@fpftech.educacional

FPFtech
Escola Tecnológica

FPFtech
Escola Tecnológica

Exemplos de algoritmo (Atividade)

Em grupos, criem algoritmos que:

1. Escolher um restaurante para jantar
2. Definir um caminho para ir ao trabalho
3. Fazer um ovo frito
4. Preparar café
5. Organizar a agenda da semana
6. Escolher uma roupa para sair
7. Escolher uma sobremesa no self-service
8. Escovar os dentes
9. Planejar uma viagem de férias
10. Comprar um bilhete de cinema
11. Fazer compras no mercado
12. Ligar um carro

Representação De Algoritmos

Formas de Representação

Linguagem natural: Escrito em português, sem formatação específica.

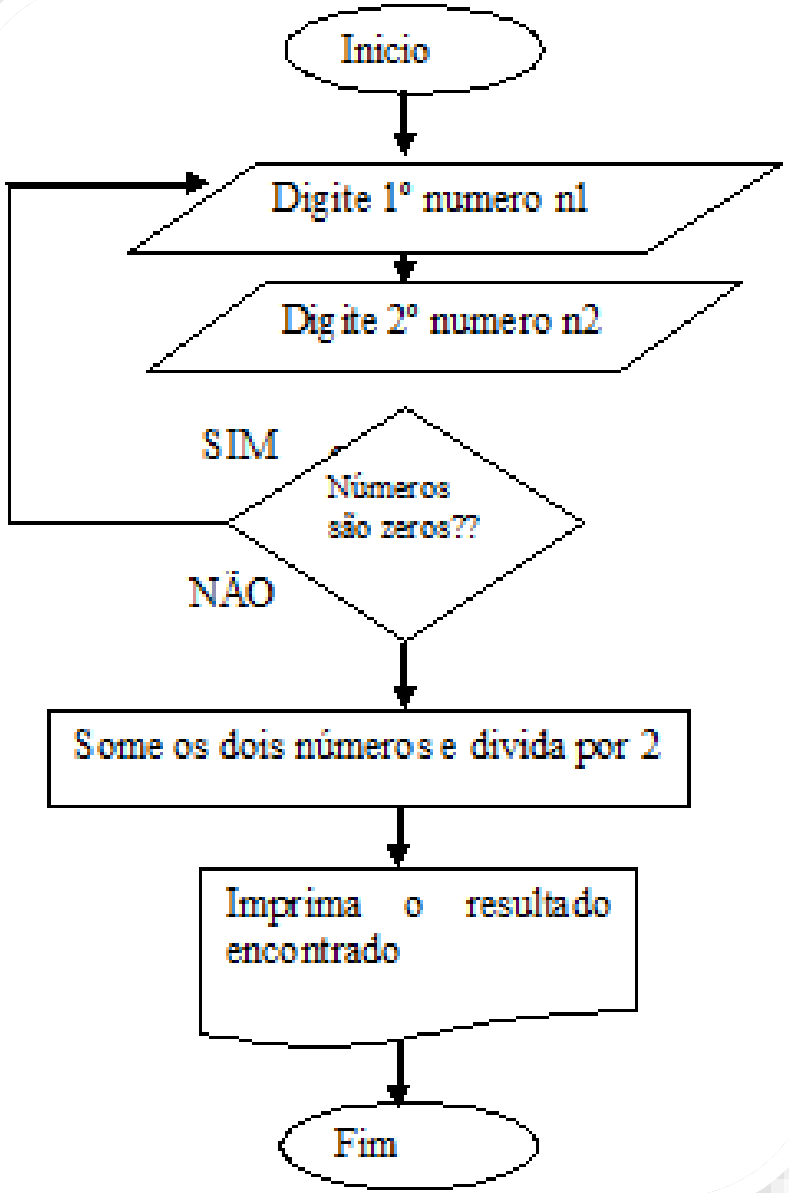
Fluxograma: Representação gráfica com símbolos para cada etapa.

Pseudocódigo: Representação estruturada próxima da programação real.

Fluxograma

Representação gráfica de um algoritmo.

excalidraw/drawio



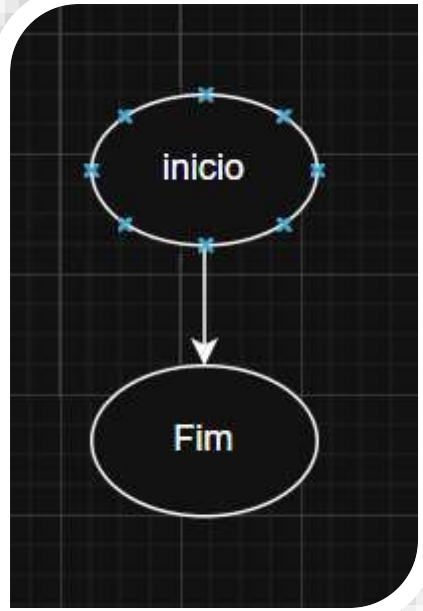
Principais Símbolos do Fluxograma

Símbolo de Início e Fim (Terminal)

Forma: Oval

Função: Representa o início e o fim do algoritmo.

