IA GENERATIVA E BARREIRAS PERCEBIDAS À PROTOTIPAÇÃO TECNOLÓGICA: UM LEVANTAMENTO COM ESTUDANTES DE ÁREAS NÃO TECNOLÓGICAS

João Francisco Trencher Martins[1], Rodnil da Silva Moreira Lisboa[2], Lucy Mari Tabuti[3]

[1]Fundação Escola de Comércio Álvares Penteado (FECAP) / joao.martins@fecap.br
[2]Fundação Escola de Comércio Álvares Penteado (FECAP) / rodnil.lisboa@fecap.br
[3]Escola Superior de Propaganda e Marketing (ESPM) / lucy.tabuti@espm.br

Resumo: Este estudo investiga como estudantes de áreas não tecnológicas percebem o papel da Inteligência Artificial Generativa (IAG) diante de atividades de programação, robótica e sistemas embarcados. Realizou-se um levantamento pontual, anônimo e voluntário de opinião com 24 estudantes matriculados na disciplina eletiva Negócios Inovadores Baseados em Tecnologia. O instrumento impresso reuniu questões sobre experiência prévia, barreiras percebidas, disposição para programar e capacidade percebida para executar tarefas tecnológicas em cenários com e sem apoio de IAG. Os resultados mostram baixa experiência prévia: 70,8% nunca haviam programado ou tiveram apenas contato superficial e 87,5% nunca haviam utilizado microcontroladores. Para 66,7%, a necessidade de programar poderia ser motivo para evitar uma eletiva. Com apoio de IAG, 91,7% declararam maior disposição para experimentar tecnologias que ainda não dominam. A parcela que não se percebia capaz de executar nenhuma das oito tarefas propostas caiu de 58,3%, no cenário sem IAG, para 12,5% com IAG. Conclui-se que os participantes atribuem à IAG um papel relevante na redução da barreira percebida de entrada à prototipação, embora permaneçam preocupações quanto à compreensão e validação do código gerado.

Palavras-chave: Inteligência Artificial Generativa. Programação. Prototipação Tecnológica.

Introdução

A programação deixou de constituir uma competência restrita a profissionais de Ciência da Computação. A crescente presença de sistemas digitais em diferentes áreas do conhecimento e do trabalho faz com que estudantes de formações diversas encontrem situações nas quais compreender, adaptar ou produzir código se torna útil para analisar dados, automatizar tarefas e desenvolver protótipos. Ali e Dasgupta (2025) observam que o ensino de programação tem se expandido para cursos não pertencentes à Computação e ressaltam que os objetivos desses estudantes podem diferir significativamente daqueles de quem pretende atuar profissionalmente como desenvolvedor. Nesse contexto, ganha relevância a figura do programador não profissional, que utiliza a programação como instrumento para resolver problemas de seu próprio domínio ou para participar de processos de criação tecnológica.

Essa expansão, entretanto, não elimina as dificuldades associadas ao primeiro contato com a programação. Em estudo com 241 estudantes de Engenharia sem formação em Ciência da Computação, Kadar et al. (2022) identificaram dificuldades relacionadas à natureza da

linguagem de programação, às capacidades dos estudantes e às estratégias pedagógicas utilizadas. Entre os obstáculos recorrentes estão a necessidade de elaborar algoritmos, traduzir soluções para uma sintaxe formal, testar programas e localizar erros. Tais desafios podem ser acompanhados por percepções negativas acerca da própria capacidade de programar, o que transforma a programação não apenas em uma dificuldade técnica, mas também em uma barreira de entrada para experiências educacionais que envolvem tecnologia.

