

FARMA-ALG-IA (Sistema para a Promoção da Mediação do Erro no Ensino de Programação de Computadores com Inteligência Artificial)

Yuri Rafael Grajefe Feitosa^{1,2}, Marcos Alexandre Castilho², Alexander Robert Kutzke³

¹Departamento de Informática
Instituto Federal do Paraná - (IFPR)
Campus Astorga - PR.

²Programa de Pós-Graduação em Informática (PPGInf)
Universidade Federal do Paraná (UFPR)
Campus Curitiba - PR.

³Setor de Educação Profissional e Tecnológica (SEPT)
Universidade Federal do Paraná (UFPR)
Campus Curitiba - PR.

yuri.feitosa@ifpr.edu.br, marcos@inf.ufpr.br, alexander@ufpr.br

Abstract. *Generative Artificial Intelligence has transformed the way students interact with programming. While these technologies democratize access to knowledge and offer valuable support in the learning process, they also pose significant pedagogical challenges. Programming instruction, which relies on the development of logical reasoning, problem-solving, and constant practice, now faces the risk of superficiality and dependence on this technology. This paper discusses the problems that arise in programming instruction in this new scenario and proposes FARMA-ALG-IA (System for Promoting Error Mediation in Computer Programming Instruction with Artificial Intelligence).*

Resumo. *A Inteligência Artificial Generativa transformou a forma de como os estudantes interagem com a programação. Se, por um lado, essas tecnologias democratizam o acesso ao conhecimento e oferecem um suporte valioso no processo de aprendizado, por outro, impõem desafios pedagógicos importantes. O ensino de programação, que se apoia no desenvolvimento do raciocínio lógico, na resolução de problemas e na prática constante, passa a conviver com o risco da superficialidade e da dependência desta tecnológica. Este trabalho discute os problemas que surgem no ensino da programação diante desse novo cenário e propõe o FARMA-ALG-IA (Sistema para a Promoção da Mediação do Erro no Ensino de Programação de Computadores com Inteligência Artificial).*

1. Introdução

Na programação, a Inteligência Artificial (IA) está revolucionando a maneira de escrever algoritmos, tornando o desenvolvimento de software mais eficiente e acessível. Desde assistentes inteligentes até a geração automática de código, a IA já é uma ferramenta complementar para programadores. Entretanto, pode estar atrapalhando o processo de ensino e aprendizagem.

Segundo [Alexander et al. 2018] o crescimento na utilização da IA terá duas consequências impressionantes. Primeiro, a esfera educacional precisa se preparar para rápidas mudanças na vida profissional, especialmente em termos de competência profissional e necessidade de novos empregos. Em segundo, como resultado dos avanços na tecnologia de IA e suas aplicações no setor educacional, haverá mudanças nas técnicas pedagógicas em sala de aula.

No entanto, muitos problemas estão surgindo diante do uso de Inteligência Artificial Generativa (IAG) no contexto educacional. Em primeiro lugar, a IAG demonstra excelente desempenho na produção de soluções para exercícios introdutórios de programação [Denny et al. 2024]. Isso impacta diretamente na estratégia de ensino que muitos professores de programação introduzirão adotam: a resolução de problemas, exercícios e atividades avaliativas [Oliveira et al. 2023]. Em segundo lugar, os softwares que analisam a integridade das soluções produzidas por alunos iniciantes em programação são ineficazes para identificar códigos produzidos por uma IAG [Biderman and Raff 2022].

Outro problema é o plágio, que não se limita à cópia do trabalho de outra pessoa, pois compreende o uso de qualquer produto fruto da inteligência de uma pessoa sem creditar a devida autoria [Salvagno et al. 2023]. No ensino inicial de programação, o uso indevido de IAG pode ter consequências ainda piores, um déficit na aprendizagem de algoritmos pode gerar maiores dificuldades nas disciplinas posteriores e até na vida profissional do aluno.

Este trabalho começou a ser desenvolvido em Janeiro de 2025 (fazem nove meses) e o prazo para finalização é Maio de 2027 (21 meses). Faço um agradecimento em especial ao PROEX CAPES pelo apoio a pesquisa.

2. Motivação para a pesquisa

A abordagem tradicional baseada em aulas expositivas e exercícios práticos nem sempre é suficiente para promover uma aprendizagem profunda e duradoura [Junior et al. 2023]. É necessário explorar novas abordagens que incentivem a criatividade, a resolução de problemas autênticos e a colaboração entre os alunos [Kim et al. 2022].

