
AGENDA 3

**DESENVOLVIMENTO
LÓGICA DE
PROGRAMAÇÃO**

GEEaD - Grupo de Estudos de Educação a Distância

Centro de Educação Tecnológica Paula Souza

GOVERNO DO ESTADO DE SÃO PAULO

EIXO TECNOLÓGICO DE INFORMAÇÃO E COMUNICAÇÃO

CURSO TÉCNICO EM DESENVOLVIMENTO DE SISTEMAS

LÓGICA DE PROGRAMAÇÃO

Expediente

Autores:

KELLY CRISTIANE DE OLIVEIRA DAL POZZO

Revisão Técnica:

Eduardo Chagas Ferreira

São Paulo – SP, 2025



MERGULHANDO NO TEMA

A lógica está presente em nosso cotidiano em diversas situações. Vamos analisar o seguinte contexto:

Uma pessoa tem a rotina de acordar cedo, tomar banho, tomar café e sair para o trabalho, podemos considerar tal rotina um exemplo clássico de atividades que requerem uma sequência lógica na aplicação dos afazeres, para que haja sucesso em sua realização.

Analisando o contexto, o objetivo desta tarefa era “sair para trabalhar”, sendo que, para que isso fosse possível, algumas instruções foram necessárias serem realizadas para o sucesso da tarefa.

Em nosso cotidiano para chegarmos a um resultado seguimos um “conjunto de instruções”, na informática não é muito diferente, uma instrução (comando) indica a um computador uma ação elementar a executar.



Fonte: <https://br.freepik.com/>

Instruções lógicas

Uma instrução isolada não permite realizar o processo completo, para seguir uma sequência lógica é necessário um conjunto de instruções colocadas em ordem sequencial lógica.

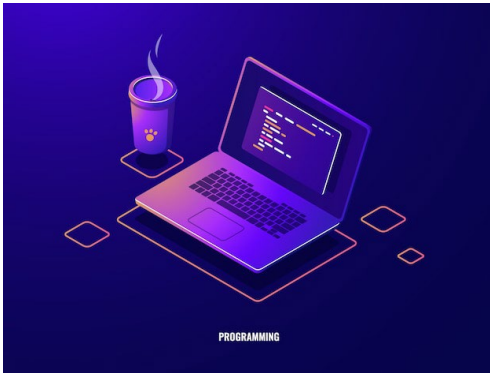
É evidente que temos que seguir as instruções em uma ordem adequada para realizar uma determinada tarefa.

Imagine a tarefa beber um refrigerante em lata por exemplo:



Para chegar a um resultado é necessário ordem sequencial lógica. A mesma coisa acontece com a lógica de programação. Quando pensamos em iniciar uma programação para computador, temos que ter em mente que a máquina desconhece totalmente alguns conceitos que para nós são óbvios. Por isto, devemos descrever cada passo de forma detalhada, por mais simples que possa parecer, para que tenha uma sequência lógica na programação e assim, o computador executar todas as instruções necessárias para uma determinada tarefa.

Por que a lógica de programação é importante?



Fonte: <https://br.freepik.com/>

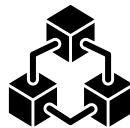
A lógica para programação é o processo procurar pensar na mesma sequência em que o computador executa as tarefas. Seguindo a lógica de programação procuramos imaginar como as ações serão executadas partindo-se do estudo de um problema até chegar à solução dele por meio da construção de um algoritmo. Por isso, a lógica de programação é tão importante!

Ilustrando a sequência lógica.

As instruções lógicas devem ser executadas em uma ordem adequada para o sucesso de uma programação. Uma instrução tomada fora de ordem perde o sentido e influência no resultado, precisamos colocar em prática o conjunto de todas as instruções, na ordem correta, para tal podemos utilizar os algoritmos.

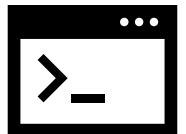
Algoritmos

Sequência finita de passos que levam a execução de uma tarefa. Podemos pensar em um algoritmo como uma receita, ou seja, apresenta uma sequência de instruções para um fim específico. Essas instruções devem ser descritas de maneira simples e objetiva.



Programas de Computadores

Os programas de computador são algoritmos escritos em uma linguagem de programação, como por exemplo, Java, C#, Python entre outras. Essas instruções são interpretadas e executadas por um computador.