Cada fluxograma deve ter um único início e pelo menos um fim.



Principais Símbolos do Fluxograma

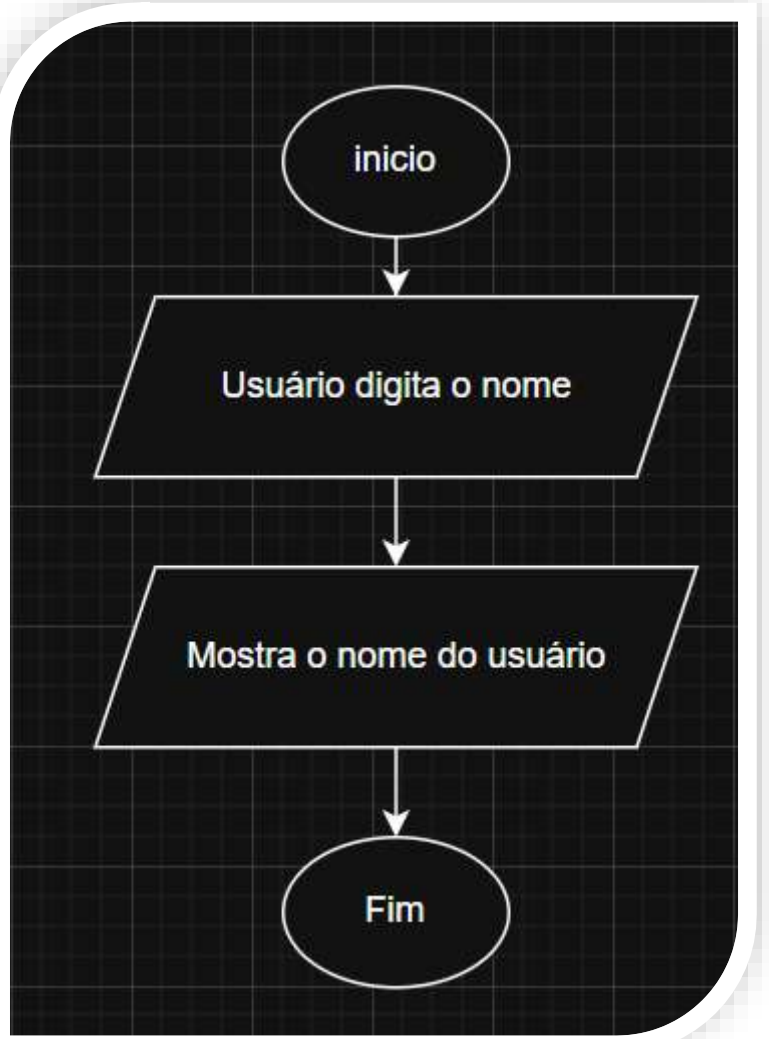
Símbolo de Entrada e Saída de Dados

Forma: Paralelogramo

Função: Representa a entrada de dados pelo usuário ou a exibição de informações.

Exemplo de Entrada: O usuário insere um número no sistema.

Exemplo de Saída: O sistema exibe um resultado na tela.