A IA, especialmente os *chatbots* baseados em IA, como o ChatGPT, apresentam vantagens significativas no contexto de auxiliar a aprendizagem de programação. Essas ferramentas permitem uma interação personalizada e adaptativa, fornecendo suporte, *feedback* imediato e exemplos práticos de códigos aos alunos. É necessário equilibrar o uso da IA com a interação em sala de aula, garantindo uma abordagem pedagógica abrangente [Junior et al. 2023].

Na educação, a IA está fazendo uma transformação, desde a personalização do ensino até a automação de tarefas administrativas. Ela tem o potencial de revolucionar a educação, tornando-a mais personalizada, eficiente e acessível. No entanto, seu uso deve ser acompanhado de um olhar crítico para garantir que beneficie todos os alunos sem comprometer a qualidade e a equidade do ensino.

Um Juiz Online oferece problemas para serem resolvidos com a submissão de códigos-fonte, em uma linguagem de programação, e os corrige automaticamente usando casos de teste [Kurnia et al. 2001]. Cada caso de teste possui um conjunto de

entradas e suas respectivas saídas. Para verificar se a resposta de um problema é uma solução correta, um programa passa por seu respectivo conjunto de testes. Esses sistemas são muito utilizados em competições de programação, como a Maratona de Programação organizada pela Sociedade Brasileira de Computação (SBC), e vêm sendo gradativamente adaptados para o ensino de CS1 devido à funcionalidade de automatizarem a correção dos programas [Francisco et al. 2016].

O objetivo deste trabalho é fornecer uma ferramenta de *feedback* imediato, ilustrado e explicativo para o aluno que está aprendendo a programar, para isso é proposto o uso da IA na interface de desenvolvimento de código utilizada em aula.

3. Metodologia de Pesquisa

A integração da IAG nos primeiros passos da programação não é apenas para facilitar a compreensão de conceitos complexos e a aplicação prática deles, mas também estimular a capacidade dos alunos de resolver problemas de forma autônoma e criativa.

A aplicação da pesquisa será realizada na turma dos alunos ingressantes, da disciplina de Algoritmo I do Curso de Ciência da Computação da Universidade Federal do Paraná (UFPR), serão cinco turmas com média de quarenta alunos cada. Esses alunos serão divididos em dois grupos, o grupo de controle (duas turmas) que farão as atividades normalmente e o grupo experimental (três turmas), que farão as atividades com o auxílio da IA.

O grupo de controle deve seguir o currículo tradicional da disciplina de programação, utilizando métodos convencionais de ensino que incluem teoria, exemplos práticos e exercícios de programação, sem o auxílio de ferramentas de IAG. Este grupo é a base de comparação para avaliar os benefícios da introdução de tecnologias baseadas em IAG no processo de ensino-aprendizagem de programação.

Por outro lado, o grupo experimental terá uma abordagem diferenciada através da integração da IAG com a ferramenta de suporte ao aprendizado, que será apresentada na próxima seção. Os estudantes deste grupo terão *feedback* imediato sobre os exercícios de programação com explicações personalizadas e um fluxograma do algoritmo desenvolvido, deste modo, tendo uma perspectiva gráfica do código recém programado.

Após o término do semestre letivo, o desempenho de ambos os grupos serão comparados entre si e com turmas de semestres anteriores. Serão analisados dados como: evasão de alunos, notas, quantidade de exercícios resolvidos e taxas de assertividade.

4. API

O FARMA-ALG¹ é uma aplicação *Web* que visa a promoção da mediação do erro no ensino de programação de computadores. É a implementação para o domínio de programação do arcabouço de sistema. Esse arcabouço visa promover a mediação do erro por meio da visualização e manipulação de registros de respostas e de suas relações [Kutzke and Direne 2016].

Para a implementação do FARMA-ALG-IA², foi desenvolvido uma API (Application Programming Interface) para comunicação entre o FARMA-ALG e o *Large Lan-*

¹<http://farmaalg.c3sl.ufpr.br>

²<http://farma-alg-ia.c3sl.ufpr.br>

guage Model (LLM). Em caráter de teste, as primeiras requisições para o servidor foram enviadas localmente pelo Postman³, utilizando um método *POST*.

