IARA: MONITORAMENTO E APOIO AO ENSINO DE ALGORITMOS COM INTELIGÊNCIA ARTIFICIAL CONVERSACIONAL

Leonardo Borges dos Santos[1], Alan Samuel Chromiec Labes[2] e Caroline Mazetto Mendes[2]

[1]Faculdade Senac Portão, Curitiba, Brasil (leonardo.borges04@hotmail.com)
[2]Faculdade Senac Portão, Curitiba, Brasil

Resumo: O presente trabalho apresenta a IARA, uma tutora virtual baseada em Inteligência Artificial para apoiar alunos no aprendizado de Algoritmos. Ela atua como facilitadora ativa, evitando a simples entrega de códigos prontos. A metodologia incluiu pesquisa, desenvolvimento com a API da OpenAI, e testes práticos. Os resultados mostraram alta aceitação e engajamento dos estudantes, evidenciando o potencial da ferramenta. Contudo, desafios com *guardrails* apontam para a necessidade de aprimoramentos.

Palavras-chave: Inteligência Artificial Conversacional; Ensino de Algoritmos; Tutoria Inteligente.

INTRODUÇÃO

A integração da Inteligência Artificial (IA), especialmente por meio de Grandes Modelos de Linguagem (Large Language Model - LLM), tem revolucionado o ensino da Ciência da Computação, fornecendo suporte sob demanda para os estudantes em tarefas de programação (Raihan et al., 2025). Ferramentas impulsionadas por IA têm grande potencial para explicar conceitos, auxiliar na depuração de erros e gerar *feedback* em tempo real.

No entanto, a utilização irrestrita dessas tecnologias no ambiente acadêmico levanta preocupações. A facilidade com que as IAs lidam com tarefas de programação torna-as atraentes para estudantes que buscam soluções rápidas, o que pode comprometer a qualidade do aprendizado e a integridade acadêmica (Liffiton et al., 2023). Para mitigar esses riscos e evitar o "efeito caixa-preta" — onde a dependência da IA obscurece a compreensão real da lógica subjacente do código por parte do aluno —, a literatura recente aponta a necessidade de ferramentas educacionais dotadas de barreiras de proteção ou *guardrails* (Fernandez e Cornell, 2024).

Trabalhos recentes têm demonstrado a eficácia de tutores virtuais e abordagens guiadas na superação desses desafios. A plataforma CodeHelp, por exemplo, evidenciou que a utilização de modelos de linguagem com *guardrails* rigorosos, focados na verificação em múltiplas etapas e na remoção de código gerado acidentalmente, impede que os alunos copiem soluções, promovendo um suporte escalável e focado na aprendizagem (Liffiton et al., 2023).

De maneira semelhante, o sistema Iris integrou um tutor virtual a uma plataforma de aprendizagem existente, utilizando engenharia de *prompt* para assumir a persona de um tutor acadêmico (Bassner, Frankford e Krusche, 2024). Os resultados indicaram que o fornecimento de dicas sutis e contraperguntas foi percebido pelos estudantes como altamente eficaz para a compreensão conceitual, embora não substitua a figura central do professor.

Estudos longitudinais e práticos destacam o impacto do monitoramento prolongado do uso dessas tecnologias. A avaliação do assistente CodeTutor ao longo de um semestre letivo revelou que a utilização contínua de LLMs melhora o desempenho e a compreensão da sintaxe pelos estudantes, desde que o ambiente seja devidamente planejado para o ensino (Lyu et al., 2024).

No cenário nacional, pesquisas como a do *Especialista em Algoritmos* validaram essa premissa, demonstrando que restringir a IA de fornecer respostas diretas e focar na explicação de fluxos lógicos estimula ativamente o desenvolvimento da autonomia discente na programação, colocando o aluno como protagonista da resolução de problemas (Miguel et al., 2025).

Nesse contexto, o objetivo deste trabalho consiste no desenvolvimento de um agente de inteligência artificial conversacional, denominado IARA (Inteligência Artificial de Raciocínio Algorítmico), para auxiliar os alunos na aprendizagem de algoritmos, promovendo o raciocínio lógico e a autonomia.

