INFINITY SCHOOL

VISUAL ART CREATIVE CENTER

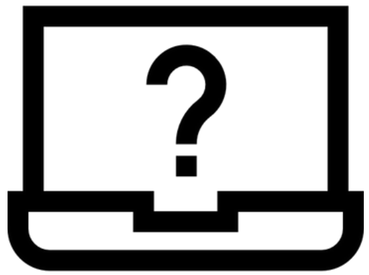
Aula 01 - Lógica da Programação

O que é Lógica da Programação?

A lógica da programação é basicamente o jeito que os programadores pensam e organizam as informações para criar programas de computador.

Aula 01 - Lógica da Programação

Para entender isso, imagine que você está jogando um jogo de adivinhação com seu amigo. Seu amigo pensa em um número e você precisa adivinhar qual é. Você pode começar perguntando: "O número é maior do que 5?" Se seu amigo disser "sim", você pode eliminar todos os números menores que 6 como possíveis respostas. Assim, você pode continuar fazendo perguntas para chegar à resposta certa.



Aula 01 - Lógica da Programação

Na programação, é a mesma ideia. Os programadores criam uma série de instruções que o computador segue para realizar uma tarefa. Por exemplo, se queremos criar um programa que imprima os números de 1 a 10 na tela, podemos usar a lógica da programação para escrever um código que instrua o computador a fazer isso.

Uma das coisas mais importantes na lógica da programação é a sequência de ações. Assim como você precisou fazer as perguntas na ordem certa para adivinhar o número do seu amigo, os programadores precisam organizar as instruções em uma ordem lógica para que o computador possa executá-las corretamente.

IN

Aula 01 - Algoritmo

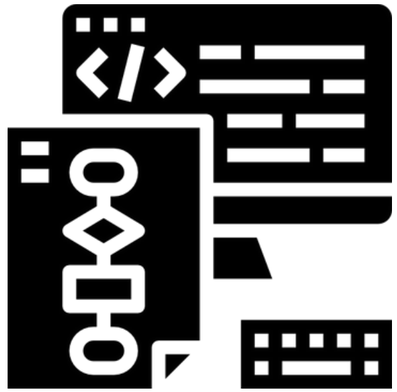
Um algoritmo é como uma receita de bolo. Assim como você segue uma receita para fazer um bolo delicioso, os computadores seguem um algoritmo para realizar uma tarefa.

Um algoritmo funciona de maneira similar. É uma lista de instruções que precisam ser seguidas para realizar uma tarefa específica. Essas instruções são escritas de forma lógica e sequencial para que o computador possa entendê-las e executá-las corretamente.

Aula 01 - Algoritmo

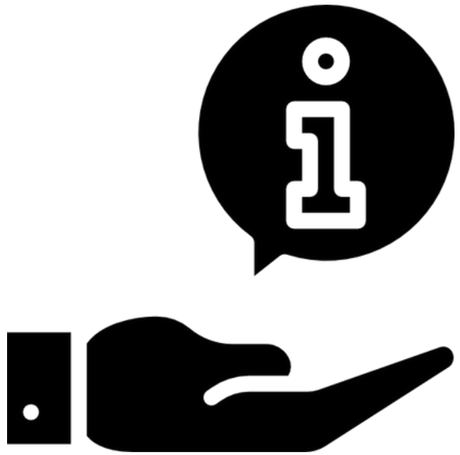
Por exemplo, um algoritmo para fazer uma conta matemática simples pode ser assim:

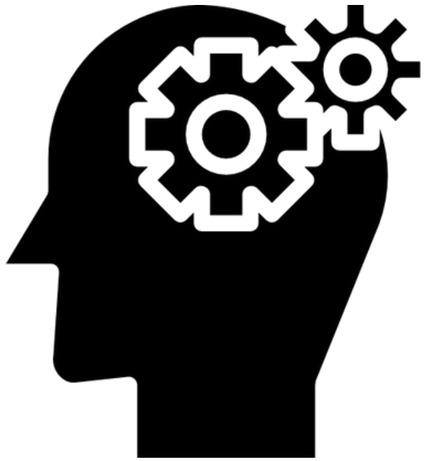
1. Pegue o primeiro número
2. Pegue o segundo número
3. Some os dois números
4. Mostre o resultado na tela