A figura 1 demonstra o primeiro resultado. O exercício proposto foi para o aluno fazer um algoritmo que retorne a quantidade de notas de cem reais necessárias para fornecer um determinado troco. Note que a coluna da direita, “Insights da Solução”, traz a resposta gerada pelo ChatGPT, o fluxograma e uma análise detalhada sobre o código.

The screenshot shows the FARMA-ALG-IA platform. On the left, there's a sidebar with 'Menu de operações' (Minhas respostas, Turmas, Mensagens) and a message: 'Você já respondeu corretamente esta questão. Sinta-se à vontade para realizar novas tentativas, mas a partir de agora, elas não incrementarão seu score nem surtirão qualquer efeito sobre o score desta questão.' Below this is a breadcrumb navigation: Turmas / Teste / Lista 1 - Questões / Exercício 079 - Responder. The main area has a heading 'Responder questão 'Exercício 079''. A section titled 'Enunciado:' contains the problem statement: 'calcular notasDeCem = valorTroco div 100'. To the right, a 'Insights da Solução' panel is open, showing:

Análise Visual

```

graph TD
    Inicio([Início]) --> LerLógica([Ler valorTroco])
    LerLógica --> Calcular([calcular notasDeCem = valorTroco div 100])
    Calcular --> Escrever([Escrever notasDeCem])
    Escrever --> Fim([Fim])
  
```

Comentários
Vamos analisar o seu código, que tem o objetivo de calcular a quantidade de notas de cem necessárias para fornecer um determinado valor de troco.

Erros no Código:
1. **Divisão Incorreta:** O divisor utilizado (`10`) parece estar incorreto, dado que o enunciado sugere que você está buscando notas de cem. Assim, o

Figura 1. Tela do Farma-Alg-IA

Fonte: Elaboração própria.

O Node.js foi escolhido como interface de desenvolvimento, pois é um ambiente de tempo de execução JavaScript de código aberto e multiplataforma que permite executar o código fora do navegador. Entretanto, o JavaScript era executado apenas no navegador para interatividade no lado do cliente. Com o Node.js, é possível utilizar o JavaScript para construir aplicações para servidores, ferramentas de linha de comando, *scripts* de automação e muito mais.

O LLM utilizado na API foi o ChatGPT na versão GPT-4o (“o”de “omni”), pois oferece uma capacidade de escrita criativa aprimorada, com uma escrita mais natural, envolvente e personalizada para melhorar a relevância e a legibilidade. Também é melhor para trabalhar com arquivos enviados, fornecendo *insights* mais profundos e respostas mais completas.

A comunicação entre o servidor e o ChatGPT foi realizada utilizando o Openrouter⁴, uma plataforma de API unificada, uma “API gateway”, que permite acessar e gerenciar LLMs, como os da OpenAI (GPT), Anthropic (Claude), Google (Gemini), Mistral

³<https://www.postman.com>

⁴<https://openrouter.ai/>

AI, Meta (Llama), entre outros, através de um único ponto de integração.

Neste projeto, o ChatGPT gera um *script* em DOT, e o Graphviz o renderiza em uma representação visual. Essa abordagem baseada em texto facilita a geração de grafos e o compartilhamento, o que por consequência, diminui significativamente o tempo de resposta entre Farma-Alg, API e o ChatGPT.

O fluxo do funcionamento da API, conforme ilustrado na figura 2, e funciona de seguinte maneira:

- Recebe um arquivo Pascal (.pas) do Farma-Alg no formato JSON, via HTTP.
- Envia o conteúdo do JSON para ChatGPT via OpenRouter, solicitando uma análise do código e a geração de um fluxograma em formato DOT.
- Salva a análise textual em um arquivo .txt na pasta “respostas”.
- Extrai o código DOT da resposta e gera uma imagem PNG do fluxograma utilizando o Graphviz na pasta “fluxogramas”.
- Codifica a imagem em Base64 e salva em um arquivo .txt na mesma pasta.
- Devolve no JSON o conteúdo da analise do algoritmo do aluno e o conteúdo do fluxograma em base64, conforme demonstra a imagem no apêndice E.