Principais Símbolos do Fluxograma

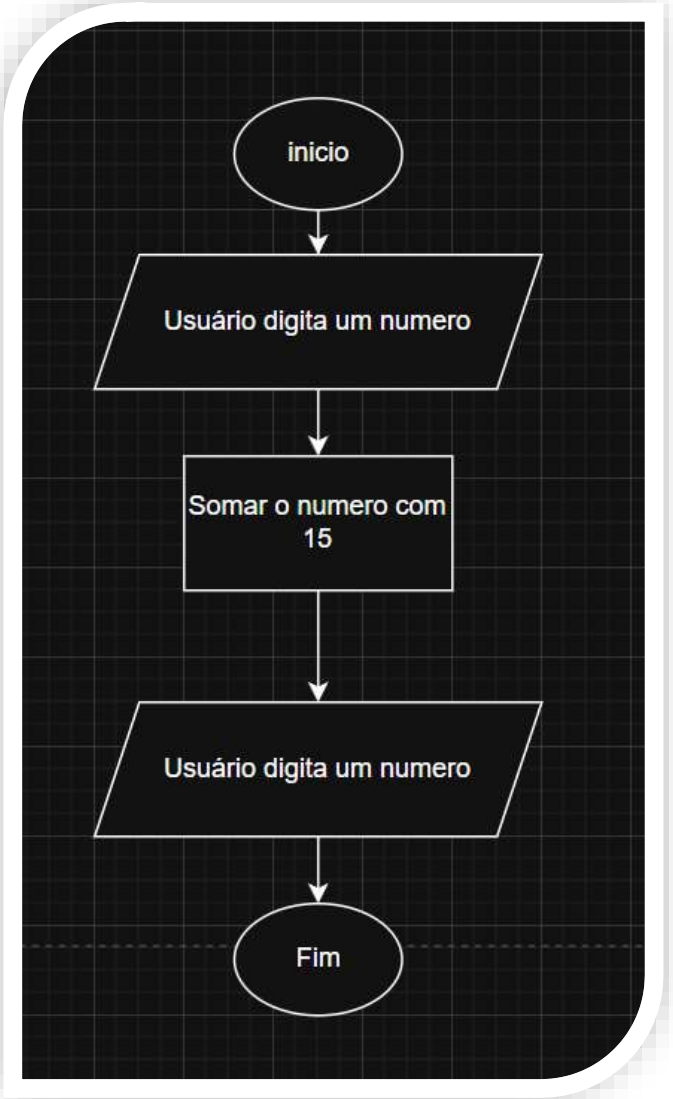
Símbolo de Processamento

(Atribuição e Cálculos)

Forma: Retângulo

Função: Representa qualquer tipo de cálculo ou processamento de dados.

Exemplo: Somar dois números, armazenar um valor em uma variável.



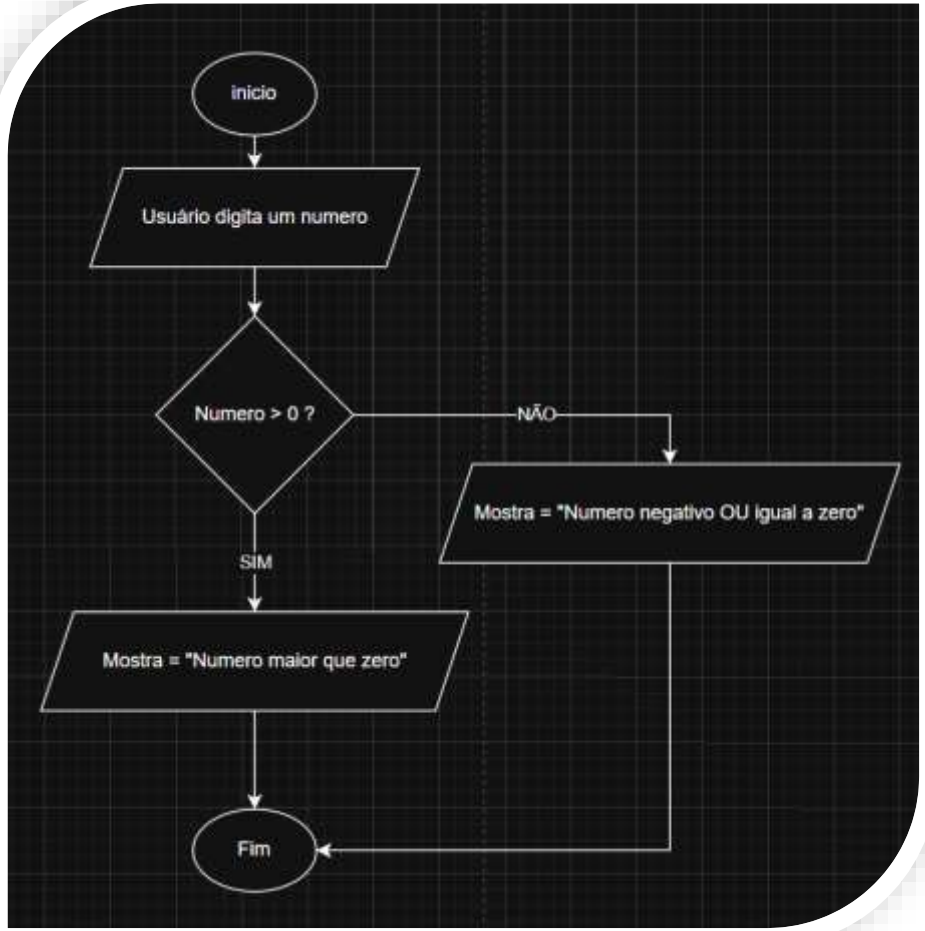
Principais Símbolos do Fluxograma

Símbolo de Decisão (Condicional - if/else)

Forma: Losango

Função: Representa perguntas de sim/não, verdadeiro/falso ou escolhas múltiplas.

Exemplo: Verificar se um número é positivo ou negativo.