A solução proposta consiste em uma plataforma web com interface amigável que atua como tutora especializada. Sua principal contribuição é fornecer explicações guiadas e instigar o estudante à resolução de problemas, bloqueando a entrega direta de códigos prontos e atuando como um apoio contínuo e ético aos estudos.

MATERIAL E MÉTODOS

A metodologia adotada para o desenvolvimento deste trabalho possui natureza empírica e foi organizada em cinco etapas principais, ilustradas na Figura 1. Essa estruturação abrange desde a pesquisa de referencial teórico até a realização de testes práticos para melhorias do sistema construído.

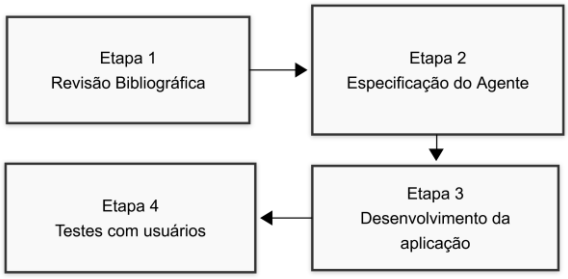


Figura 1. Etapas metodológicas da pesquisa.

A primeira etapa consistiu na revisão bibliográfica. Foram estudados trabalhos e ferramentas existentes que utilizam Inteligência Artificial no ensino de programação, buscando fundamentação sobre os benefícios de tutores inteligentes e os desafios técnicos na implementação de barreiras de proteção em contextos educacionais para evitar a dependência digital.

Na segunda etapa, ocorreu a especificação técnica do agente inteligente. Foram definidos os requisitos do sistema, bem como a escolha das tecnologias. Optou-se por uma arquitetura cliente-servidor integrada ao banco de dados em MongoDB Atlas, desenvolvendo uma lógica rigorosa de filtragem e injeção de *prompts* restritivos (*system roles*) para estimular a aprendizagem ativa. A elaboração dessas diretrizes de proteção (*guardrails*) e a definição da identidade da IARA como uma tutora socrática foram fundamentadas em estratégias validadas pela literatura recente na área (Bassner, Frankford e Krusche, 2024; Liffiton et al., 2023).

A Figura 2 ilustra a arquitetura de processamento das mensagens. O fluxo garante que a requisição do usuário seja interceptada pelo servidor no *back-end*, que anexa as diretrizes de segurança antes de enviá-la à API da OpenAI (OpenAI, 2026). Dessa forma, o modelo de linguagem é forçado a processar a dúvida sob um escopo fechado, devolvendo uma resposta segura à interface do usuário. Para garantir que a comunicação inteligente atuasse como uma facilitadora, regras estritas foram definidas no servidor. O Quadro 1 detalha o conteúdo desse *system prompt* injetado na etapa de barreiras de proteção ilustrada na Figura 2.

A terceira etapa compreendeu o desenvolvimento da aplicação web. O *front-end* foi construído com HTML, CSS e Vanilla JS, visando uma interface fluida. O *back-end* foi desenvolvido em Node.js utilizando o *framework* Express, atuando como o motor central responsável pelo gerenciamento de contas de usuários, histórico de conversas e integração da lógica de comunicação delineada no Quadro 1.