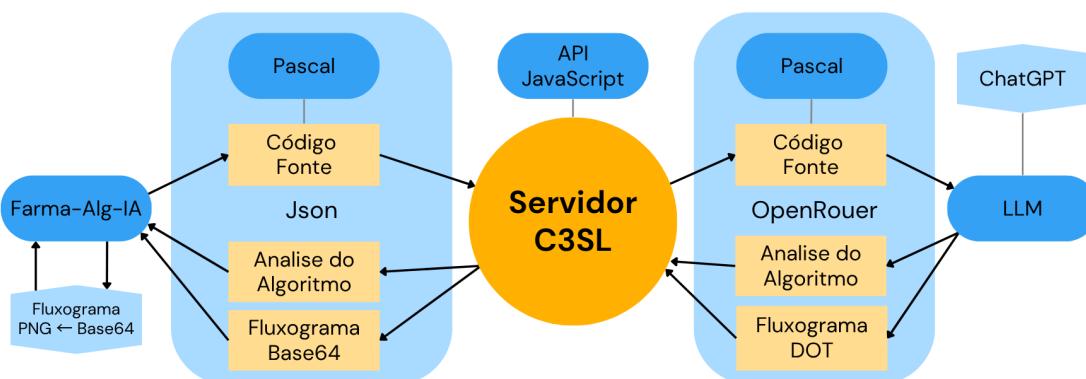


Figura 2. Fluxo de Funcionamento Farma-Alg-IA

Fonte: Elaboração própria.

Nos primeiros testes foram escolhidos três clássicos problemas da programação, a fórmula Bhaskara, números primos e o *Bubble Sort*. Aqui será detalhado apenas o exercício dos números primos, pois ele envolve estruturas de condição e repetição.

Um número primo é um número natural (inteiro e positivo), maior que 1, que possui apenas dois divisores positivos distintos, o número 1 e ele próprio. O algoritmo em Pascal que imprime todos os números primos entre 1 e 1000000, demonstrado no

apêndice A, foi enviado para o FARMA-ALG-IA analisar, supostamente, um código que o aluno está fazendo durante a aula de Algoritmos.

O fluxograma do Algoritmo anterior, gerado pelo FARMA-ALG-IA, está no apêndice B. A análise do algoritmo está no apêndice no apêndice C e arquivo DOT no apêndice D.

O prompt utilizado foi: *Você é um professor experiente de Ciência da Computação. Sua tarefa é analisar códigos Pascal fornecidos por alunos, identificando erros, sugerindo melhorias e explicando o funcionamento do algoritmo de forma didática e detalhada, mas não deve fornecer o código correto para o aluno, deve induzi-lo a pensar na melhor forma de resolver o problema. Além disso, forneça a representação do fluxograma do algoritmo em formato DOT.*

5. Resultados Esperados

Para medir a opinião dos participantes, será utilizada a escala Likert de cinco pontos. Essa escala permite classificar as respostas em níveis quantitativos de acordo com afirmações, variando de “concordo totalmente” a “discordo totalmente”. Baseado no uso da inteligência artificial generativa no ensino introdutório de programação:

- QP01: De que forma o feedback mediado por IAG e a visualização do fluxograma influenciam a capacidade de alunos iniciantes entenderem melhor a programação?
- QP02: O uso do FARMA-ALG-IA promove um aprendizado mais profundo e reduz a dependência dos alunos em relação à tecnologia, em comparação com o uso direto de uma IAG?
- QP03: A mediação do erro, promovida pelo sistema FARMA-ALG-IA, impacta positivamente na capacidade de estudantes iniciantes de programação em depurar algoritmos e desenvolver o pensamento computacional melhor?

Espera-se que o uso da IAG possa contribuir com o ensino introdutório de programação utilizando o FARMA-ALG-IA.

6. Considerações Finais

A IA oferece benefícios como auxílio na resolução de problemas e geração de código, mas também apresenta riscos como dependência, aprendizado superficial e plágio. A falta de regulamentação e treinamento adequado para educadores são desafios significativos.

A combinação de juízes online com IA tem o potencial de revolucionar o ensino de programação, oferecendo *feedback* personalizado, análise de código semântica e representações visuais como fluxogramas. A integração do FARMA-ALG com o ChatGPT é um exemplo concreto dessa sinergia.

A pesquisa aborda a crescente integração da IA no ensino de programação, destacando tanto as oportunidades quanto os desafios. A IA pode ser uma ferramenta poderosa para personalizar o aprendizado, fornecer *feedback* detalhado e auxiliar na visualização de conceitos complexos.

A integração bem-sucedida da IA no ensino de programação requer uma abordagem equilibrada que aproveite seus benefícios, mitigando seus riscos e garantindo que a tecnologia sirva como um catalisador para um aprendizado melhor.