Ali e Dasgupta (2025), ao investigarem estudantes não pertencentes à Ciência da Computação, encontraram interesse e curiosidade pela programação coexistindo com frustrações provocadas pelo desenho de cursos e pela ausência de recursos adequados às necessidades de não especialistas. Os autores retomam, ainda, a noção de *conversational programmer*, isto é, alguém que não necessariamente deseja tornar-se programador profissional, mas busca domínio suficiente da linguagem e dos conceitos computacionais para dialogar, trabalhar e criar em contextos nos quais a tecnologia está presente. Essa perspectiva é particularmente pertinente em disciplinas voltadas à inovação e aos negócios, nas quais a programação pode atuar como meio para materializar uma ideia, e não como finalidade formativa isolada.

O surgimento de ferramentas de Inteligência Artificial Generativa baseadas em grandes modelos de linguagem acrescentou um novo elemento a esse cenário. Sistemas como ChatGPT e outros assistentes de programação podem gerar trechos de código, explicar estruturas, sugerir correções, auxiliar na depuração e responder a dúvidas em linguagem natural. Phung et al. (2023) avaliaram modelos generativos em diferentes situações do ensino introdutório de programação — incluindo reparo de programas, geração de dicas, explicações contextualizadas e programação em pares — e observaram desempenho competitivo em diversas tarefas, embora persistam limitações em cenários mais complexos. A possibilidade de solicitar auxílio em linguagem cotidiana cria, portanto, uma interface entre a intenção do estudante e conhecimentos técnicos que antes exigiriam maior domínio prévio.

Além da dimensão operacional, a IAG pode produzir efeitos sobre a maneira como o estudante percebe a própria capacidade de enfrentar uma tarefa. Boguslawski, Deer e Dawson (2025), analisando a motivação no ensino de programação, identificaram que o uso de grandes modelos de linguagem pode favorecer autonomia e percepção de competência, oferecer apoio personalizado e reduzir estados afetivos negativos, como frustração e sensação de estar bloqueado diante de um problema. Os autores relacionam esses efeitos à teoria da autodeterminação, segundo a qual autonomia e competência constituem dimensões relevantes

da motivação. Isso não implica afirmar que a presença da IAG produz automaticamente aprendizagem ou domínio de programação, mas sugere que ela pode alterar o limiar a partir do qual um estudante se dispõe a tentar.

Esse potencial de redução de barreiras precisa ser analisado juntamente com seus limites. Kohen-Vacs, Usher e Jansen (2025) verificaram percepções favoráveis ao uso de IAG em cursos de programação, especialmente quanto à eficiência e ao apoio à resolução de problemas, mas também destacaram o risco de dependência excessiva e a dificuldade dos estudantes em corrigir código produzido por modelos generativos. Os autores defendem que a integração educacional dessas ferramentas deve preservar competências de análise, depuração e avaliação crítica. De modo complementar, Jin et al. (2025) mostram que os benefícios obtidos em interações com sistemas generativos podem depender do nível de letramento em IAG do estudante; estratégias de scaffolding, nas quais o sistema fornece orientação estruturada, podem reduzir parte dessa dependência e ampliar o acesso ao suporte.

Essas tensões são especialmente relevantes em atividades de prototipação tecnológica com estudantes de áreas não tecnológicas. Em uma disciplina que utiliza microcontroladores, sensores, atuadores e interfaces web, o estudante pode precisar escrever ou adaptar códigos em C/C++ para Arduino e, eventualmente, utilizar HTML e JavaScript para apresentar dados coletados. Para esse público, uma questão central não é apenas se a IAG permite produzir código mais rapidamente, mas se sua disponibilidade modifica a disposição para entrar em um território percebido como técnico e, portanto, potencialmente inacessível.

Diante desse contexto, este trabalho tem como objetivo investigar como estudantes de áreas não tecnológicas percebem o papel da Inteligência Artificial Generativa na redução de barreiras associadas à programação e à prototipação tecnológica. Busca-se, especificamente, caracterizar a experiência tecnológica prévia dos participantes; identificar em que medida a programação é percebida como obstáculo à escolha e realização de atividades tecnológicas; comparar a capacidade que os estudantes atribuem a si mesmos em tarefas realizadas com e sem apoio de IAG; e discutir a tensão entre facilitação, autonomia e necessidade de compreensão crítica do código produzido por sistemas generativos.

