


ALGORITMOS E LÓGICA DE PROGRAMAÇÃO



- **Algoritmo**
- **Lógica de Programação**
 - Fases na concepção de um algoritmo.
 - Formas de Representação de Algoritmos
 - Descrição Narrativa;
 - Fluxograma Convencional;
 - Pseudocódigo.
- **Tipos de Dados**
- **Constantes e Variáveis**
- **Desenvolvendo os Programas**
 - Programação (Python)
 - Tipos de Dados
 - Operadores Matemáticos
- **Diagrama de Bloco (Fluxograma)**
- **Operadores**
 - Operadores Aritméticos
 - Expressões Aritméticas
 - Operadores Relacionais
 - Operadores Lógicos

- **Estrutura de Decisão e Repetição**
 - SE ENTÃO SENÃO / IF ... THEN ... ELSE
 - Programação (Python)
 - Expressões Aritméticas - MOD
 - CASO SELECIONE / SELECT... CASE
 - Programação (Python)
- **Comandos de Repetição ou Laços**
 - Laço Para (For)
 - Programação (Python)
 - Laço Enquanto (while)
 - Programação (Python)
 - Laço Repita...Até que (repeat ... Until)
 - Programação (Python)

- 
- **Ambientes de programação**
 - **Dados estruturados: vetores, matrizes e registros**
 - **Introdução a uma linguagem de programação estruturada**

Algoritmo

Um algoritmo é formado de uma sequência de passos para a solução de um problema e os passos que levam a uma solução são muitos.

Por exemplo, a mesma tarefa pode ser resolvida de várias maneiras diferentes e com passos diferentes.

Mas será que esta afirmação está correta?

Um exemplo que podemos citar de algoritmos são os das operações básicas (adição, subtração, multiplicação e divisão) de números reais decimais.

Exemplo:

Somar dois números.


- Escreva o primeiro numero no retângulo **A**.
- Escreva o segundo numero no retângulo **B**.
- Some o numero do retângulo **A** com o numero do retângulo **B** e coloque o resultado no Retângulo **C**.

Retângulo **A**

Retângulo **B**

Retângulo **C**





“Os algoritmos, servem para representar a solução de qualquer problema, mas no caso do **Processamento de Dados**, eles devem seguir as regras básicas de programação para que sejam compatíveis com as linguagens de programação.”

Lógica de Programação

Por existir varias formas de resolver um problema é ai que entra a lógica de programação.

A Lógica de programação é a técnica de encadear pensamentos para atingir determinado objetivo.

Na **Lógica de programação** é onde você usa a:

- Coerência
- Racionalidade
- Por que fazer de uma forma e não de outra?
- Arte de bem pensar
- Ordem no pensamento

Algoritmo é aquilo que você trouxe para solucionar, e a lógica é como você fez para solucionar aquele algoritmo.

Fases na concepção de um algoritmo:

- **Entradas:** Dados que alimentam o sistema, informações inseridas pelo usuário;
Exemplo: A nota de um aluno é um dado de entrada, o nome de um funcionário, quantas pessoas tem mais de 18 anos, etc.
- **Processamento:** Cálculos e manipulação dos dados de entrada;
Exemplo: O cálculo da média de um aluno:
A entrada seria duas notas: 8, 10.
A soma dividida pelo numero de notas: $(8+10)/2 = \text{Processamento}$.
- **Saída:** Resultado obtido do processamento dos dados;
Exemplo: Seria o resultado do processamento = $(8+10)/2 = 9$
- **Teste de mesa:** É a execução das três fases acima mencionada, verificando o funcionamento na prática das instruções executadas.
Exemplo: Testar a veracidade do algoritmo para ver se está funcionando.

Formas de Representação de Algoritmos

Dentre as formas de representação de algoritmos mais conhecidas podemos citar:

- **Descrição Narrativa;**
- **Fluxograma Convencional;**
- **Pseudocódigo**, também conhecido como Linguagem Estruturada ou Portugol.

Descrição Narrativa

Nesta forma de representação os algoritmos são expressos diretamente em linguagem natural.

- **Seqüência Lógica**

Estes pensamentos, podem ser descritos como uma seqüência de instruções, que devem ser seguidas para se cumprir uma determinada tarefa.