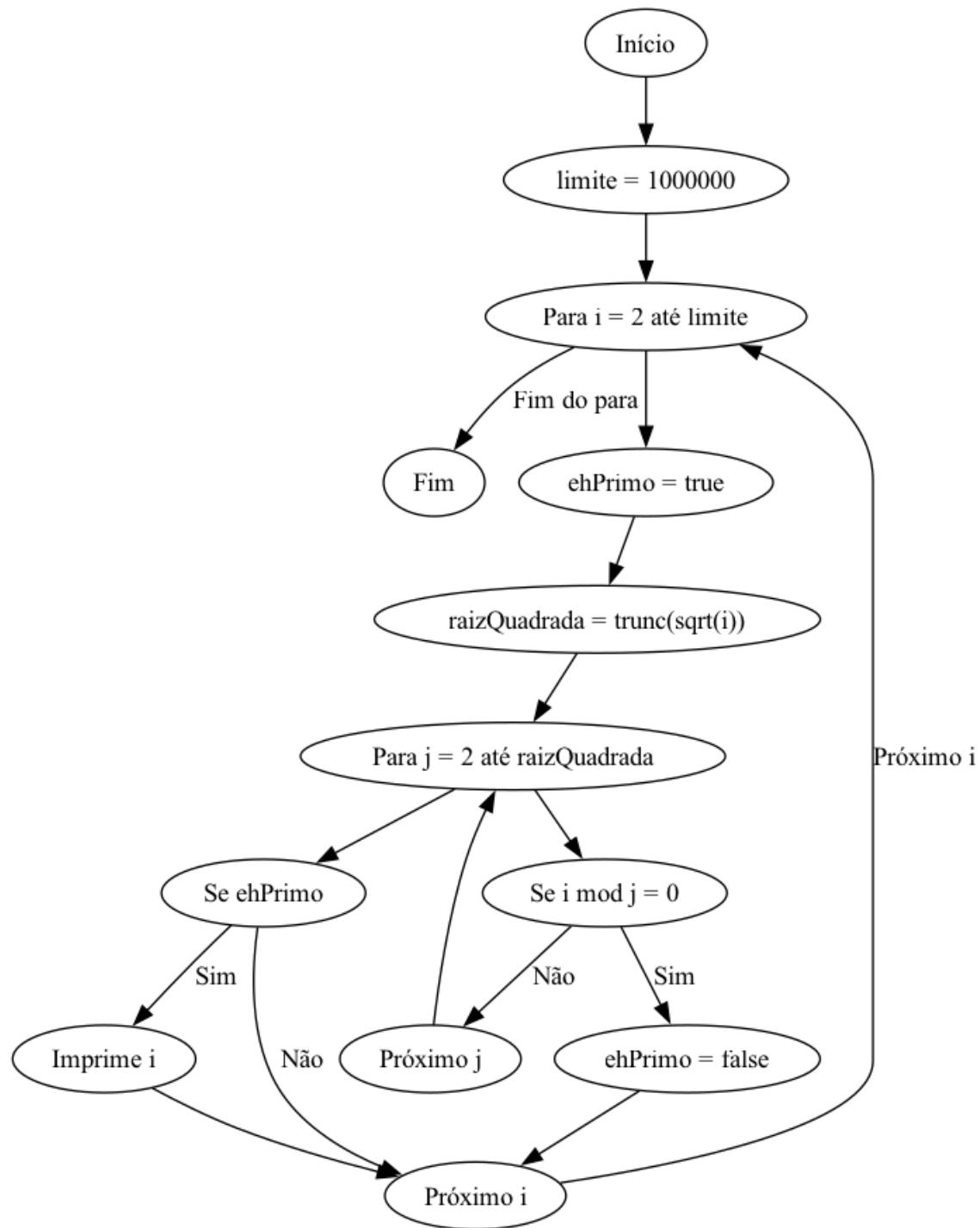
Referências

- Alexander, T., Stefanova, V., and Zahidi, S. (2018). The future of jobs report 2018. cologne/geneva (switzerland): World economic forum.
- Biderman, S. and Raff, E. (2022). Fooling moss detection with pretrained language models. In *Proceedings of the 31st ACM international conference on information & knowledge management*, pages 2933–2943.
- Denny, P., Leinonen, J., Prather, J., Luxton-Reilly, A., Amarouche, T., Becker, B. A., and Reeves, B. N. (2024). Prompt problems: A new programming exercise for the generative ai era. In *Proceedings of the 55th ACM Technical Symposium on Computer Science Education V. 1*, pages 296–302.
- Francisco, R., Júnior, C. P., and Ambrósio, A. P. (2016). Juiz online no ensino de programação introdutória-uma revisão sistemática da literatura. In *Brazilian Symposium on Computers in Education (Simpósio Brasileiro de Informática na Educação-SBIE)*, volume 27, page 11.
- Junior, S. S., Freitas, R., Moraes, M., and Costa, D. (2023). Chatgpt no auxílio da aprendizagem de programação: Um estudo de caso. In *Anais do XXXIV Simpósio Brasileiro de Informática na Educação*, pages 1375–1384, Porto Alegre, RS, Brasil. SBC.
- Kim, J., Lee, H., and Cho, Y. H. (2022). Learning design to support student-ai collaboration: Perspectives of leading teachers for ai in education. *Education and information technologies*, 27(5):6069–6104.
- Kurnia, A., Lim, A., and Cheang, B. (2001). Online judge. *Computers & Education*, 36(4):299–315.
- Kutzke, A. R. and Direne, A. (2016). Mediação do erro no ensino de programação de computadores: fundamentos e aplicação da ferramenta farma-alg. *Anais dos Workshops do Congresso Brasileiro de Informática na Educação*.
- Oliveira, K. K. d. S., Marcolino, A. d. S., Deus, W. S. d., Falcão, T. P. d. R., and Barbosa, E. F. (2023). Pensamento computacional na programação introdutória e habilidades do século xxi: um mapeamento sistemático da literatura. *Revista Novas Tecnologias na Educação-RENOTE*, 21(2):519–531.
- Salvagno, M., Taccone, F. S., and Gerli, A. G. (2023). Can artificial intelligence help for scientific writing? *Critical care*, 27(1):75.

