

# ALGORITMO E LÓGICA DE PROGRAMAÇÃO

- Material de apoio

- **Algoritmo:**

Um algoritmo é formalmente uma sequência finita de passos que levam a execução de uma tarefa. Podemos pensar em algoritmo como uma receita, uma sequência de instruções que dão cabo de uma meta específica. Estas tarefas não podem ser redundantes nem subjetivas na sua definição, devem ser claras e precisas.

**Exemplo:** “Chupar uma bala”.

1. Pegar a bala.
2. Retirar o papel.
3. Chupar a bala.
4. Jogar o papel no lixo.

- **Linguagem de programação:**

Um método padronizado onde expressa instruções para um computador.

Uma linguagem de programação pode ser de nível alto ou baixo. As chamadas de **alto nível** possuem uma construção linguística que se aproxima mais da forma humana de se comunicar. Por exemplo, é comum ver termos em inglês como “delete” que oferecem instruções para que o programa apague alguma informação. Isso torna esse modelo um pouco mais intuitivo e fácil de aprender.

Já a linguagem de **baixo nível**, utiliza comandos mais complexos e o código binário, que são sequências combinadas dos números 0 e 1. Isso otimiza a comunicação com a máquina, aumentando a agilidade do processo. Os diferentes tipos funcionam melhor de acordo com o objetivo que se busca atingir.

- **Variável:**

Uma variável é um espaço na memória do computador destinado a um dado que é alterado durante a execução do algoritmo. Para funcionar corretamente, as variáveis precisam ser definidas por nomes e tipos.

int	Número inteiro.
double	Número com casas decimais.
boolean	Apenas pode valer Verdadeiro ou Falso.
char	Um caracter, que pode ser letra ou número ou sinal gráfico.
String	Uma série de “chars”, formando uma frase ou texto.

Figura 1: Tipos de variáveis

Operador	Conceito	Exemplo
+ (Adição ou sinal positivo)	- Realiza a soma entre operandos - Adiciona o sinal de positivo ao número	- 10 + 7 - +4
- (Subtração ou sinal negativo)	- Realiza a subtração entre operandos - Adiciona o sinal de negativo ao número	- 10 - 7 - -4
* (Multiplicação)	Realiza a multiplicação entre operandos	3 * 4
/ (Divisão)	Realiza a divisão entre operandos	10 / 5
// (Divisão inteira)	Realiza a divisão entre operandos e a parte decimal do resultado	10 // 6
% (Módulo)	Retorna o resto da divisão entre operandos	4 % 2
** (Exponenciação)	Retorna um número elevado a potência de outro	4 ** 2

Figura 2: Operadores aritméticos

Operador	Exemplo	Equivalente a
=	x = 1	x = 1
+=	x += 1	x = x + 1
-=	x -= 1	x = x - 1
*=	x *= 1	x = x * 1
/=	x /= 1	x = x / 1
%=	x %= 1	x = x % 1

Figura 3: Operadores de atribuição

Operador	Conceito	Exemplo
>(Maior que)	Verifica se um valor é maior que outro	x > 5
<(Menor que)	Verifica se um valor é menor que outro	x < 5
== (Igual a)	Verifica se um valor é igual a outro	x == 5
!= (Diferente de)	Verifica se um valor é diferente de outro	x != 5
>= (Maior ou igual a)	Verifica se um valor é maior ou igual a outro	x >= 5
<= (Menor ou igual a)	Verifica se um valor é menor ou igual a outro	x <= 5

Figura 4: Operadores de comparação

Operador	Conceito	Exemplo
and	Retorna True se todas as condições forem verdadeiras, caso contrário retorna False	x > 1 and x < 5
or	Retorna True se uma das condições for verdadeiras, caso contrário retorna False	x > 1 or x < 5
not	Inverte o resultado: se o resultado da expressão for True, o operador retorna false	not(x > 1 and x < 5)

Figura 5: Operadores lógicos

Operador	Conceito	Exemplo
is	Retorna True se as variáveis comparadas forem o mesmo objeto	nome is 'Marcos'
is not	Retorna True se as variáveis comparadas não forem o mesmo objeto	x is not 'Python'

Figura 6: Operadores de identidade

Operador	Conceito	Exemplo
in	Retorna True caso o valor seja encontrado na sequência	2 in x
not in	Retorna True caso o valor não seja encontrado na sequência	2 not in x

Figura 7: Operadores de associação

- **Pseudocódigo:**

É uma forma de representar um algoritmo de maneira informal e compreensível, usando uma mistura de linguagem humana e elementos de programação. Ele não é uma linguagem de programação real, mas sim uma descrição de alto nível de como um algoritmo deve funcionar. O objetivo principal do pseudocódigo é expressar a lógica e a sequência de passos de um algoritmo de forma clara antes de implementá-lo em uma linguagem de programação específica.