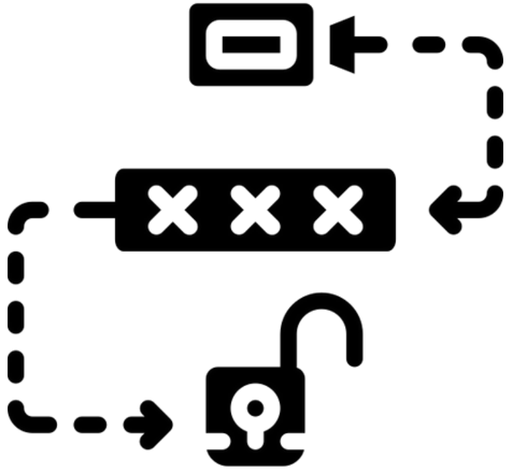


Aula 01 - Algoritmo

No exemplo anterior, era necessário "pegar" o primeiro e o segundo número. Em outras palavras, precisamos pedir certas informações ao usuário. Um algoritmo possui a entrada de dados, o processamento, e a saída dos dados.







Aula 01 - Entrada, processamento e saída de dados

- Entrada é quando o programa recebe informações do usuário ou do ambiente externo. Por exemplo, imagine que estamos criando um programa para calcular a média de duas notas. A entrada seria as duas notas que o usuário digita.

Processamento de dados é quando o programa manipula as informações recebidas para produzir um resultado desejado. No nosso exemplo, o processamento de dados seria somar as duas notas e dividir o resultado por dois para encontrar a média.

Aula 01 - Entrada, processamento e saída de dados

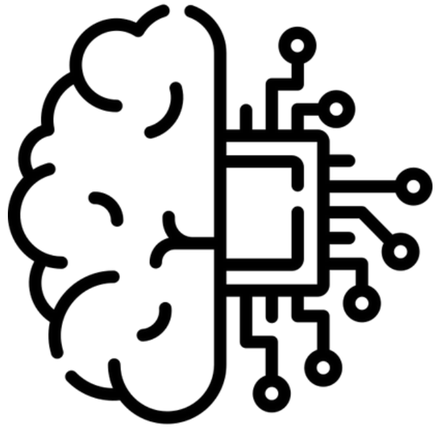
Saída é quando o programa exibe o resultado final. No nosso exemplo, a saída seria mostrar ao usuário a média das duas notas que ele digitou.

Então, em resumo, a entrada é a informação que o programa recebe, o processamento de dados é a manipulação dessas informações para produzir um resultado e a saída é a exibição do resultado final.

Aula 01 - Tipos de Algortimos

Existem alguns tipos de algoritmos. Os mais utilizados são:

- Descrição Narrativa
- Fluxograma
- Pseudocódigo



Aula 01 - Descrição Narrativa

- A descrição narrativa é uma forma de escrever em linguagem natural, como se fosse uma história, as instruções que um programa deve seguir para realizar uma tarefa. É como se fosse uma explicação detalhada, passo a passo, de tudo que precisa ser feito para alcançar o objetivo.

Por exemplo, se quisermos criar um programa que calcule a média de duas notas, podemos escrever uma descrição narrativa assim:

Aula 01 - Descrição Narrativa

1. O programa deve pedir ao usuário para digitar a primeira nota.
2. O programa deve armazenar a primeira nota em uma variável.
3. O programa deve pedir ao usuário para digitar a segunda nota.
4. O programa deve armazenar a segunda nota em outra variável.
5. O programa deve somar as duas notas armazenadas.
6. O programa deve dividir a soma por 2 para calcular a média.
7. O programa deve exibir a média na tela para o usuário.