Metodologia

O estudo possui caráter descritivo e abordagem quantitativa, sendo estruturado como levantamento pontual de opinião. A coleta foi realizada em 26 de agosto de 2026 com

estudantes matriculados na disciplina eletiva "Negócios Inovadores Baseados em Tecnologia", ofertada em uma instituição de ensino superior privada na cidade de São Paulo. A disciplina reúne estudantes de cursos não pertencentes à área de tecnologia e utiliza, como eixo prático, atividades de robótica, sistemas embarcados e prototipação, incluindo programação de microcontroladores em C/C++ e, conforme as necessidades dos projetos, construção de interfaces simples em HTML e JavaScript.

O instrumento foi aplicado em papel durante o horário de aula. A participação foi voluntária e não esteve vinculada a nota, frequência ou qualquer forma de avaliação acadêmica. Não foram solicitados nome, matrícula, endereço eletrônico, assinatura, curso individual ou outros identificadores. Os questionários preenchidos foram recolhidos sem associação a listas de presença ou registros institucionais e analisados apenas de forma agregada. O desenho foi deliberadamente restrito à expressão de percepções, preferências e avaliações dos participantes, sem mensuração de desempenho real ou intervenção experimental.

Esse delineamento foi adotado em consonância com a Resolução CNS nº 510/2016, que define pesquisa de opinião pública como consulta pontual na qual o participante expressa preferência, avaliação ou sentido atribuído a determinado tema sem possibilidade de identificação. A mesma resolução estabelece que pesquisas de opinião pública com participantes não identificados não são registradas nem avaliadas pelo Sistema CEP/CONEP (BRASIL, 2016). O Ofício Circular nº 17/2022 da CONEP reforça que esse enquadramento se aplica a levantamentos cujo propósito é descrever a valoração atribuída pelos participantes ao objeto consultado (BRASIL, 2022).

O questionário foi organizado em cinco grupos de itens. O primeiro caracterizou a experiência prévia com programação, microcontroladores e uso de IAG para criação, compreensão ou correção de códigos. O segundo reuniu oito afirmações avaliadas em escala Likert de cinco pontos, de "discordo totalmente" a "concordo totalmente", abordando barreira percebida, capacidade percebida, disposição para experimentar tecnologias, importância da compreensão do código e receio de utilizar código não validado. O terceiro apresentou um cenário contrafactual no qual a disciplina exigiria programação, mas não permitiria IAG para criar, compreender ou corrigir códigos. O quarto apresentou oito tarefas tecnológicas e solicitou que os estudantes indicassem quais acreditavam conseguir realizar sem IAG e, em seguida, com IAG. O quinto grupo investigou usos pretendidos da ferramenta e a percepção global de seu papel na programação.

Foram obtidos 24 questionários válidos (N=24). Os dados foram digitados em planilha

e submetidos a análise descritiva, utilizando frequências absolutas e relativas. Nas questões de múltipla escolha, cada alternativa foi tratada como variável dicotômica. A comparação entre os cenários "sem IAG" e "com IAG" foi realizada no nível agregado, preservando o caráter descritivo do levantamento. Optou-se por não utilizar os resultados como evidência de aprendizagem, desempenho ou causalidade, pois o instrumento mede exclusivamente capacidade e disposição percebidas pelos respondentes.

Resultados e discussões

Experiência prévia e programação como barreira

Os dados iniciais confirmam que a maior parte dos participantes ingressou na disciplina com pouca familiaridade com programação. Nove estudantes (37,5%) afirmaram nunca ter programado e oito (33,3%) relataram apenas contato superficial. Os sete restantes (29,2%) haviam realizado algumas atividades ou pequenos códigos; nenhum participante se classificou como tendo experiência intermediária ou avançada. Em relação a Arduino, ESP32 ou outros microcontroladores, 21 estudantes (87,5%) nunca haviam utilizado tais dispositivos, um (4,2%) apenas havia ouvido falar e dois (8,3%) declararam alguma experiência prática.