Características Básicas de um Algoritmo

Todo algoritmo tem por padrão que apresentar algumas características básicas.

- ✓ **Ponto inicial e ponto final:** Todo algoritmo deve ter um ponto inicial e chegar a um ponto final.
- ✓ **Não ser ambíguo:** A leitura de um algoritmo tem que ser clara, não podendo ter dupla interpretação.
- ✓ **Tratar dados externos:** O algoritmo tem que ter o poder de receber dados externos e ser capaz de retornar resultados aos mesmos.
- ✓ **Etapas alcançáveis:** O algoritmo deve ter suas etapas alcançáveis em algum momento da programação.

Ao iniciar um algoritmo, é necessário dividir a situação problema em três fases fundamentais.



0

Formas de Representação

Podemos representar algoritmos estruturados de três formas:

<p>Descrição Narrativa</p> <p>Forma de representação utilizada para descrever um algoritmo de forma que o receptor da informação entenda do assunto mesmo não conhecendo sobre algoritmos, mas sim entendendo e interpretando as instruções.</p>	<p>Passo 1: Receber os números que serão multiplicados;</p> <p>Passo 2: Multiplicar os números;</p> <p>Passo 3: Exibir resultado da multiplicação.</p>
<p>Fluxograma</p> <p>Forma de representação que utiliza símbolos gráficos para representar os algoritmos. Existem símbolos padronizados para cada tipo de instrução, como por exemplo, início, entrada de dados, processamento, entre outros.</p>	<p>Fluxograma</p> <pre> graph TD INICIO([INICIO]) --> LEIA1[/LEIA NUM 1/] LEIA1 --> LEIA2[/LEIA NUM 2/] LEIA2 --> PROC[RESULTADO = NUM1 * NUM2] PROC --> SAIDA[/RESULTADO/] SAIDA --> FIM([FIM]) </pre>
<p>Pseudocódigo</p> <p>Também conhecido como Portugol ou Português Estruturado, sendo a principal porta de entrada para a linguagem de programação, esse formato consiste na definição de uma pseudolinguagem de programação, cujos comandos são em português para representar algoritmos.</p>	<p>Declare num1, num2, resultado;</p> <p>Leia num1, num2;</p> <p>resultado = num1+num2;</p> <p>Escreva Resultado.</p>

**Na prática**

Pronto! Agora que já sabemos o que é um Algoritmo e suas formas de representação. Para praticar escreva um algoritmo na representação de descrição narrativa para fritar um ovo. Faça este algoritmo como uma sequência de no mínimo 15 passos.



Quanto mais detalhado for as instruções do algoritmo para o computador, mais rápido e fácil ele compreenderá e irá executá-las chegando assim no objetivo final.

Variáveis e Constantes

Quando iniciamos programação temos que nos preocupar onde armazenar as informações. Para isso temos as variáveis e constantes.

Variáveis

Variável é um recurso utilizado na programação para armazenar e recuperar dados, ou seja, é um espaço na memória que reservamos atribuindo um nome, organizando os dados manipulados no programa. Por exemplo, podemos criar uma variável chamada “nome” que irá armazenar o nome de uma pessoa, fazendo uma analogia, seria como você criar uma gaveta de escritório com repartições, onde a gaveta em si, seria a memória e cada repartição uma variável para armazenar algum objeto.

Chamamos o espaço alocado na memória de variável, porque o valor armazenado neste espaço pode ser alterado ao longo do tempo, ou seja, o valor ali alocado é “variável” ao longo da execução do programa.

Constantes

Uma constante é responsável por armazenar um valor fixo, em um espaço da memória, sendo que esse valor não se altera durante a execução do programa. Um exemplo clássico é o valor de PI, suponha que você precise trabalhar com o número PI, que é um valor fixo de aproximadamente 3.14. Você pode simplesmente declará-lo e utilizá-lo em todo o seu programa.

Tipos de Dados

Ao criarmos uma variável em um programa temos que especificar o tipo de dado que será armazenado na mesma. Por exemplo, uma variável que irá armazenar o nome de uma pessoa será do tipo “caractere”, pois armazena letras, já uma variável que armazena a idade de uma pessoa, somente poderá receber números inteiros, portanto deve ser declarada como sendo do tipo “inteiro”. Já uma variável “valor”, como está representando um espaço para armazenar número com casas decimais, deve ser declarada como sendo do tipo “real”.