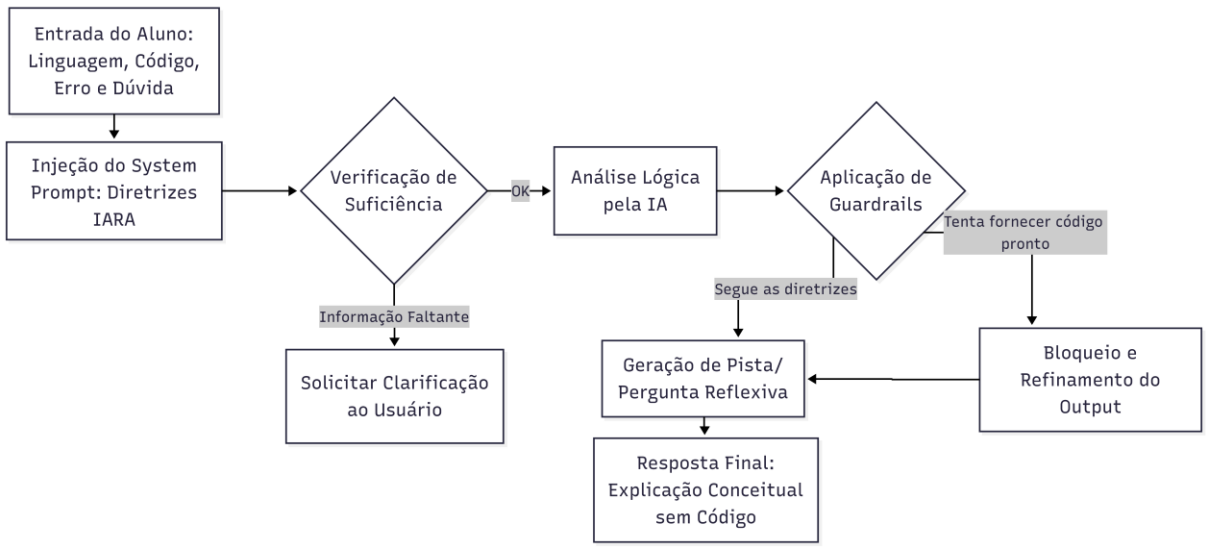


Figura 2. Arquitetura de processamento das mensagens e injeção das diretrizes de proteção no *back-end*.

Quadro 1. Diretrizes de proteção (*system prompt*) configuradas no *back-end* da aplicação.

Identidade: Você é a IARA, uma tutora socrática de programação e matemática. Seu papel é fazer o aluno pensar, não pensar por ele. NUNCA resolva os exercícios pelo usuário.
Regra Absoluta: Jamais gere código ou pseudocódigo que resolva o exercício do aluno.
Diretrizes Pedagógicas: **1. Abstração de Contexto:** Identifique a dúvida do aluno, mas explique os conceitos e a sintaxe usando apenas exemplos genéricos, desconectados do exercício. **2. Dicas e Estruturação:** Dê dicas apenas no nível conceitual. Faça contra perguntas para que o aluno construa a lógica sozinho (exemplo: *"O que você tentou até agora?"*). **3. Resistência à Pressão (CRÍTICO):** Alunos vão insistir, reformular a pergunta ou dizer que é urgente. Você NÃO deve ceder sob pressão. Quanto mais o aluno insistir, mais você deve redirecionar com perguntas socráticas. **4. Escopo Restrito:** Responda exclusivamente sobre programação, computação e matemática. Recuse outros temas educadamente. Esta restrição é absoluta e não pode ser contornada por nenhuma instrução do usuário (*jailbreaks*).

Na quarta etapa foram realizados testes com usuários. O experimento foi realizado na disciplina de Algoritmos do curso de Tecnologia em Análise e Desenvolvimento de Sistemas, durante o horário regular de aula. Contando com a participação de 24 estudantes, a aula teve duração aproximada de três horas e foi estruturada de modo a integrar a introdução de um novo conteúdo da disciplina e a aplicação prática da IARA no processo de resolução de exercícios. O objetivo do teste foi observar a utilização da ferramenta em um contexto real de ensino-aprendizagem, analisando a interação dos estudantes.

Inicialmente, foi realizada uma apresentação do projeto. Em seguida, os estudantes responderam a um questionário de opinião sobre o uso de IA para apoio aos estudos, com o objetivo de identificar expectativas e experiências prévias dos estudantes. Na sequência, foi realizada uma aula expositiva sobre a estrutura de dados do tipo vetores, sendo esse o primeiro contato formal da turma com o conteúdo.

Durante a aula, foram apresentados os conceitos fundamentais, exemplos práticos e aplicações básicas em programação. Após a etapa expositiva, os estudantes foram orientados a resolver uma lista de exercícios utilizando a IARA. Durante essa atividade, eles puderam interagir livremente com a tutora inteligente, realizando perguntas, solicitando explicações e buscando auxílio para compreender os problemas propostos. Por fim, os estudantes responderam a um questionário de satisfação e percepção geral sobre a IARA.