A experiência anterior com IAG aplicada à programação também era limitada. Dez respondentes (41,7%) nunca haviam utilizado uma ferramenta generativa para criar, compreender ou corrigir códigos, dois (8,3%) a haviam utilizado uma ou duas vezes e 12 (50,0%) declararam algumas utilizações; nenhum indicou uso frequente ou regular. Assim, embora a IAG não fosse completamente desconhecida, o grupo não pode ser caracterizado como composto por usuários experientes de assistentes de programação.

Os itens de percepção reforçam a existência de uma barreira inicial. Dezesseis participantes (66,7%) concordaram ou concordaram totalmente que, antes da disciplina, a necessidade de programar poderia fazê-los evitar uma eletiva. Apenas dois (8,3%) concordaram que se consideravam capazes de desenvolver um projeto envolvendo programação, enquanto 20 (83,3%) discordaram ou discordaram totalmente dessa afirmação. A mesma proporção de discordância (83,3%) apareceu quando a afirmação tratou de projetos envolvendo sensores e microcontroladores. Esses resultados aproximam-se da literatura que descreve programação como atividade capaz de produzir frustração, insegurança e sensação de sobrecarga entre estudantes não pertencentes à Computação (KADAR et al., 2022; ALI; DASGUPTA, 2025).

A IAG e a disposição para enfrentar tarefas tecnológicas

Quando as afirmações passaram a incorporar explicitamente a possibilidade de apoio por IAG, observou-se mudança importante nas avaliações. Treze participantes (54,2%) concordaram que saber que poderiam recorrer a uma IAG aumentava sua disposição para realizar atividades envolvendo programação. Esse percentual subiu para 75,0% na afirmação segundo a qual a IAG reduz a sensação de que programação é uma barreira para desenvolver um projeto tecnológico. O resultado mais expressivo ocorreu na disposição para experimentar tecnologias ainda não dominadas: 22 dos 24 respondentes (91,7%) concordaram ou concordaram totalmente com essa afirmação, e nenhum discordou.

Esse padrão é coerente com Boguslawski, Deer e Dawson (2025), que associam o uso de LLMs em programação a maior percepção de autonomia e competência e à redução de estados afetivos negativos que podem interromper ou desencorajar a aprendizagem. No presente estudo não se mediu autoeficácia por escala psicométrica validada, nem se avaliou aprendizagem efetiva; portanto, os resultados devem ser interpretados como percepção declarada de maior disposição e capacidade. Ainda assim, a magnitude das respostas sugere que, para esse grupo, a IAG não é percebida apenas como ferramenta de produtividade, mas como suporte que altera a disposição para tentar.

Essa interpretação é reforçada pela questão que apresentou um cenário no qual a disciplina exigiria programação, mas não permitiria o uso de IAG para criar, compreender ou corrigir códigos. Oito estudantes (33,3%) afirmaram que provavelmente ainda escolheriam a disciplina, quatro (16,7%) responderam “não sei”, sete (29,2%) disseram que provavelmente não a escolheriam e cinco (20,8%) afirmaram que certamente não a escolheriam. Nenhum participante selecionou a opção “certamente sim”. Assim, metade da amostra declarou que provavelmente ou certamente não teria escolhido a disciplina nesse cenário.

Esse resultado, contudo, não pode ser interpretado como evidência de que a disponibilidade de IAG tenha causado a matrícula. Apenas dez participantes (41,7%) disseram que, no momento da escolha da disciplina, já imaginavam que poderiam utilizar ferramentas generativas como apoio à programação; cinco (20,8%) não imaginavam essa possibilidade, cinco (20,8%) não haviam pensado sobre o tema e quatro (16,7%) sequer sabiam que haveria atividades de programação. O dado contrafactual expressa, portanto, a avaliação retrospectiva dos estudantes sobre sua disposição atual, e não um efeito causal comprovado sobre a decisão original de matrícula.