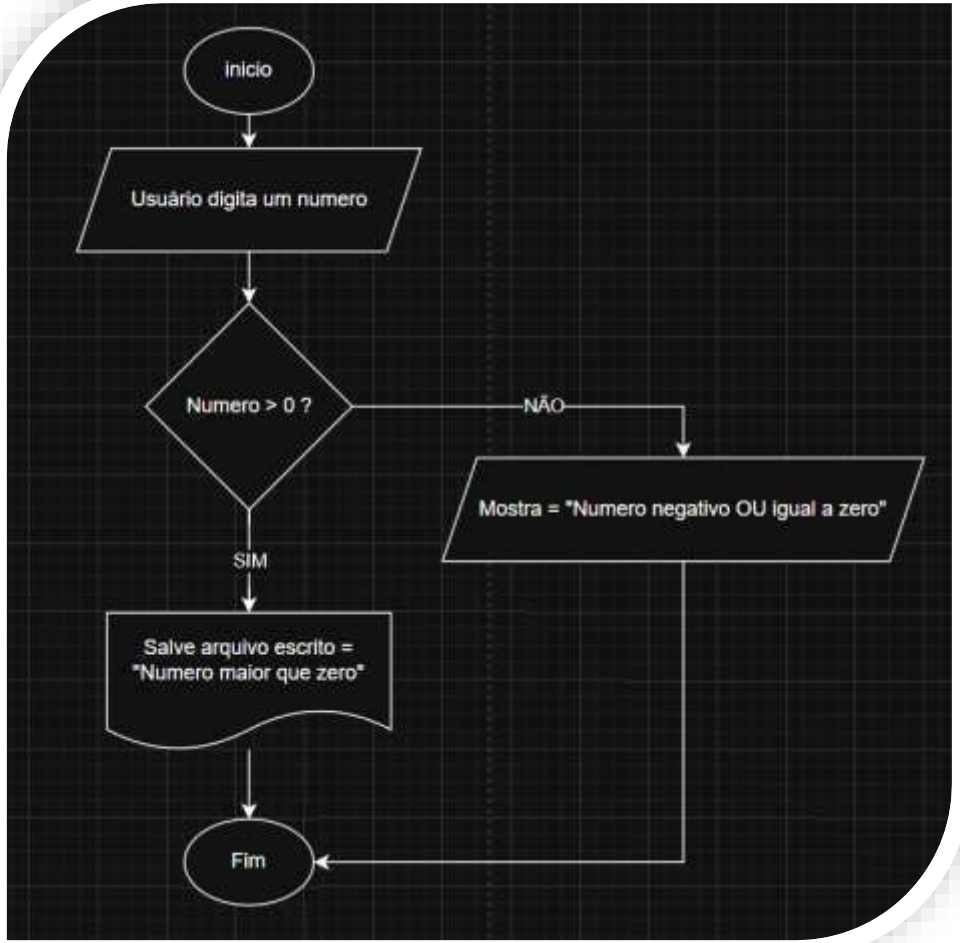
Principais Símbolos do Fluxograma

Símbolo de Saída (Documento)

Forma: Documento

Função: uma saída de dados ocorre na forma de um documento físico ou digital.

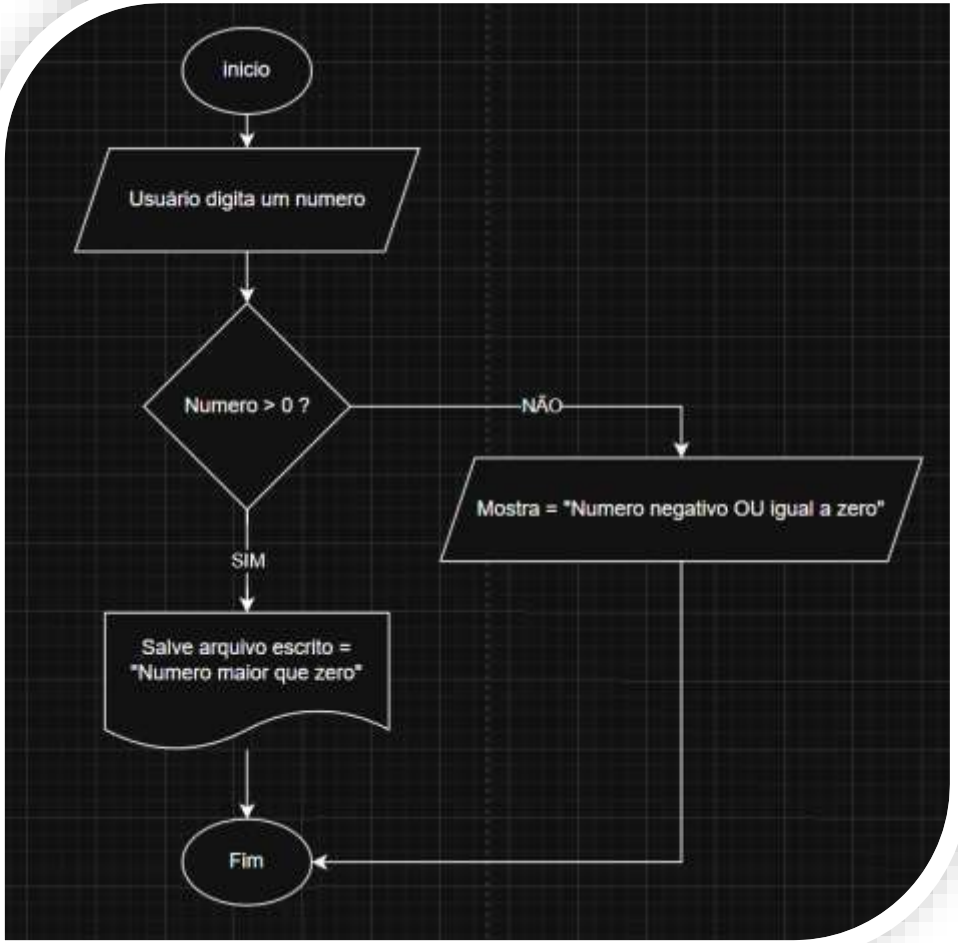
Exemplo: Gerar um relatório (extrato bancário). Criar um arquivo de texto ou PDF com informações processadas. Imprimir um recibo ou nota fiscal.