Seqüência lógica são passos executados até atingir um objetivo ou solução de um problema.

- **Instruções**

Instruções são um conjunto de regras ou normas definidas para a realização ou emprego de algo. Em informática, é o que indica a um computador uma ação elementar a executar e na linguagem natural são um conjunto de regras para se fazer algo.

Essas instruções tem que ser executadas em conjunto e numa ordem adequada.

Exemplo:

Instruções de uma Receita de bolo:

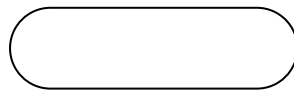
- Misture os ingredientes
- Unte a forma com manteiga
- Despeje a mistura na forma
- Se houver coco ralado
- Então despeje sobre a mistura
- Leve a forma ao forno
- Enquanto não corar
- Deixe a forma no forno
- Retire do forno
- Deixe esfriar

Fluxograma Convencional

- É uma representação gráfica de algoritmos onde formas geométricas diferentes implicam ações (instruções, comandos) distintos. Tal propriedade facilita o entendimento das idéias contidas nos algoritmos e justifica sua popularidade.



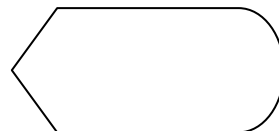
Início e final do fluxograma



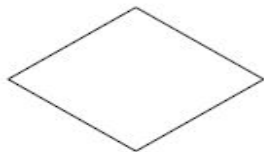
Operação de entrada de dados



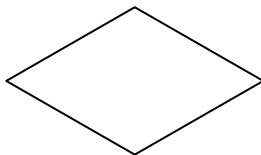
Operação de saída de dados



Operação de atribuição




Decisão




- 
1. **Faça um algoritmo que receba 2 notas e calcule a média aritmética**

**Usando descrição narrativa e
Fluxograma**




2. Crie um algoritmo que leia um número diferente de zero e diga se este número é positivo ou negativo

Usando descrição narrativa e Fluxograma



3. Crie um algoritmo que leia a idade de uma pessoa e diga em qual ano ela nasceu

**Usando descrição narrativa e
Fluxograma**



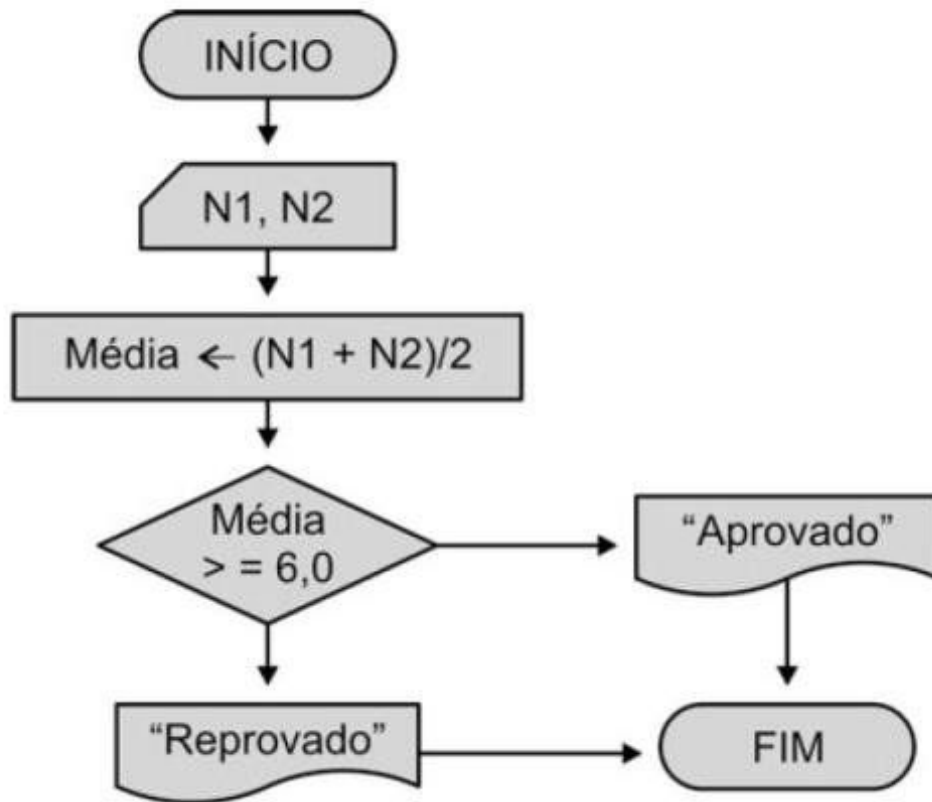
4. Crie um algoritmo que receba 3 números e informe qual o maior entre eles.

