

SEMANA ACADÊMICA · ADS · MÓDULO 1 · 2026.2

Fundamentos importam na era da IA

Por que dominar lógica e Python continua sendo o seu maior diferencial — mesmo quando a máquina escreve o código por você.

Prof. José William Firmino Bezerra

Fundamentos da Programação · Faculdade de Tecnologia Senac RJ

2 horas

Quem está falando com vocês



**José William
Firmino Bezerra**

- **Professor** Faculdade de Tecnologia Senac RJ
- **Coordenador** CST em IA e Ciência de Dados
- **Engenheiro de software** Educpay

FORMAÇÃO

Engenharia de Software

Gestão da Tecnologia da
Informação

Técnico em Informática

PÓS-GRADUAÇÃO E MBA

Pós

Projetos de Cloud Computing

MBA

Web 3.0

MBA

Big Data e Inteligência Competitiva

Pós

Data Science

MBA

Segurança da Informação

Pós

IA Aplicada a Growth Marketing

Pós

Análise de Dados

Pós

Privacidade e Proteção de Dados Pessoais

Pós

Engenharia de Software

Pós

Tecnologias Aplicadas à Educação

Pós

Desenvolvimento Full Stack

Pós

Projetos de Aplicativos Móveis Multiplataforma

Eu escrevo código com IA todo dia. E é exatamente por isso que estou aqui defendendo os fundamentos.

Como vamos usar as próximas 2 horas

1

Abertura

Quem sou eu e por que esse tema

2

O elefante na sala

A IA vai me substituir?

3

Como a IA escreve código

O que acontece por trás do prompt

4

Atividade 1 • Caça ao bug

3 códigos de IA que rodam e estão errados

5

Intervalo

Café, água e conversa

6

Os 5 fundamentos

O que a IA não faz por você

7

Atividade 2 • Você é o arquiteto

Resolver um problema sem escrever código

8

A jornada da UC + perguntas

O que vem pela frente no semestre

Duas atividades práticas, um intervalo e muita conversa. Sentem perto de alguém — você vai precisar de dupla.

A PERGUNTA QUE TODO CALOURO FAZ EM 2026

**“Se a IA já escreve o código...
por que eu ainda preciso
aprender a programar?”**

Guarde essa pergunta. Nós vamos respondê-la juntos — e no final desta palestra a resposta vai ser sua, não minha.

O cenário real do mercado

~76%

dos desenvolvedores usam ou pretendem usar ferramentas de IA no fluxo de trabalho

Stack Overflow Developer Survey, 2024

~92%

dos desenvolvedores nos EUA já usavam ferramentas de IA para escrever código

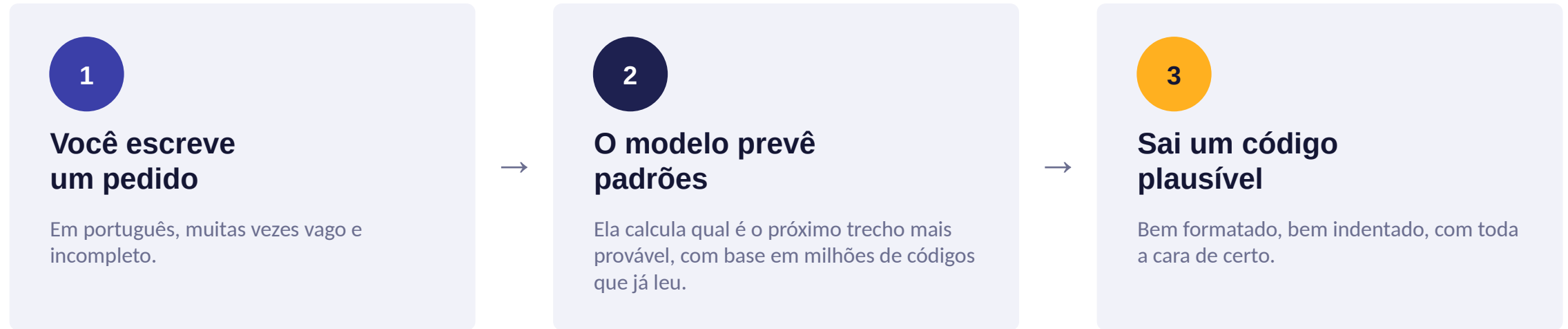
Pesquisa GitHub, 2023

A vaga não desapareceu.