Principais Símbolos do Fluxograma

Agora é a sua vez.

No Draw.io, faça o fluxograma do seu algoritmo da atividade anterior.



Pense Algoritmicamente

1 - Todo algoritmo começa com uma definição clara do problema ou tarefa.
Observe a situação e descreva o problema de forma objetiva.
Com isso, você poderá planejar os passos para uma solução lógica.
Após identificar o problema, resuma-o em uma frase simples.
Por exemplo: "Preciso desenvolver um sistema para organizar melhor meus pertences".
Os problemas podem ser simples, como decidir o que comer.
Um exemplo: "Não consigo decidir o que pedir do menu", o que representa uma tarefa a ser resolvida.
Também pode haver apenas uma tarefa, como: "Preciso terminar de comprar comida em 30 minutos". Use o mesmo raciocínio lógico para cumprir essa tarefa.

Pense Algoritmicamente

2 - Um algoritmo depende de entradas corretas para funcionar.

Após identificar o problema, colete dados relevantes e planeje uma estratégia.

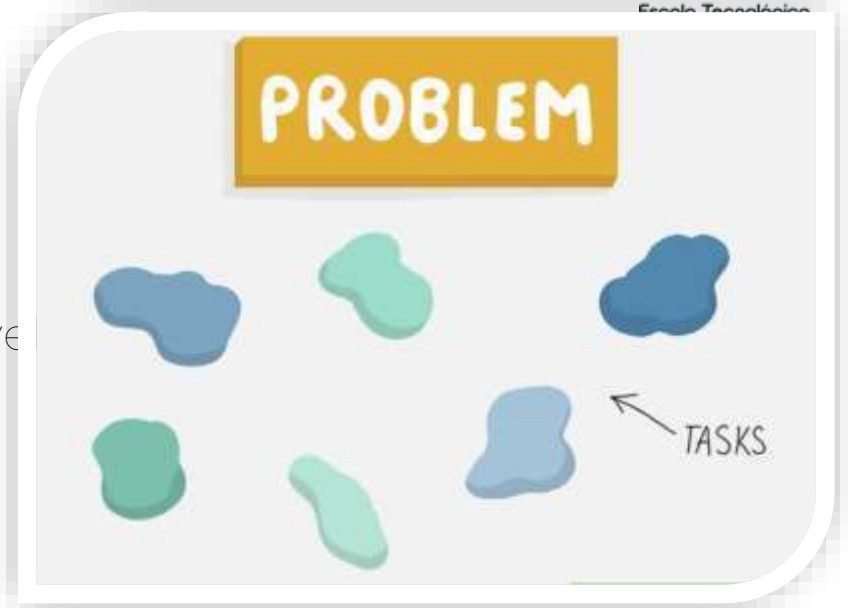
Por exemplo, "Meu carro está fazendo um barulho estranho" é um começo, mas vago.

Ao reunir mais detalhes, você pode refinar para: "Meu carro faz um barulho metálico na frente quando aplico os freios a mais de 30 km/h".

Essa lógica também se aplica a tarefas simples.

Se tiver 30 minutos para fazer compras, insira sua lista e o layout da loja, e use essas informações para planejar seu trajeto pelos corredores.