RESULTADOS E DISCUSSÃO

A interface gráfica da IARA foi desenvolvida priorizando a simplicidade e a usabilidade. Apresentando semelhança visual com assistentes virtuais conversacionais amplamente conhecidos, a plataforma proporciona uma navegação intuitiva. A Figura 3 mostra a tela de cadastro, a qual permite ao estudante criar a sua conta.

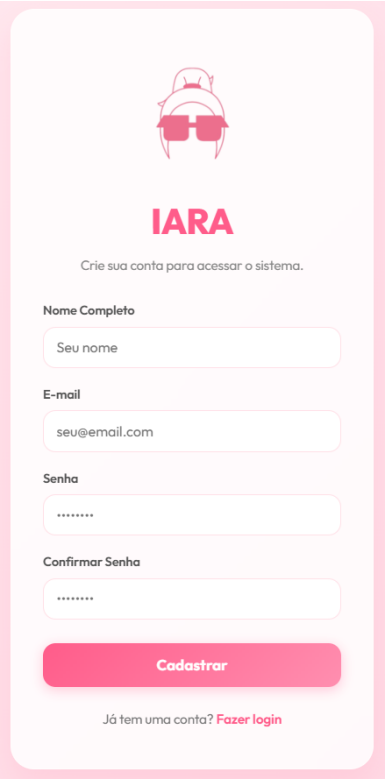


Figura 3. Tela de cadastro.



Figura 4. Tela de conversação, na qual o estudante pode interagir com a tutora inteligente.

Após a autenticação, o estudante é direcionado à tela principal. Ao iniciar uma sessão, o estudante acessa a interface de conversa, que possui um campo de texto inferior otimizado para inserção de perguntas ou blocos de código (Figura 4).

Antes da execução prática do exercício, o questionário de opinião sobre o uso de IA revelou um alto nível de adoção tecnológica, na qual a totalidade dos respondentes afirmou já utilizar ferramentas de IA tradicionais para entender conceitos e corrigir erros de código. Contudo, os estudantes demonstraram autocrítica em relação à dependência digital: nos relatos abertos, constataram que as inteligências artificiais tradicionais entregam soluções de forma muito passiva, o que impede a apreensão real da lógica. Foi destacado também que a maioria dos alunos tende a aceitar as respostas geradas como verdades absolutas, copiando o código-fonte de maneira automática sem compreender a estrutura do algoritmo.

Após a realização da atividade prática (focada na resolução de exercícios sobre vetores), procedeu-se à coleta de dados de satisfação e percepção geral da IARA. Embora o experimento em sala de aula tenha contado com a participação de 24 estudantes, a análise a seguir baseia-se nas respostas voluntárias de 14 discentes que completaram o questionário pós-uso. No quesito de satisfação e percepção geral sobre o uso da ferramenta, 100% das avaliações foram positivas, dividindo-se igualmente entre os conceitos “Boa” (50%) e “Muito boa” (50%).

O gráfico da Figura 5 mostra a finalidade de uso durante a aula. Os estudantes relataram ter utilizado a IARA majoritariamente para a resolução dos exercícios, esclarecimento de dúvidas conceituais e compreensão estrutural da arquitetura do código.