Ela mudou de exigência. Hoje ninguém paga para digitar código — paga para garantir que o código está certo.

A IA não tirou o programador do mercado. Ela tirou do mercado o programador que só sabia copiar, colar e torcer para funcionar.

O que a IA realmente faz quando gera código



Repare no que NÃO aconteceu nesse processo: **ela não executou o código, não testou, não conhece a sua regra de negócio e não sabe se acertou.**

Plausível ≠ Correto

Onde ela brilha e onde ela falha

ONDE ELA BRILHA

- Código repetitivo e estrutura inicial de projeto
- Sintaxe e comandos que você esqueceu na hora
- Traduzir código de uma linguagem para outra
- Explicar um código que outra pessoa escreveu
- Gerar um primeiro rascunho para você melhorar
- Sugerir testes básicos que você não pensou

ONDE ELA FALHA

- Regra de negócio que só existe na sua empresa
- Casos de borda: lista vazia, valor negativo, campo em branco
- Decisões de arquitetura e escolhas de longo prazo
- Entender o usuário real do outro lado da tela
- Saber se o resultado que ela mesma gerou está certo
- Assumir a responsabilidade quando der errado

A coluna da esquerda economiza o seu tempo. A coluna da direita é onde você constrói a sua carreira.

ATIVIDADE 1

Caça ao bug

Três códigos gerados por IA. Todos rodam sem nenhum erro na tela. Todos estão errados.

Em dupla

Sente com o colega do lado.
Ninguém resolve sozinho.

Um de cada vez

Leiam com calma antes de
responder.

Sem executar

Nada de computador. Só leitura
e raciocínio.

Sem saber Python

Vocês não precisam. Precisam
pensar.

Vocês vão achar os três. Sem computador.

“Faz um código que classifica a nota do aluno”

```
def classificar(nota):  
    if nota >= 5:  
        return "Aprovado"  
    elif nota >= 7:  
        return "Aprovado com destaque"  
    else:  
        return "Reprovado"  
  
print(classificar(9.5))
```

A regra do Senac

- Abaixo de 5: reprovado
- De 5 a 6.9: aprovado
- De 7 para cima: aprovado com destaque

Sua tarefa

O aluno tirou 9.5. O que vai aparecer na tela? E o que deveria aparecer?

Dica: o Python lê as condições de cima para baixo e para na primeira que for verdadeira.

O bug: a ordem das condições

O QUE APARECE NA TELA

Aprovado

O QUE DEVERIA APARECER

Aprovado com destaque

Com nota 9.5 a primeira condição ($\text{nota} \geq 5$) já é verdadeira. O Python retorna ali e nunca chega no elif. A linha do “destaque” é código morto: ela nunca vai executar, para nenhum aluno da faculdade.

```
# a correção
if nota >= 7:
    return "Aprovado com destaque"
elif nota >= 5:
    return "Aprovado"
```

A IA acertou 100% da sintaxe e errou 100% da regra.
Quem só olha se “rodou” entrega esse código.

“Calcula a média das notas da turma”

```
def media_da_turma(notas):  
    total = 0  
    for nota in notas:  
        total = total + nota  
    return total / len(notas)  
  
turma_a = [7.0, 8.5, 6.0]  
turma_b = [] # turma nova, sem notas  
  
print(media_da_turma(turma_a))  
print(media_da_turma(turma_b))
```

O contexto

Esse código vai para o sistema acadêmico. No primeiro dia do semestre, nenhuma turma tem nota lançada ainda.

Sua tarefa

A turma A funciona. E a turma B? O que acontece com o sistema inteiro nesse momento?

Dica: len(notas) devolve quantos itens tem na lista. Quantos itens tem uma lista vazia?