Capacidade percebida em cenários com e sem IAG

A comparação direta entre as listas de tarefas “sem IAG” e “com IAG” evidencia de forma ainda mais clara a função de suporte atribuída às ferramentas generativas. Sem auxílio de IAG, cinco estudantes (20,8%) acreditavam conseguir escrever um código simples para Arduino ou outro microcontrolador; com IAG, esse número passou a 14 (58,3%). A leitura de dados de sensores passou de 8,3% para 50,0%, a correção de erros em código de 16,7% para 75,0% e a adaptação de código existente de 12,5% para 66,7%.

O contraste mais sintético aparece na alternativa “nenhuma das anteriores”. No cenário sem IAG, 14 participantes (58,3%) afirmaram não se perceber capazes de realizar nenhuma das oito tarefas apresentadas. Com IAG, essa alternativa foi assinalada por apenas três estudantes (12,5%). Em outras palavras, a presença hipotética da IAG desloca a percepção do grupo de uma condição em que a maioria não se considera capaz de realizar qualquer tarefa para outra em que 87,5% reconhecem ao menos uma possibilidade de ação tecnológica.

As diferenças são particularmente intensas em tarefas diretamente compatíveis com as capacidades atuais de assistentes generativos: gerar um código inicial, interpretar uma leitura de sensor, localizar erros e adaptar código existente. Phung et al. (2023) demonstram que grandes modelos de linguagem podem desempenhar papéis de tutor, assistente e parceiro em atividades como reparo de programas, geração de dicas e explicações contextualizadas. Assim, a percepção dos estudantes de que a IAG amplia aquilo que conseguem tentar encontra correspondência nas funcionalidades efetivamente oferecidas por esses sistemas.

A ampliação da capacidade percebida, porém, não deve ser confundida com aquisição automática de competência. Desenvolver um protótipo funcional a partir de uma ideia foi assinalado por apenas 4,2% dos participantes no cenário sem IAG e por 29,2% no cenário com IAG. Mesmo com a ferramenta, portanto, a maioria não se considera pronta para executar integralmente a cadeia de projeto. Isso sugere que a IAG pode reduzir a barreira inicial e apoiar etapas específicas, mas não elimina a necessidade de estrutura pedagógica, experimentação prática, mediação docente e integração entre conhecimentos de hardware, software e projeto.