Pense Algoritmicamente



3 - Divida todas as tarefas em partes menores.
Torne cada tarefa o mais básica possível.
Isso torna o processo de solução de problemas muito mais gerenciáve
Não se preocupe em chegar à ordem correta dos eventos ainda.
Neste ponto, basta listar todas as coisas minuciosas que você precisa realizar para resolver o problema. [ver figura]
Se você quisesse limpar sua casa, por exemplo, pense em como você dividiria a tarefa. Você teria que aspirar, esfregar o chão do banheiro, pegar roupas sujas, tirar o lixo, lavar a louça, tirar o pó dos armários e lavar as janelas.
Essas tarefas não estão necessariamente em ordem, mas são tarefas gerenciáveis que você pode dividir ainda mais.

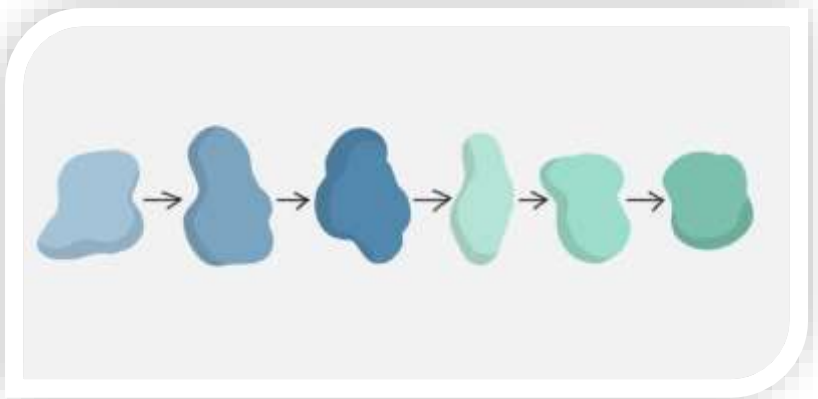
Pense Algoritmicamente

4 - Organize as etapas na ordem mais lógica.

Os algoritmos têm tudo a ver com a resolução de tarefas da maneira mais eficiente possível.

Depois de definir seu problema e dividir as tarefas necessárias, coloque-as na ordem mais lógica.

Pense em cada etapa que depende de uma anterior e ordene as tarefas em torno dessa relação.[ver imagem]



Continuando com o exemplo da limpeza doméstica, pense na ordem mais lógica para suas tarefas. Logicamente, você não pode aspirar o chão até pegar as roupas, então pegue as roupas primeiro. Da mesma forma, você não pode lavar as janelas se o chão estiver molhado de esfregar, então limpe as janelas antes de esfregar o chão.

Pense Algoritmicamente

5 - Antecipe variáveis usando a lógica "se-então".

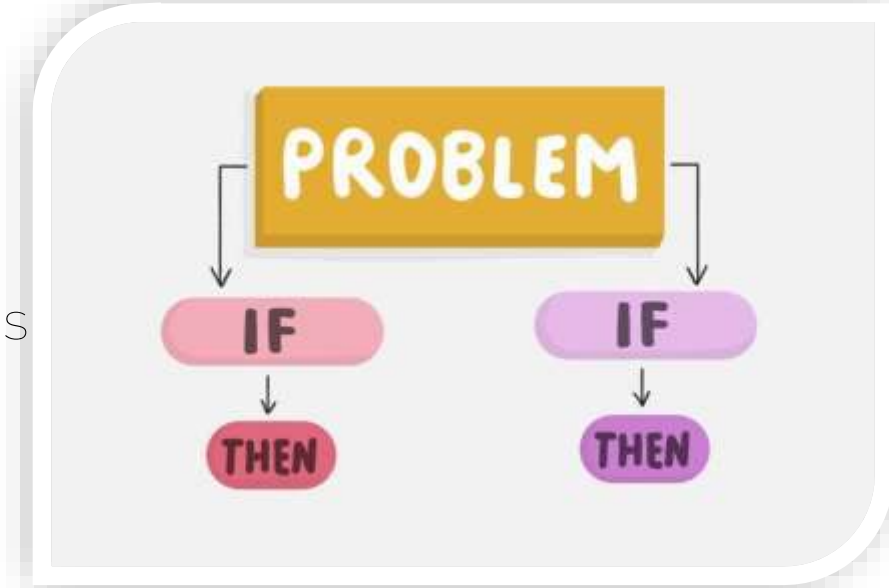
Muitos problemas não seguem uma única sequência e podem mudar conforme as entradas.

A abordagem "se-então" ajuda a lidar com essas variações sendo essencial nos algoritmos.