O bug: o caso de borda que ninguém imaginou

ZeroDivisionError: division by zero

A turma A calcula certinho. A turma B derruba a aplicação inteira. A IA escreveu um código correto para o caso que ela imaginou — e o caso que ela imaginou foi o mais fácil. Ela não sabia que existe um primeiro dia de semestre, porque ela não conhece o Senac.

```
def media_da_turma(notas):  
    if len(notas) == 0:  
        return None  
    return sum(notas) / len(notas)
```

Retornar 0, None ou “sem notas”? Isso não é decisão de código. É decisão de negócio — e quem toma é você, conversando com quem usa o sistema.

Caso de borda é a situação rara que ninguém lembra — e que sempre acontece em produção.

“Aplica os descontos do cliente na compra”

```
preco = 100.00
cliente_vip = True
cupom_promocional = True

if cliente_vip:
    preco = preco - (preco * 0.10)

if cupom_promocional:
    preco = preco - (preco * 0.20)

print(preco)
```

A regra da empresa

- Cliente VIP tem 10% de desconto
- O cupom promocional dá mais 20%
- Somando, o cliente tem 30% de desconto

Sua tarefa

Pela regra, o cliente deveria pagar R\$ 70,00. Faça a conta na mão: quanto esse código cobra dele?

Este código não dá erro nenhum. Roda, imprime um número e vai direto para produção.

O bug silencioso: o mais caro de todos

R\$ 70,00

o que a regra manda cobrar

R\$ 72,00

o que o código realmente cobra

R\$ 2,00

de erro. Em cada venda. Todo dia.

Os descontos foram aplicados em cascata: os 20% incidiram sobre os R\$ 90,00 que já tinham sido reduzidos, e não sobre os R\$ 100,00 originais. O desconto real foi de 28%, não de 30%. Nenhum erro de sintaxe, nenhuma mensagem vermelha, nenhum sistema fora do ar — e a empresa perdendo dinheiro por meses, até alguém do financeiro perceber.

Nenhum teste automático pega esse bug. Nenhuma IA pega esse bug. Só pega quem conhece a regra e sabe ler o código.

O que os três bugs têm em comum

1

Nenhum era erro de sintaxe

Justamente a parte que a IA domina melhor do que qualquer um de nós.

2

Todos eram erro de raciocínio ou de contexto

Justamente a parte que só pode ser sua, porque depende de conhecer o problema real.

3

Todos foram encontrados lendo, não rodando

Vocês não usaram computador nenhum durante essa atividade inteira.

**Você não precisa ser mais rápido que a IA para escrever código.
Precisa ser melhor que ela para julgar o código.**

INTERVALO

15 minutos

Na volta: os 5 fundamentos que fazem de você alguém que a IA não substitui.

BLOCO 3

Os 5 fundamentos que te tornam insubstituível

Nenhum deles se aprende pedindo pronto. Todos eles se aprendem errando — e por isso estão no plano da nossa unidade curricular.

As 5 habilidades que a IA não entrega pronta

1

Lógica e pensamento computacional

Quebrar um problema grande em passos pequenos, claros e na ordem certa.

2

Leitura crítica de código

Bater o olho e saber se aquilo faz o que promete, antes de executar.

3

Depuração

Encontrar a causa do erro, e não ficar tratando o sintoma.

4

Modelagem do problema

Traduzir uma regra do mundo real para dados, condições e repetições.

5

Responsabilidade técnica

O nome que assina o commit é o seu. A IA não é chamada na reunião.

Repare: nenhuma das cinco é “decorar comandos do Python”. Essa parte a IA faz. As cinco de cima, não.

Pensamento computacional na prática

1

Decompor

Quebrar o problema em partes menores

Buscar as notas • calcular a média • comparar com o mínimo • avisar o aluno

2

Reconhecer padrões

Ver o que se repete entre as partes

Todo aluno passa pelo mesmo cálculo — isso vai virar um laço

3

Abstrair

Ignorar o que não importa agora

Nome e foto do aluno não mudam a média. Fora do problema.