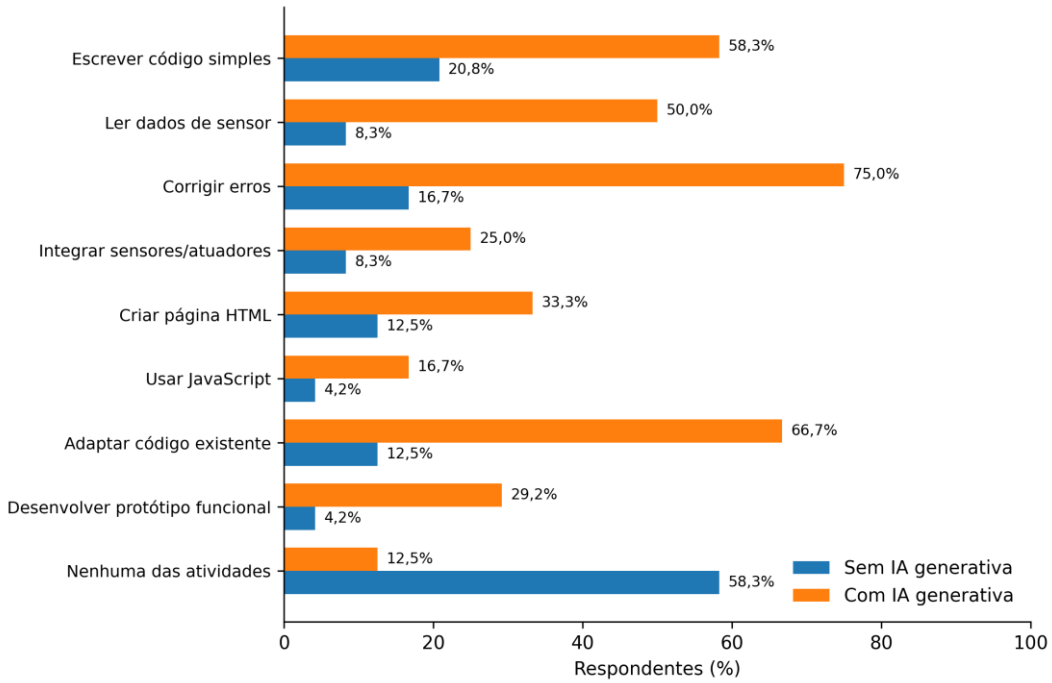


Figura 1: Capacidade percebida para realizar tarefas tecnológicas sem e com apoio de IAG

Usos esperados, compreensão e risco de dependência

Os participantes também indicaram como imaginavam utilizar IAG ao longo da disciplina. Os usos mais frequentes foram gerar códigos (75,0%), encontrar e corrigir erros (66,7%), explicar códigos não compreendidos (62,5%), modificar ou adaptar códigos existentes (62,5%), entender sensores, componentes ou microcontroladores (58,3%) e aprender conceitos de programação (58,3%). Nenhum participante marcou a opção “não pretendo utilizar IA Generativa”, embora cinco (20,8%) tenham indicado que ainda não sabiam como pretendiam utilizá-la. O padrão reforça a ideia de que os estudantes esperam utilizar a tecnologia tanto para produção quanto para compreensão e aprendizagem.

Na questão de síntese, 12 participantes (50,0%) escolheram a alternativa “sem IA, eu provavelmente evitaria atividades que exigissem programação” e nove (37,5%) afirmaram que a IAG torna possível realizar atividades de programação que consideram difíceis demais para fazer sozinhos. Apenas um estudante (4,2%) avaliou que conseguiria realizar essas atividades também sem a ferramenta, um (4,2%) considerou que a IAG apenas facilita e acelera o trabalho sem alterar sua disposição e um (4,2%) ainda não tinha opinião formada. Nenhum declarou preferir programar sem depender de IAG. Assim, 87,5% escolheram justamente as duas alternativas nas quais a presença da tecnologia modifica substancialmente sua relação percebida com a programação.

Os mesmos estudantes, entretanto, demonstraram consciência de que obtenção de um código funcional não encerra o problema educacional. Vinte respondentes (83,3%) concordaram que é importante compreender como o código funciona mesmo quando uma IAG o gera, e 16 (66,7%) relataram receio de utilizar código gerado sem conseguir avaliar se está correto. Essa tensão é compatível com os achados de Kohen-Vacs, Usher e Jansen (2025): embora estudantes reconheçam utilidade e potencial criativo em ferramentas generativas, a correção e a avaliação crítica de códigos produzidos por LLMs permanecem desafios, e a dependência excessiva pode comprometer oportunidades de desenvolver raciocínio algorítmico e habilidades de depuração.

Consequentemente, interpretar a IAG como redutora de barreiras não implica defendê-la como substituta do conhecimento técnico. Ao contrário, os dados sugerem um deslocamento do papel pedagógico da programação para estudantes de áreas não tecnológicas. Em vez de exigir que o estudante domine previamente toda a sintaxe necessária antes de construir, pode-se permitir que ele inicie a prototipação com apoio generativo, ao mesmo tempo em que se estabelecem atividades que exijam leitura, explicação, teste, comparação e correção do código produzido. Nesse modelo, a capacidade de dialogar com a IAG, avaliar suas respostas e relacioná-las ao comportamento físico do protótipo passa a integrar a própria alfabetização tecnológica.