**Exemplo:** Ler dois números e realizar a soma entre eles.

Início

Ler numero1

Ler numero2





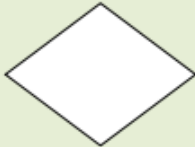




Soma = numero1 + numero2

Exibir "A soma é: ", Soma

Fim

- **Fluxograma:**

É uma representação gráfica de um processo, algoritmo ou sistema por meio de símbolos e linhas que conectam esses símbolos para indicar a sequência lógica das etapas. Essa representação visual ajuda a compreender de forma clara e intuitiva como um processo funciona, sem depender de uma linguagem de programação específica ou detalhes técnicos.

Símbolo	Descrição
	<b>Terminal:</b> Representa o início e fim do fluxograma.
	<b>Processamento:</b> Representa a execução de operações ou ações como cálculos, atribuições de valores das variáveis, dentre outras.
	<b>Entrada de Dados:</b> Representa a entrada de dados para as variáveis através do teclado.
	<b>Saída de vídeo:</b> Através deste símbolo podemos representar a saída de informações (dados ou mensagens) por meio de um dispositivo visual de saída de dados, o monitor de vídeo e outros.
	<b>Decisão:</b> Representa uma ação lógica, que realizará uma sequência de instruções sendo verdadeiro ou falso, se o teste lógico apresentar o resultado "verdadeiro", realizará uma sequência se o teste lógico apresentar resultado "false" será executado outra sequência. <i><b>Na aula 10 em estruturas de decisão veremos mais afundo os teste lógicos com implementações em uma linguagem de programação, pseudocódigo e fluxograma.</b></i>
	<b>Preparação:</b> Representa uma ação de preparação para o processamento, um processamento predefinido, este tipo de representação será bastante utilizado na aula 14 onde trataremos de procedimentos e funções.
	<b>Conector:</b> Este símbolo é utilizado para interligar partes do fluxograma ou desviar o fluxo para um determinado trecho do fluxograma.
	<b>Conector de Página:</b> Utilizado para ligar partes do fluxograma em páginas distintas.
	<b>Seta:</b> Orienta a sequência de execução ou leitura, que poderá ser horizontal ou vertical.

- **Sugestão de programas para fluxogramas e pseudocódigo:**

- **Portugol IDE:** O Portugol IDE é um ambiente de desenvolvimento de algoritmos, mais especificamente, trata-se de um simulador de linguagem algorítmica que visa o desenvolvimento do raciocínio lógico. Usando um ambiente simples e com ferramentas visuais, o Portugol destaca o desenvolvimento dos algoritmos ao invés do desenvolvimento de

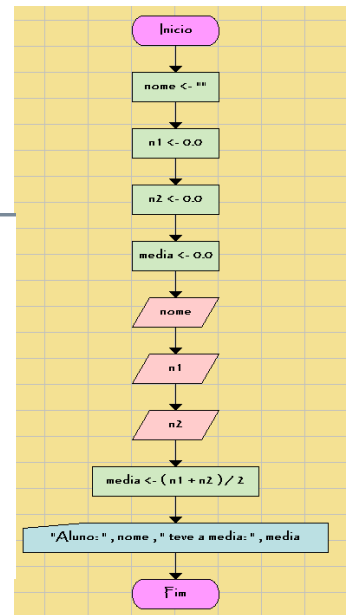
programas. Para isto ele utiliza uma linguagem algorítmica, uma pseudo-linguagem de programação que utiliza o português como base das suas instruções. Seu editor tem ferramentas que complementam e corrigem algumas falhas que o aprendiz comete. Já o editor de fluxogramas é uma abordagem gráfica da programação que permite o desenvolvimento de algoritmos de forma visual e com ferramentas que fazem a tradução para linguagem algorítmica.

Obs: Precisa do java em sua máquina para executar o programa com total sucesso.

