Carvalho, L. S., Oliveira, E. H., Oliveira, D. B., and Pereira, F. D. (2021). Uso de atributos de código para classificar a dificuldade de questões de programação em juízes online. *Revista Brasileira de Informática na Educação*, 29:1137–1157.

Lopes, A. D. (2017). Affirmative action in Brazil: how students' field of study choice reproduces social inequalities. *Studies in Higher Education*, 42(12):2343–2359.

Qian, Y. and Lehman, J. (2017). Students' misconceptions and other difficulties in introductory programming: A literature review. *ACM Transactions on Computing Education (TOCE)*, 18(1):1–24.

Silva, E. P., Caceffo, R., and Azevedo, R. (2023a). When Test Cases Are Not Enough: Identification, Assessment, and Rationale of Misconceptions in Correct Code (MC³). *Revista Brasileira de Informática na Educação*, 31:1165–1199.

Silva, E. P., Caceffo, R. E., and Azevedo, R. (2023b). Passar nos casos de teste é suficiente? Identificação e análise de problemas de compreensão em códigos corretos.

In *Simpósio Brasileiro de Educação em Computação (EDUCOMP)*, pages 119–129. SBC.

Silveira Filho, A. N., Passos, V. A., Carvalho, L. S. G., Oliveira, E. H. T., Fernandes, D., and Nakamura, F. G. (2025). Estudo sobre o surgimento e evolução de misconceptions em uma disciplina de introdução à programação. In *Anais do V Simpósio Brasileiro de Educação em Computação (EDUCOMP 2025)*, pages 153–164. Sociedade Brasileira de Computação (SBC).

Sorva, J. (2023). Misconceptions and the beginner programmer. In Sentance, S., Barendsen, E., Howard, N. R., and Schulte, C., editors, *Computer science education: Perspectives on teaching and learning in school*, pages 259–273. Bloomsbury, 2 edition.

Souza, D. M., Batista, M. H. S., and Barbosa, E. F. (2016). Problemas e dificuldades no ensino e na aprendizagem de programação: Um mapeamento sistemático. *Revista Brasileira de Informática na Educação*, 24:39–52.

Watson, C. and Li, F. W. (2014). Failure rates in introductory programming revisited. In *Proceedings of the 2014 Conference on Innovation & Technology in Computer Science Education*, ITiCSE '14, pages 39–44, New York, NY, USA. Association for Computing Machinery.