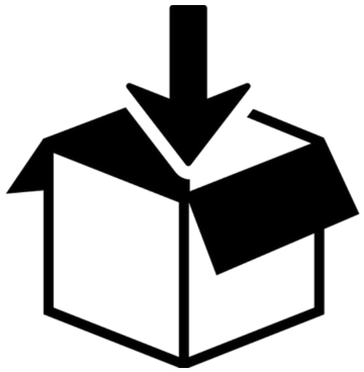
Mas o que seria uma variável?

IN

Aula 01 - Descrição Narrativa - Variável

- Variável é como uma caixinha que usamos para guardar valores em um programa. É um espaço na memória do computador que reservamos para armazenar informações temporariamente.

  Podemos pensar em uma variável como um armário com uma etiqueta, onde podemos colocar diferentes objetos em cada prateleira e depois acessá-los sempre que precisarmos.

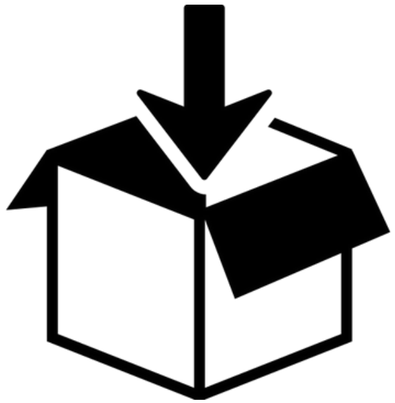


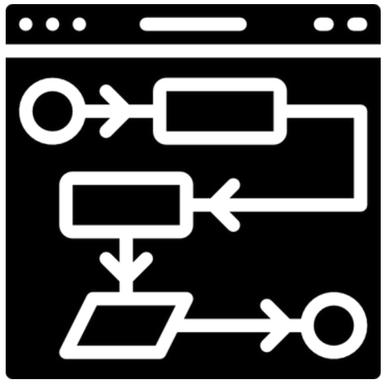
Aula 01 - Descrição Narrativa - Variável

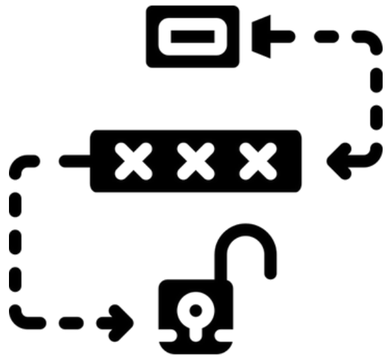
Depois, podemos fazer operações com essas variáveis, como somar ou dividir, para obter o resultado desejado. E o melhor de tudo é que podemos reutilizar as variáveis em diferentes partes do programa, sem precisar digitar as informações novamente.

Aula 01 - Fluxograma

Um fluxograma é uma representação visual de um algoritmo ou processo. É uma ferramenta muito útil para ilustrar como as instruções de um programa se relacionam e para facilitar a compreensão do fluxo de informações dentro dele.







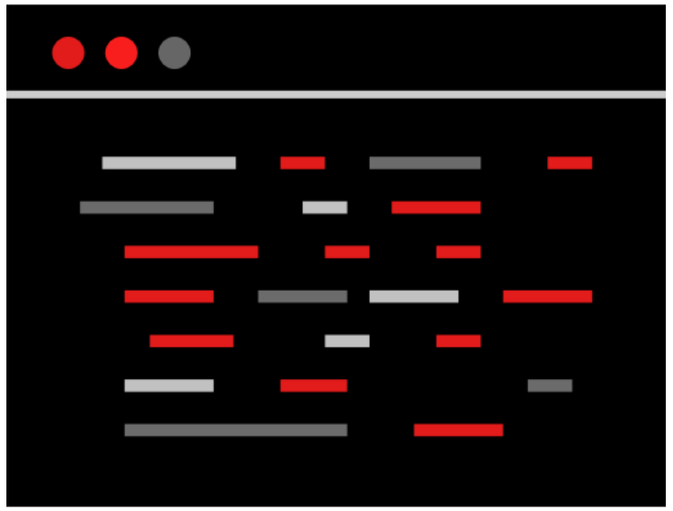
Aula 01 - Pseudocódigo

- Pseudocódigo é uma forma de escrever um algoritmo em linguagem natural, sem se preocupar com a sintaxe de uma linguagem de programação específica. É como se fosse uma descrição narrativa do algoritmo, mas com uma estrutura mais formal e organizada, que permite que outras pessoas entendam as instruções de forma clara e precisa.