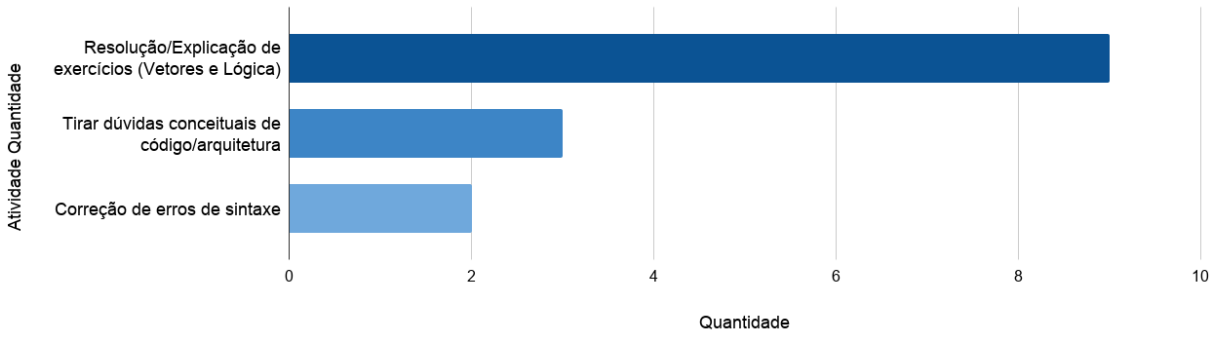


Figura 5. Principais finalidades de uso da IARA durante a aula prática.

Analisando qualitativamente os resultados, nota-se coerência entre a necessidade prévia apontada pelos alunos e a entrega de valor da ferramenta. A IARA foi recebida como uma solução direta para a lacuna deixada pelas IAs tradicionais.

Nos *feedbacks*, os estudantes afirmaram que a ferramenta atua de maneira eficaz na melhoria do raciocínio lógico, ensinando o passo a passo. Elogiaram a dinâmica do sistema, ressaltando que a IARA se destaca por explicar de forma minuciosa os conceitos teóricos antes de sugerir qualquer abordagem de solução. Apesar do elevado índice de aprovação, a avaliação honesta dos usuários revelou importantes pontos de melhoria, categorizados no gráfico da Figura 6.

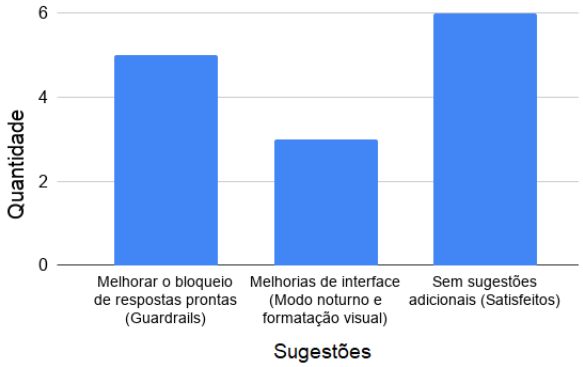


Figura 6. Principais sugestões de melhoria indicadas pelos usuários.

Sobre a interface gráfica, os participantes sugeriram a implementação de um modo noturno visando maior conforto visual e apontaram a necessidade de aprimorar a renderização de sintaxes devolvidas pela API. Já no aspecto estrutural do *back-end*, evidenciou-se uma limitação pertinente nas barreiras de proteção. Apesar da instrução diretiva para o modelo não entregar códigos prontos, estudantes relataram que, durante o esclarecimento de dúvidas, bastava exercer certa insistência no *prompt* para que a tutora acabasse cedendo e fornecesse o código-fonte do exercício. Esse relato comprova que as diretrizes primárias de restrição podem ser contornadas por *jailbreaks* conversacionais cotidianos.

Considerando observações durante o teste, foi possível perceber que os estudantes demonstraram entusiasmo ao utilizar uma nova ferramenta para auxiliar na resolução dos exercícios. De modo geral, os estudantes não apresentaram dificuldades em utilizar a IARA, principalmente em razão da interface possuir características semelhantes às de ferramentas conversacionais amplamente conhecidas, como o ChatGPT.

Além disso, observou-se que a proposta de uma ferramenta direcionada especificamente ao ensino de Algoritmos contribuiu para manter os estudantes focados no objetivo principal da atividade, que consistia em estudar e compreender melhor o conteúdo abordado. Durante a aula, também foram registrados comentários positivos dos alunos em relação ao projeto, especialmente pela iniciativa de desenvolver uma ferramenta voltada ao apoio dos estudos na área de Algoritmos.