Jin et al. (2025) ajudam a qualificar esse argumento ao mostrar que o benefício de sistemas generativos depende também do letramento em IAG do usuário. Estudantes com maior capacidade de formular solicitações e interpretar respostas podem extrair mais valor de chatbots convencionais, enquanto mecanismos de scaffolding podem oferecer orientação adicional a usuários menos experientes. Para disciplinas de prototipação, isso aponta para a necessidade de ensinar explicitamente como formular problemas, fornecer contexto do circuito e do código, solicitar explicações, testar hipóteses e verificar respostas. A disponibilidade da ferramenta, isoladamente, não garante uma mediação pedagogicamente produtiva.

Os resultados deste estudo permitem, portanto, propor uma interpretação específica para o contexto investigado: a IAG funciona, na percepção dos participantes, como uma espécie de ponte entre uma intenção de projeto e uma camada técnica inicialmente percebida como inacessível. Essa ponte parece especialmente relevante quando o objetivo do estudante não é tornar-se programador profissional, mas construir um artefato, validar uma ideia ou compreender suficientemente o sistema para tomar decisões de projeto. Essa leitura aproxima-se da discussão de Ali e Dasgupta (2025) sobre diferentes objetivos de aprendizagem entre

estudantes não pertencentes à Computação e ajuda a deslocar a pergunta educacional de "o estudante consegue escrever todo o código sozinho?" para "o estudante consegue compreender, testar, modificar e assumir responsabilidade pelo sistema que está construindo?".

Por fim, é necessário reconhecer as limitações do levantamento. A amostra é pequena, composta por 24 respondentes de uma única disciplina e obtida por conveniência, o que impede generalizações para outros cursos ou instituições. As respostas representam percepções no início do processo formativo e não demonstram aprendizagem, ganho de desempenho ou autonomia efetiva. O cenário "com IAG" é hipotético em parte das questões, e os participantes apresentavam diferentes níveis de familiaridade com ferramentas generativas. Estudos futuros podem acompanhar a turma longitudinalmente, comparar percepções antes e depois dos projetos, analisar os tipos de prompts e interações utilizados e avaliar em que medida a redução da barreira percebida se converte em capacidade de compreender, depurar e modificar sistemas reais.

Considerações finais

O levantamento mostrou que a programação constitui uma barreira relevante para os estudantes de áreas não tecnológicas participantes deste estudo. A maioria ingressou na disciplina com pouca ou nenhuma experiência em programação e praticamente sem contato prévio com microcontroladores. Dois terços declararam que a necessidade de programar poderia levá-los a evitar uma disciplina eletiva, e mais de 80% não se consideravam capazes de desenvolver projetos envolvendo programação ou sistemas embarcados.

A presença da Inteligência Artificial Generativa aparece associada, nas respostas, a uma mudança expressiva na maneira como esses estudantes descrevem sua disposição para tentar. A maioria afirmou que a IAG reduz a sensação de que programação é uma barreira e 91,7% declararam maior disposição para experimentar tecnologias ainda não dominadas. A comparação entre tarefas com e sem apoio generativo sintetiza esse fenômeno: a proporção de estudantes que não se percebia capaz de realizar nenhuma das oito tarefas propostas caiu de 58,3% para 12,5%. Esses dados não demonstram aumento de competência real, mas evidenciam que os participantes atribuem à IAG um importante papel de redução da barreira percebida de entrada.

Ao mesmo tempo, o grupo não apresentou uma visão acrítica da tecnologia. A grande maioria considera necessário compreender o código mesmo quando ele é gerado por IAG, e

dois terços manifestaram receio de utilizar respostas sem conseguir verificar sua correção. Assim, a contribuição pedagógica mais promissora parece residir não na substituição da programação pela IAG, mas em sua utilização como suporte para que estudantes não especialistas possam iniciar processos de criação tecnológica antes que dominem completamente todas as camadas técnicas envolvidas.