Por exemplo, se o problema for um ruído estranho no carro, pense: "Se for um guincho vindo dos pneus, verificarei os freios. Se for som metálico, verificarei o motor.“

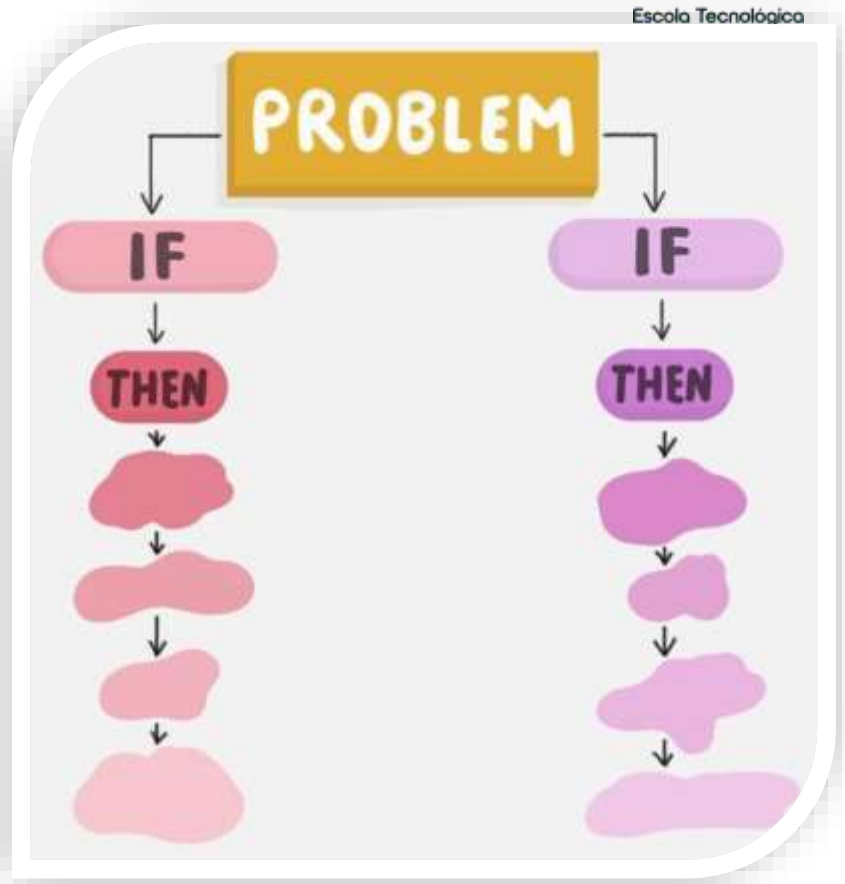
Essa lógica reflete o funcionamento de um algoritmo.

Quanto mais entradas você considerar, melhor poderá prever e lidar com as diferentes possibilidades.

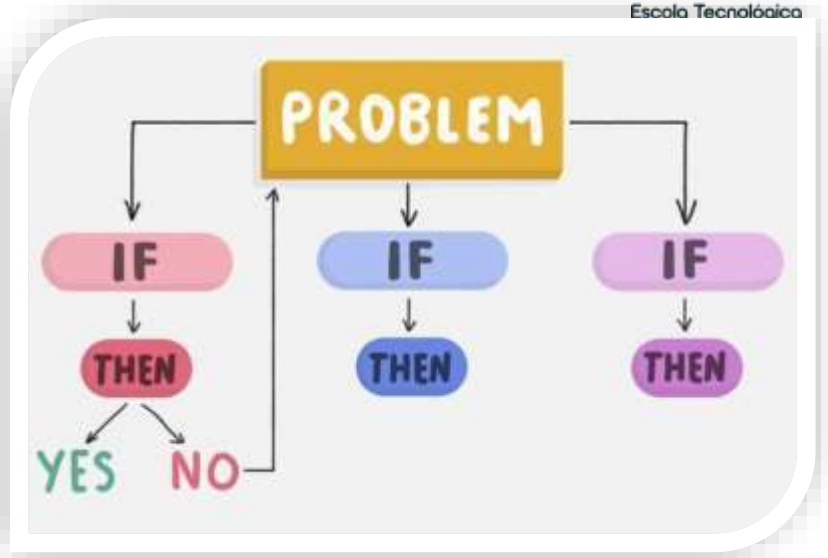


Pense Algoritmicamente

6 - Projete suas etapas com base nas variáveis identificadas.
Após dividir o problema e antecipar possíveis cenários, organize uma sequência de ações — como em um fluxograma — considerando o que fazer em cada situação.
No caso do barulho no carro, identifique a origem do som e, com base nisso, planeje como resolver.
Por exemplo: se o som vier dos pneus, verifique os freios. Se os freios estiverem bons, verifique os rolamentos. Se os rolamentos estiverem com defeito, substitua-os. Se não encontrar a origem, leve o carro ao mecânico.
Ao programar um algoritmo, as etapas e entradas precisam ser extremamente precisas.
Já na vida real, a mente humana permite certa flexibilidade nos passos, lidando com mais nuances.