A. Algoritmo dos Números Primos em Pascal

```
1 program Primos1AMilhao;
2   var
3     i, j, limite, raizQuadrada: integer;
4     ehPrimo: boolean;
5   begin
6     limite := 1000000;
7     writeln('Números primos entre 1 e ', limite, ':');
8     for i := 2 to limite do
9       begin
10         ehPrimo := true;
11         raizQuadrada := trunc(sqrt(i));
12         for j := 2 to raizQuadrada do
13           begin
14             if (i mod j = 0) then
15               begin
16                 ehPrimo := false
17                 break;
18               end;
19             end;
20             if ehPrimo then
21               begin
22                 write(i, ' ');
23               end;
24             end;
25   writeln;
26   readln;
27 end.
```

B. Fluxograma do Algoritmo dos Números Primos



C. Explicação sobre o Algoritmo dos Números Primos

O código Pascal apresentado tem a intenção de listar todos os números primos entre 1 e 1.000.000. Vamos analisar o código, identificar erros, dar sugestões de melhorias e fornecer uma explicação detalhada de seu funcionamento.

Análise do Código

1. ****Variáveis e Inicialização**:** - ‘limite’: Define até onde os números primos serão procurados. Está corretamente definido como 1.000.000. - ‘ehPrimo’: Variável booleana utilizada para verificar se um número é primo ou não. - ‘raizQuadrada’: Utilizado para calcular até onde devemos verificar se ‘i’ é divisível.
2. ****Laço de Repetição (FOR)**:** - Começa em 2, pois 1 não é primo. - Vai até ‘limite’, que é 1.000.000.
3. ****Verificação de Primalidade**:** - Calcula a ‘raizQuadrada’ do número ‘i’ com ‘trunc(sqrt(i))’. - Verifica se ‘i’ é divisível por qualquer número entre 2 e ‘raizQuadrada’.
4. ****Erro no Código**:** - A função ‘break’ não é válida em Pascal. Para sair do loop antecipadamente, você pode usar uma instrução ‘goto’ para quebrar o loop, ou simplesmente sair da sua lógica usando um controle adequado.
5. ****Print dos Resultados**:** - ‘write’ é utilizado para escrever os números primos na mesma linha. - ‘writeln’ é chamado ao final para uma nova linha.