Para disciplinas de inovação, maker e prototipação, isso sugere um modelo no qual a IAG atua como scaffolding inicial, enquanto o professor desloca parte de sua atenção para competências de formulação do problema, leitura de código, validação, depuração e compreensão do comportamento do sistema. Estudos longitudinais serão necessários para verificar se essa disposição inicial se converte em autonomia tecnológica ao término da disciplina e para identificar estratégias capazes de reduzir dependência sem restabelecer a barreira de entrada que a ferramenta parece ajudar a diminuir.

Termos de uso de Inteligência Artificial Generativa

Neste trabalho foi utilizada a ferramenta de Inteligência Artificial Generativa ChatGPT (OpenAI) como apoio, especialmente para fins de organização de ideias, apoio à redação, revisão textual e sistematização preliminar de informações e dados. A definição do problema, a coleta dos dados, a seleção das referências, a interpretação dos resultados e a revisão da versão final permaneceram sob responsabilidade dos autores.

Referências

ALI, Murtaza; DASGUPTA, Sayamindu. **“Even Though I Went Through Everything, I Didn’t Feel Like I Learned a Lot”: Insights From Experiences of Non-Computer Science Students Learning to Code**. In: CHI CONFERENCE ON HUMAN FACTORS IN COMPUTING SYSTEMS (CHI ’25), 2025, Yokohama. Proceedings [...]. New York: ACM, 2025. 18 p. DOI: 10.1145/3706598.3713624.

BOGUSLAWSKI, Samuel; DEER, Rowan; DAWSON, Mark G. **Programming education and learner motivation in the age of generative AI: student and educator perspectives.** Information and Learning Sciences, v. 126, n. 1/2, p. 91-109, 2025. DOI: 10.1108/ILS-10-2023-0163.

BRASIL. **Conselho Nacional de Saúde**. Resolução nº 510, de 7 de abril de 2016. Dispõe sobre as normas aplicáveis a pesquisas em Ciências Humanas e Sociais. Brasília, DF: Conselho Nacional de Saúde, 2016.

BRASIL. **Comissão Nacional de Ética em Pesquisa.** Ofício Circular nº 17, de 5 de julho de 2022. Orientações acerca do artigo 1º da Resolução CNS nº 510, de 7 de abril de 2016.

Brasília, DF: CONEP, 2022.

JIN, Yueqiao et al. **Chatting with a Learning Analytics Dashboard: The Role of Generative AI Literacy on Learner Interaction with Conventional and Scaffolding Chatbots.** In: INTERNATIONAL LEARNING ANALYTICS AND KNOWLEDGE CONFERENCE (LAK 2025), 15., 2025, Dublin. Proceedings [...]. New York: ACM, 2025. 12 p. DOI: 10.1145/3706468.3706545.

KADAR, Rozita; MAHLAN, Siti Balqis; SHAMSUDDIN, Maisurah; ABDUL WAHAB, Naemah; OTHMAN, Jamal. **Analysis of Factors Contributing to the Difficulties in Learning Computer Programming among Non-Computer Science Students**. In: IEEE SYMPOSIUM ON COMPUTER APPLICATIONS & INDUSTRIAL ELECTRONICS (ISCAIE), 12., 2022. Proceedings [...]. IEEE, 2022. DOI: 10.1109/ISCAIE54458.2022.9794546.

KOHEN-VACS, Dan; USHER, Maya; JANSEN, Marc. **Integrating Generative AI into Programming Education: Student Perceptions and the Challenge of Correcting AI Errors**. International Journal of Artificial Intelligence in Education, v. 35, p. 3166-3184, 2025. DOI: 10.1007/s40593-025-00496-4.

PHUNG, Tung et al. **Generative AI for Programming Education: Benchmarking ChatGPT, GPT-4, and Human Tutors**. In: INTERNATIONAL COMPUTING EDUCATION RESEARCH CONFERENCE (ICER), v. 2, 2023. arXiv:2306.17156v3, 2023.