Pense Algoritmicamente



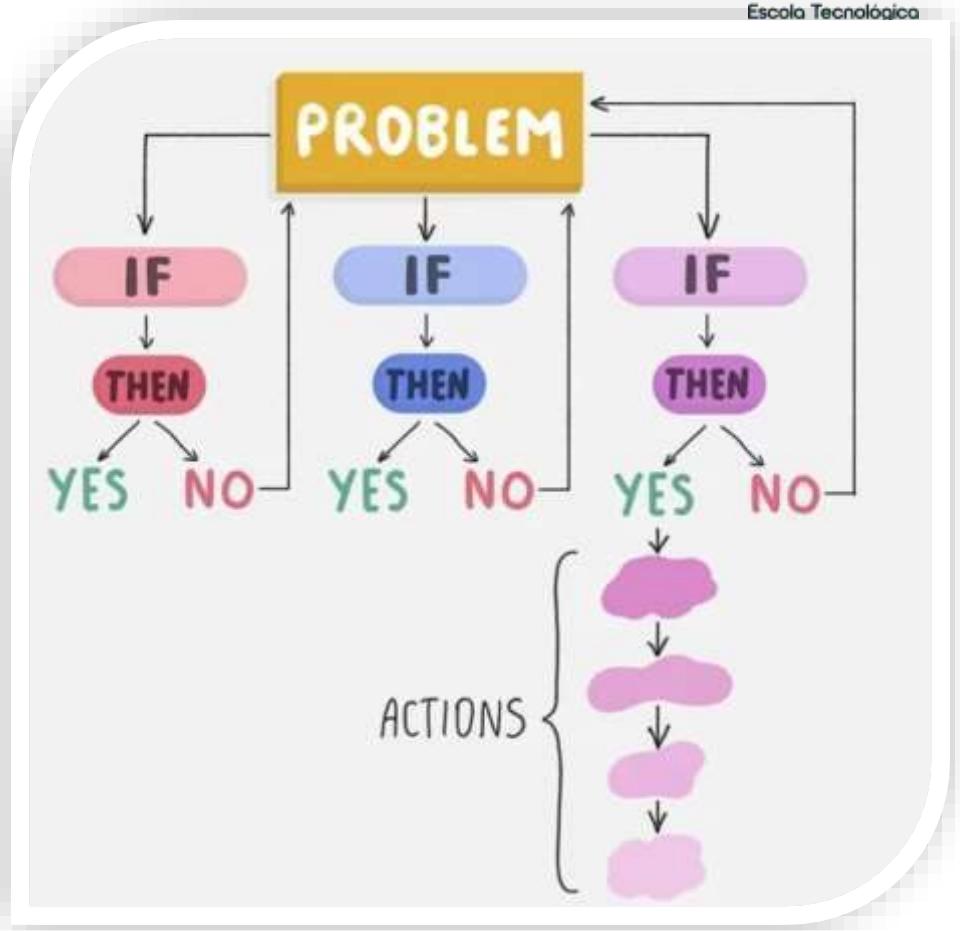
Planeje um loop caso não resolva o problema de primeira. Resolver problemas pode exigir tentativas e erros, então esteja preparado para recomeçar se a solução inicial falhar. Na programação, isso é chamado de loop — uma forma de retornar ao início e tentar novamente.

Loops são úteis porque evitam insistir em soluções ineficazes. Se algo não está funcionando, é melhor rever o caminho seguido e tentar outra abordagem.

Por exemplo, se não encontrar a origem do ruído no carro após verificar os freios e o motor, reinicie o processo. Dirija novamente, aplique os freios em diferentes velocidades e tente identificar novos padrões que ajudem a localizar o problema.

Pense Algoritmicamente

8 - Execute as ações conforme as variáveis surgirem.
Com o plano pronto, comece a seguir seu fluxograma e realize etapas específicas de acordo com o que encontrar.
Continue até identificar e resolver a causa do problema.
Por exemplo: há um barulho estranho no carro. Se for um guincho, verifico os pneus; se for uma pancada, verifico o motor.
O som é um guincho, então verifico os freios.
Vejo que a pastilha está gasta, instalo uma nova e o barulho desaparece — problema resolvido.
Esteja pronto para lidar com variáveis inesperadas.
Ao verificar os freios, talvez descubra um furo no pneu — um problema novo que exige outro conjunto de ações.
Ajuste sua abordagem conforme necessário.



FPFtech
Escola Tecnológica

Obrigado!

FPFtech
Escola Tecnológica

@fpftech.educacional