Por exemplo, um pseudocódigo para calcular a média de duas notas pode ser assim:

Aula 01 - Pseudocódigo

```
Início
    Ler nota1
    Ler nota2
    soma = nota1 + nota2
    media = soma / 2
    Exibir "A média é: ", media
Fim
```



Aula 01 - Tipos de Variáveis

Em Portugol (uma linguagem de programação em português baseada em pseudocódigo), existem três tipos principais de variáveis:

Variáveis inteiras: armazenam valores numéricos inteiros, positivos ou negativos. Na linguagem Portugol, elas são definidas usando a palavra-chave "inteiro". Por exemplo:

```
inteiro idade = 25
```

Neste exemplo, a variável "idade" armazena o valor inteiro 25.

Aula 01 - Tipos de Variáveis

Variáveis reais: armazenam valores numéricos com casas decimais, positivos ou negativos. Na linguagem Portugol, elas são definidas usando a palavra-chave "real". Por exemplo:

```
real altura = 1.75
```

Neste exemplo, a variável "altura" armazena o valor real 1.75.

Aula 01 - Tipos de Variáveis

Variáveis caracteres: armazenam caracteres individuais, como letras, números ou símbolos. Na linguagem Portugol, elas são definidas usando a palavra-chave "caractere". Por exemplo:

```
caractere inicial = "J"
```

Neste exemplo, a variável "inicial" armazena o caractere "J".

Além desses tipos principais, existem outros tipos de variáveis em Portugol, como os tipos lógicos (para armazenar valores verdadeiros ou falsos)

Aula 01 - Operadores Aritméticos

Os operadores aritméticos são fundamentais para a programação, pois são responsáveis por realizar cálculos matemáticos em um programa. Aqui estão os principais operadores aritméticos e sua função:

- Adição (+): este operador é usado para somar dois valores.
- Subtração (-): este operador é usado para subtrair um valor de outro.
- Multiplicação (*): este operador é usado para multiplicar dois valores.
- Divisão (/): este operador é usado para dividir um valor por outro.
- Módulo (%): este operador é usado para obter o resto da divisão entre dois valores.

Aula 01 - Operadores Aritméticos

Módulo (%): O operador de módulo é representado pelo símbolo %. Ele retorna o resto da divisão de um número pelo outro. Por exemplo, 7 % 3 retorna 1, pois 7 dividido por 3 é 2 com um resto de 1.

Resto da divisão (//): O operador de divisão de piso, representado pelo símbolo //. Ele retorna o resultado da divisão, mas arredonda o resultado para o menor inteiro. Por exemplo, 7 // 3 retorna 2, pois 7 dividido por 3 é 2 com um resto de 1 e a divisão de piso arredonda para baixo para 2.

Exponenciação (**): O operador de exponenciação é representado pelo símbolo **. Ele calcula a potência de um número elevado a outro. Por exemplo, 2 ** 3 retorna 8, pois 2 elevado à terceira potência é igual a 8

IN

Aula 01 - Operadores Relacionais

Os operadores relacionais são usados para comparar dois valores e retornar um valor lógico (verdadeiro ou falso) com base na relação entre eles. Aqui estão os operadores relacionais mais comuns e uma explicação simples de cada um:

- Igual a (==): este operador verifica se dois valores são iguais. Por exemplo, 2 == 2 é verdadeiro, mas 2 == 3 é falso.
- Diferente de (!=): este operador verifica se dois valores são diferentes. Por exemplo, 2 != 3 é verdadeiro, mas 2 != 2 é falso.
- Maior que (>): este operador verifica se um valor é maior que outro. Por exemplo, 3 > 2 é verdadeiro, mas 2 > 3 é falso.