Explicação do Funcionamento

1. ****Início do Programa**:** O programa começa definindo variáveis necessárias para os cálculos.
2. ****Loop Principal**:** O loop ‘for’ percorre cada número de 2 até 1.000.000. - Para cada número ‘i’, inicialmente supomos que ele é primo (‘ehPrimo := true’).
3. ****Verificação de Primalidade**:** - Calculamos a raiz quadrada de ‘i’. - Usamos um ‘while’ loop para verificar divisibilidade de ‘i’ por cada número ‘j’ até a raiz quadrada. - Se ‘i’ for divisível por ‘j’, ‘ehPrimo’ é ajustado para ‘false’.
4. ****Impressão**:** Números que continuam com ‘ehPrimo = true’ são impressos como primos.

D. Arquivo DOT do Algoritmo dos Números Primos

```

digraph G {
    node [shape=box];
    Start [label="Start"];
    Init [label="limite := 1000000"];
    Writeln1 [label="writeln('Números primos entre 1 e ', limite)"];
    ForLoopMain [label="for i := 2 to limite do"];
    InitializeEhPrimo [label="ehPrimo := true"];
    CalculateRoot [label="raizQuadrada := trunc(sqrt(i))"];
    ForLoopInner [label="for j := 2 to raizQuadrada do"];
    CheckDivisibility [label="if (i mod j = 0)"];
    SetEhPrimoFalse [label="ehPrimo := false"];
    Break [label="break"];
    CheckEhPrimo [label="if ehPrimo then"];
    WritePrime [label="write(i, ' ')"];
    End [label="End"];

    Start -> Init;
    Init -> Writeln1;
    Writeln1 -> ForLoopMain;
    ForLoopMain -> InitializeEhPrimo;
    InitializeEhPrimo -> CalculateRoot;
    CalculateRoot -> ForLoopInner;
    ForLoopInner -> CheckDivisibility;
    CheckDivisibility -> SetEhPrimoFalse [label="True"];
    CheckDivisibility -> CheckEhPrimo [label="False"];
    SetEhPrimoFalse -> Break;
    Break -> ForLoopMain;
    CheckEhPrimo -> WritePrime [label="True"];
    CheckEhPrimo -> ForLoopMain [label="False"];
    WritePrime -> ForLoopMain;
    ForLoopMain -> End [label="Complete"];
}

```

E. Requisição POST utilizando o Postman

HTTP API IA / pascal - primo

POST

Params Authorization Headers (9) **Body** Scripts Settings

none form-data x-www-form-urlencoded raw binary GraphQL **JSON**

```

1 {
2   "linguagem": "pascal",
3   "codigo": "program Primos1AMilhao; \n \n var \n i, j, limite, raizQuadrada: integer;\n \n for i := 2 to limite do \n begin \n ehPrimo := true; \n raizQuadrada := trunc(s\n break; \n end; \n end; \n if ehPrimo then \n begin \n write(i, ' '); \n end; \n en\n4 }
5

```

Body Cookies Headers (8) Test Results |

{ } JSON ▾ Preview Visualize |

```

1 {
2   "mensagem": "Código Pascal analisado com sucesso!",
3   "análise": "Vamos analisar o código Pascal fornecido. O objetivo do programa é li\nsugerir algumas melhorias.\n#\nO que o código faz:\n1. **Inicialização**\nindicando que listará os números primos dentro desse intervalo.\n2. **Loop**\ngeralmente não considerado primo, então começar em 2 é apropriado.\n3. **Verificação**\n(`ehPrimo := true'). Depois calcula a raiz quadrada truncada de `i` para otimizar `j`\nde 2 até a raiz quadrada de `i`. Se `i` for divisível por qualquer `j`, `ehPrimo` é\natualizado para `false`. Se `i` for confirmado como primo (ou seja, `ehPrimo` permanece `true`), o\nnúmero é adicionado à lista de primos.\n4. **Output**\nPontos de melhorias e observações:\n- **Uso da Fórmula Quadrática:**\ncaso contrário, o código gerará um erro. Assegure-se de verificar essa parte.\n- **Loop interno:**\nloop interno sem `break`. O loop é desnecessário.\n- **Desempenho:**\nEmbora otimizado, esse algoritmo ainda é lento para grandes intervalos.\n- **Output para Grandes Intervalos:**\nConsidere ambientes de linha de comando que não suportam tão grandes volumes de dados.\n5. **Imagem Base64:**\nA imagem base64 contém o resultado da análise e o código original codificado em base64.\n
```