**Usando descrição narrativa e
Fluxograma**



5. Crie um algoritmo que leia um número e diga se ele é par ou ímpar

**Usando descrição narrativa e
Fluxograma**



Pseudocódigo

Esta forma de representação de algoritmos é rica em detalhes, como a definição dos tipos das variáveis usadas no algoritmo. Por assemelhar-se bastante à forma em que os Programas são escritos, encontra muita aceitação. Na verdade, esta representação é suficientemente geral para permitir a tradução de um Algoritmo nela representado para uma linguagem de programação específica seja praticamente direta.

A forma geral da representação de um algoritmo na forma de pseudocódigo é a seguinte:

-
- **Algoritmo** <nome_do_algoritmo>
- <declaração_de_variáveis>
- <subalgoritmos>
- **Início**
- <corpo do algoritmo>
- **Fim**



Algoritmo é uma palavra que indica o início da definição de um algoritmo em forma de pseudocódigo.

<nome_do_algoritmo> é um nome simbólico dado ao algoritmo com a finalidade de distingui-los dos demais.

<declaração_de_variáveis> consiste em uma porção opcional onde são declaradas as variáveis globais usadas no algoritmo principal e, eventualmente, nos sub-algoritmos.

<subalgoritmos> consiste de uma porção opcional do pseudocódigo onde são definidos os sub-algoritmos.

Início e Fim são respectivamente as palavras que delimitam o início e o término do conjunto de instruções do corpo do algoritmo.

Tipos de Dados

Os **dados** propriamente ditos, que correspondem à porção das informações a serem processadas pelo computador.

● **Dados Numéricos Inteiros**

Os números **inteiros** são aqueles que não possuem componentes decimais ou fracionários, podendo ser positivos ou negativos.

Exemplos de números **inteiros** temos:

- 24 - número inteiro positivo
- 0 - número inteiro
- -12 - número inteiro negativo

● **Dados Numéricos Reais**

Os dados de tipo **real** são aqueles que podem possuir componentes decimais ou fracionários, e podem também ser positivos ou negativos.

Exemplos de dados do tipo **real**:

- 24.01 - número real positivo com duas casas decimais
- 144. - número real positivo com zero casas decimais
- -13.3 - número real negativo com uma casa decimal
- 0.0 - número real com uma casa decimal
- 0. - número real com zero casas decimais

● Dados Literais

O tipo de dados **literal** é constituído por uma sequência de caracteres contendo letras, dígitos e/ou símbolos especiais. Este tipo de dados é também muitas vezes chamado de **alfanumérico**, **cadeia** (ou **cordão**) de **caracteres**, ainda, do inglês, **string**.

Usualmente, os dados literais são representados nos algoritmos pela coleção de caracteres, delimitada em seu início e término com o caractere aspas (").

Exemplos de dados do tipo literal:

- "QUAL ?" - literal de comprimento 6
- " " - literal de comprimento 1
- "qUaL ?!\$" - literal de comprimento 8
- " AbCdefGHi" - literal de comprimento 9
- "1-2+3=" - literal de comprimento 6
- "0" - literal de comprimento 1

● Dados Lógicos

A existência deste tipo de dado é, de certo modo, um reflexo da maneira como os computadores funcionam. Muitas vezes, estes tipos de dados são chamados de **booleanos**, devido à significativa contribuição de BOOLE à área da lógica matemática.

O tipo de dados **lógico** é usado para representar dois únicos valores lógicos possíveis: **verdadeiro** e **falso**. É comum encontrar-se em outras referências outros tipos de pares de valores lógicos como **sim/não**, **1/0**, **true/false**.

Nos algoritmos apresentados nesta apostila os valores lógicos serão delimitados pelo caractere ponto (.).

Exemplo:

-
- .V. - valor lógico verdadeiro
- .F. - valor lógico falso