4

Algoritmizar

Escrever a sequência exata de passos

Some as notas, divida pela quantidade, compare com 5

Exemplo aplicado: “avisar automaticamente todo aluno que está em risco de reprovar”.

Nada disso é Python. Tudo isso vem antes do Python.

Quem entende de programação pede melhor

PROMPT FRACO

“faz um código de média de notas em python”

- Não diz o que fazer com lista vazia
- Não diz quantas casas decimais
- Não pede teste nenhum
- Você não tem como saber se veio certo

PROMPT FORTE

“Escreva uma função Python que receba uma lista de notas de 0 a 10 e devolva a média com 2 casas decimais. Se a lista estiver vazia, devolva None. Sem bibliotecas externas. Inclua 3 casos de teste, sendo um com lista vazia.”

Tipo de dado, faixa de valores, formato do retorno, caso de borda, restrição e verificação.

A diferença entre os dois prompts não está na IA. Está no que você sabe.

ATIVIDADE 2

Você é o arquiteto

Grupos de 4 pessoas · discussão em grupo · apresentação para a turma

Regra única: não é permitido escrever uma linha de código.

O problema

“O Senac quer um sistema que avise automaticamente o aluno que está em risco de reprovação.”

1

Quais dados o sistema precisa?

Notas? Presença? Entregas? De onde eles vêm?

2

Qual regra define “em risco”?

Escrevam a regra em português, com números.

3

Quais casos de borda quebram a regra?

Aluno que entrou depois, turma sem nota, trancamento.

4

Quem decide essa regra?

Vocês, a coordenação, o aluno — ou a IA?

Nenhuma dessas quatro perguntas a IA responde por você. E sem respondê-las, não existe código que preste.

O papel mudou. Ele não acabou.

ANTES

O programador escrevia o código.

AGORA

O programador define, revisa e responde.



Arquiteto

Entende o problema, conversa com quem vai usar e decide como a solução deve funcionar antes de ela existir.



Revisor

Lê tudo que a IA produz com olhar crítico, testa os casos de borda e recusa o que não serve.



Responsável

Assina o resultado. Quando o sistema erra, é o profissional que explica, corrige e responde.

Os três papéis exigem entender de código. Nenhum deles exige digitar rápido.

A jornada de Fundamentos da Programação

1

Lógica e algoritmos

Pensar a solução antes de digitar qualquer coisa

2

Variáveis e tipos de dados

Como o computador guarda e enxerga a informação

3

Condicionais

Ensinar o programa a decidir — e na ordem certa

4

Repetições

Fazer a máquina trabalhar por você, mil vezes, sem reclamar

5

Funções

Organizar, nomear e reaproveitar o que você já resolveu

6

Listas e dicionários

Trabalhar com muitos dados de uma vez

7

Projeto final

Um problema real, resolvido do zero, defendido por você

**Aulas começam em 17/08.
Traga o notebook e a vontade de errar.**

Como usar IA nesta UC sem virar refém dela

1

Tente sozinho primeiro. Sempre.

O tempo que você passa travado é exatamente onde o aprendizado acontece.

2

Use a IA para explicar, não para entregar.

“Me explique por que isso dá erro” vale muito mais que “me dá o código pronto”.

3

Nunca entregue código que você não sabe defender.

Se não consegue explicar linha por linha, esse código ainda não é seu.

4

Peça o porquê, não só o como.

Pergunte alternativas, peça que ela justifique a escolha e questione a resposta.

5

Rode, teste e tente quebrar o que ela deu.

Lista vazia, número negativo, texto no lugar de número. Toda vez.

Nesta unidade curricular a IA é permitida — desde que você consiga defender cada linha que entregar.

RETOMANDO A PERGUNTA DO INÍCIO

“Se a IA já escreve o código, por que eu ainda preciso aprender a programar?”

Porque alguém precisa saber se o código está certo.

E, a partir do dia 17 de agosto, esse alguém é você.

Prof. José William Firmino Bezerra

Fundamentos da Programação · ADS Módulo 1 · Faculdade de Tecnologia Senac RJ

Perguntas?