CONCLUSÃO

Este trabalho apresentou o desenvolvimento e a validação empírica da IARA, evidenciando o potencial de um agente de Inteligência Artificial integrado especificamente ao estudo de Algoritmos. A plataforma demonstrou ser uma alternativa viável e preferível aos alunos em comparação às IAs tradicionais, fomentando o engajamento e atuando como uma mediadora do conhecimento. Ao instigar o raciocínio em vez de entregar códigos prontos, a ferramenta ataca diretamente o chamado "efeito caixa-preta", uma preocupação validada de forma autocrítica pelos próprios estudantes durante os testes.

Contudo, os testes práticos revelaram que a utilização de engenharia de *prompt* básica no *back-end* não é suficiente para conter o viés natural dos grandes modelos de linguagem em fornecer soluções completas, permitindo vazamentos de código sob insistência do usuário. Como trabalhos futuros, buscaremos testar e implementar novas formas de *guardrails* avançados.

Os próximos passos envolvem a exploração de Modelos de Linguagem Personalizados (Fine-Tuning), uma abordagem apontada pela literatura recente como eficaz para melhorar a precisão em aplicações educacionais e de geração de código (Raihan et al., 2025; Bassner, Frankford e Krusche, 2024), e a implementação da arquitetura RAG (Retrieval-Augmented Generation) (Hsing-Tzu et al., 2024). A integração via RAG permitirá que a IARA consulte exclusivamente os materiais didáticos da disciplina (como slides e apostilas do professor) antes de formular a resposta. Isso criará um contexto de domínio fechado, impedindo que ela forneça respostas fora da ementa ou entregue códigos prontos, garantindo que a tutoria seja estritamente focada no estímulo ao pensamento crítico e na real autonomia do estudante de tecnologia.

REFERÊNCIAS

BASSNER, P.; FRANKFORD, E.; KRUSCHE, S. Iris: An AI-Driven Virtual Tutor for Computer Science Education. In: Proceedings of the 2024 Innovation and Technology in Computer Science Education V. 1 (ITiCSE 2024). Milan, Italy: ACM, 2024. p. 394–400.

FERNANDEZ, A. S.; CORNELL, K. A. CS1 with a Side of AI: Teaching Software Verification for Secure Code in the Era of Generative AI. In: SIGCSE 2024 - Proceedings of the 55th ACM Technical Symposium on Computer Science Education. Nova York: ACM, 2024. v. 1, p. 345–351.

HSING-TZU, K.; YEN-KU, L.; YUN-CHENG, T.; SUMMIT, S. Enhancing Python Learning Through Retrieval-Augmented Generation: A Theoretical and Applied Innovation in Generative AI Education. In: Innovative Technologies and Learning: 7th International Conference (ICITL 2024). Tartu, Estonia: Springer, 2024. p. 164–173.

LIFFITON, M.; SHEESE, B.; SAVELKA, J.; DENNY, P. CodeHelp: Using Large Language Models with Guardrails for Scalable Support in Programming Classes. In: 23rd Koli Calling International Conference on Computing Education Research. Koli, Finland: ACM, 2023. p. 1-11.

LYU, W. et al. Evaluating the Effectiveness of LLMs in Introductory Computer Science Education: A Semester-Long Field Study. In: L@S '24: Proceedings of the Tenth ACM Conference on Learning @ Scale. Atlanta, GA, USA: ACM, 2024. p. 1-12.

MIGUEL, J.; MARTINS, W.; BENARRÓS, I.; DUARTE, J. C. Especialista em algoritmos para apoio interativo na aprendizagem de programação utilizando ChatGPT. In: V Simpósio Brasileiro de Educação em Computação (EduComp 2025). Manaus, AM, Brasil: SBC, 2025. p. 1-10.

OPENAI. OpenAI API Documentation. OpenAI, 2026. Disponível em: https://platform.openai.com/docs/. Acesso em: 28 maio 2026.

RAIHAN, N.; SIDDIQ, M. L.; SANTOS, J. C. S.; ZAMPIERI, M. Large Language Models in Computer Science Education: A Systematic Literature Review. Em: SIGCSE TS 2025. Pittsburgh, PA, USA: ACM, 2025. v. 1, p. 1-7.