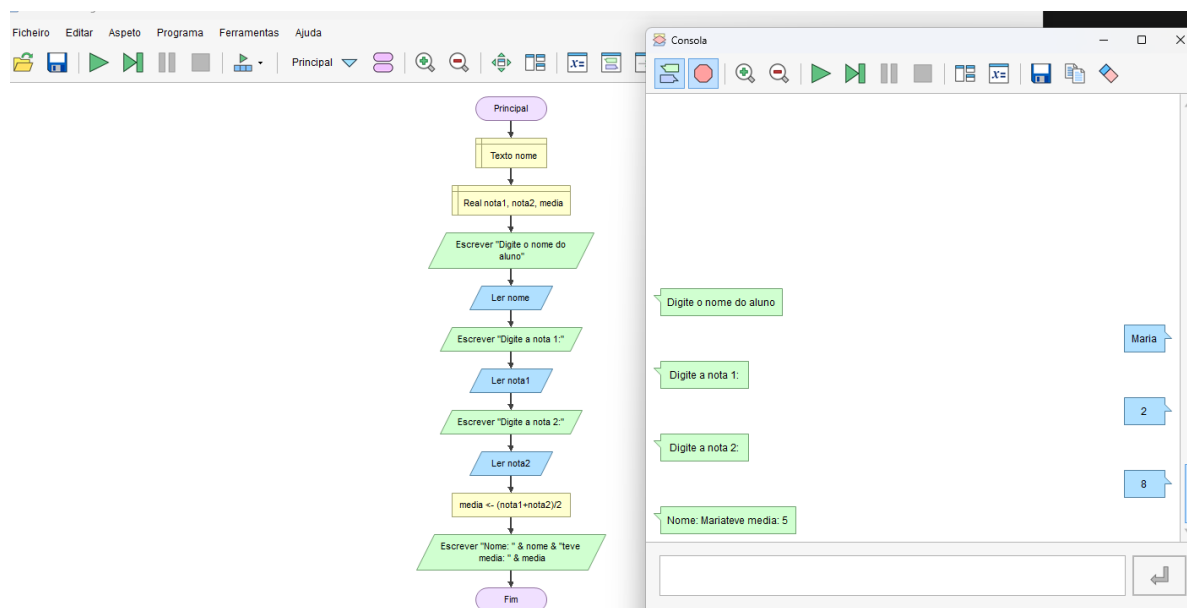
```

inicio
    texto nome
    real n1, n2, media
    ler nome
    ler n1
    ler n2
    media <- ( n1 + n2 ) / 2
    escrever "Aluno: " , nome , " teve a media: " , media
fim

```



- **Flowgorithm:** O Flowgorithm é um programa de computador que permite desenvolver e simular fluxogramas de forma rápida e fácil.



- **VisualG:** É uma ferramenta na qual pode-se simular pseudocódigos, podemos dizer que é a interpretação de uma linguagem algorítmica, utilizando comandos e instruções em português

para representar as ações dos algoritmos, também conhecida como Portugol ou Português Estruturado.

The screenshot shows a Portugol IDE interface. The main window is titled "Área dos algoritmos ( Edição do código fonte ) -> Nome do arquivo: [01-media.ALG]". It contains a code editor with the following code:

```
1 Algoritmo "semnome"
2 // Disciplina : [Linguagem e Lógica de Programação]
3 // Professor : Antonio Carlos Nicolodi
4 // Descrição : Aqui você descreve o que o programa faz! (função)
5 // Autor(a) : Nome do(a) aluno(a)
6 // Data atual : 25/08/2023
7 Var
8 // Seção de Declarações das variáveis
9 numeroA, numeroB: inteiro
10 media: real
11
12 Inicio
13 // Seção de Comandos, procedimento, funções, operadores, etc...
14 escreva("---- Média-----")
15 leia(numeroA)
16 leia(numeroB)
17 media <- (numeroA + numeroB)/2
18 escreva(media)
19
20 Fimalgoritmo
```

On the right side, there is a table titled "Áreas das variáveis de memória (Globais e Locais)".

Escopo	Nome	Tipo	Valor
GLOBAL	NUMEROA	I	5
GLOBAL	NUMEROB	I	2
GLOBAL	MEDIA	R	3,5000000000000000

Below the table is a section titled "Área de visualização dos resultados". It shows the execution output:

```
Início da execução
---- Média-----5
2
3.5
Fim da execução.
```

- **Vamos praticar:**

1. Crie um algoritmo que leia um número e mostre o mesmo na tela.
2. Crie um algoritmo que receba 3 números e apresente a média entre eles.
3. Construa um fluxograma que pede para digitar o seu nome. O fluxograma deve apresentar na tela a frase: "O nome digitado foi:" e completar com o nome digitado.
4. Construa um fluxograma que lê um número real do teclado. Se este número for maior que 5,5 o programa deve imprimir "É maior", se for igual a 5,5 o programa deve imprimir "É igual", e se for menor que 5,5 o programa deve imprimir "É menor"