Aula 01 - Operadores Relacionais

- Maior ou igual a (>=): este operador verifica se um valor é maior ou igual a outro. Por exemplo, 3 >= 2 é verdadeiro, mas 2 >= 3 é falso.
- Menor que (<): este operador verifica se um valor é menor que outro. Por exemplo, 2 < 3 é verdadeiro, mas 3 < 2 é falso.
- Menor ou igual a (<=): este operador verifica se um valor é menor ou igual a outro. Por exemplo, 2 <= 3 é verdadeiro, mas 3 <= 2 é falso.

Aula 01 - Portugol

- Portugol é uma linguagem de programação estruturada, que tem como objetivo ajudar a ensinar os conceitos básicos de programação de forma simples e didática.

Comando **leia**: permite que o programa leia um valor digitado pelo usuário e armazene em uma variável. Por exemplo:

*leia(*nome)

Neste exemplo, o programa irá ler o valor digitado pelo usuário e armazená-lo na variável **nome**.

Aula 01 - Portugol

Comando **escreva**: permite que o programa escreva um valor na tela.
Por exemplo:

escreva("Olá, mundo!")

Neste exemplo, o programa irá escrever "Olá, mundo!" na tela.

Comando **se**: é uma estrutura condicional que permite que o programa execute uma determinada ação somente se uma condição for verdadeira.
Por exemplo:

Aula 01 - Portugol

```
programa {
    funcao inicio() {
    inteiro idade
      idade = 19
  se (idade >= 18) entao
    escreva("Você é maior de idade!")
  fimse
 }
}
```

Neste exemplo, o programa irá verificar se a variável **idade** é maior ou igual a 18. Se essa condição for verdadeira, o programa irá escrever "Você é maior de idade!" na tela.

Comando **para**: é uma estrutura de repetição que permite que o programa execute um bloco de código várias vezes. Por exemplo:

Aula 01 - Portugol

```
para i de 1 ate 10 faca
    escreva(i)
fimpara
```

Neste exemplo, o programa irá escrever os números de 1 a 10 na tela.

Comando **enquanto**: é outra estrutura de repetição que permite que o programa execute um bloco de código enquanto uma condição for verdadeira. Por exemplo:

Aula 01 - Portugol

```
enquanto (idade < 18) faca
    escreva("Você ainda não é maior de idade.")
    leia(idade)
fimenquanto
```

Neste exemplo, o programa irá solicitar que o usuário digite a idade até que a condição **idade < 18** seja falsa.

Esses são alguns dos principais comandos da linguagem Portugol.

Aula 01 -Condicionais e Repetições

Condicionais são estruturas de controle de fluxo em programação que permitem que o programa tome decisões com base em uma condição. As repetições são realizadas por meio de estruturas de controle de fluxo que permitem que o programa execute um bloco de código várias vezes. As duas principais estruturas de repetição em Portugol são o para e o enquanto.

se (condição) *então*
    // Código a ser executado se a condição for verdadeira
fimse

Aqui está um exemplo simples para ajudar a entender melhor:

Aula 01 - Condicionais e Repetições

```
programa {
    funcao inicio() {
        inteiro idade
    escreva("Digite sua idade: ")
    leia(idade)

    se (idade >= 18) entao
      escreva("Você é maior de idade.")
    fimse

    escreva("Fim do programa.")
 }
}
```

Neste exemplo, o programa pede que o usuário digite sua idade e armazena o valor na variável idade. Em seguida, a condição idade >= 18 é verificada. Se essa condição for verdadeira (ou seja, se a idade for maior ou igual a 18), o programa escreverá "Você é maior de idade". Caso contrário, nada será impresso na tela. Em seguida, o programa escreve "Fim do programa" para indicar que terminou a execução.

IN

Parabéns!

Você concluiu a Aula 01 de Lógica da Programação.
Continue estudando e até a próxima!

INFINITY SCHOOL

VISUAL ART CREATIVE CENTER