Os tipos de dados influenciam na forma como o algoritmo irá trabalhar, o desempenho do algoritmo e o seu consumo de memória.

Ao utilizarmos o português estruturado temos os seguintes tipos de dados:

INTEIRO	Qualquer número inteiro, negativo, nulo ou positivo. Exemplo: -21, 0, 10, 5025...
REAL	Qualquer número real no formato decimal, negativo, nulo ou positivo. Exemplo: -0.5, 0, 5, 9.5...
CARACTER	Qualquer conjunto de caracteres alfanuméricos. Exemplo: “Fabio”, “São Paulo”, “VZ32”, “123”...
LÓGICO	Conjunto de valores (FALSO ou VERDADEIRO)



A declaração do tipo da variável, vai depender da linguagem de programação que está sendo utilizada, como por exemplo, Linguagem C, C# e Java deve ser especificado o tipo de dados, já em PHP, ASP e JavaScript não é necessário. Cada tipo de linguagem também tem seus tipos e formatos de declaração próprios.

Comandos de Entrada e Saída

Na maioria das vezes os sistemas são construídos a partir de dados adquiridos através da interação humana, e os resultados também devem ser apresentados. Para interação as linguagens de programação fornecem comandos para inserção e visualização de informações. Os comandos de entrada e saída são os que permitem a interação com o usuário através dos dispositivos de saída.

Assim como na declaração de variáveis cada linguagem de programação tem seus comandos próprios para entrada e saída de dados.

Na descrição de algoritmo utilizamos os seguintes comandos:

Entrada	Saída
O comando LEIA é utilizado para que o usuário informe um valor a ser atribuído em uma variável do Sistema.	Para escrita de dados ou mensagens utilizamos o comando ESCREVA para escrita em uma linha, e ESCREVAL para a escrita com quebra de linha ao final.

Portugol Studio

Agora que já compreendemos os conceitos fundamentais de entrada, processamento e saída, além dos tipos de variáveis, chegou o momento de praticarmos a lógica de programação utilizando uma ferramenta Portugol Studio. O Portugol Studio é um ambiente de desenvolvimento integrado (IDE) que permite escrever e executar algoritmos usando uma linguagem semelhante ao português estruturado. Ele é amplamente utilizado para ensinar programação de forma intuitiva e visual.

O Portugol Studio é gratuito e pode ser baixado diretamente do site oficial do projeto:

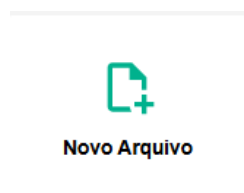
<https://univali-lite.github.io/Portugol-Studio/>

Mas também tem disponível a versão online, que funciona diretamente no navegador sem a necessidade de instalação. Acesse o Portugol WebStudio: <https://portugol.dev/>



A página inicial traz de forma intuitiva informações para criação de novos arquivos e exemplos práticos.

Clique em Novo Arquivo:



Um novo pseudocódigo é exibido:



The screenshot shows the Portugol Webstudio interface with a single file named 'Sem título'. The code editor contains the following pseudocode:

```

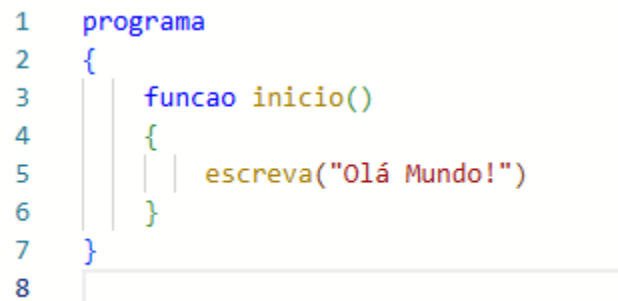
1  programa {
2      funcao inicio() {
3
4      }
5  }
6

```

Vamos compreender a estrutura:

- **programa {}**: define o início do código.
- **funcao inicio() {}**: indica onde o programa começa a ser executado.
- **escreva("texto")**: exibe uma mensagem na tela.

Estruture o código da seguinte forma:



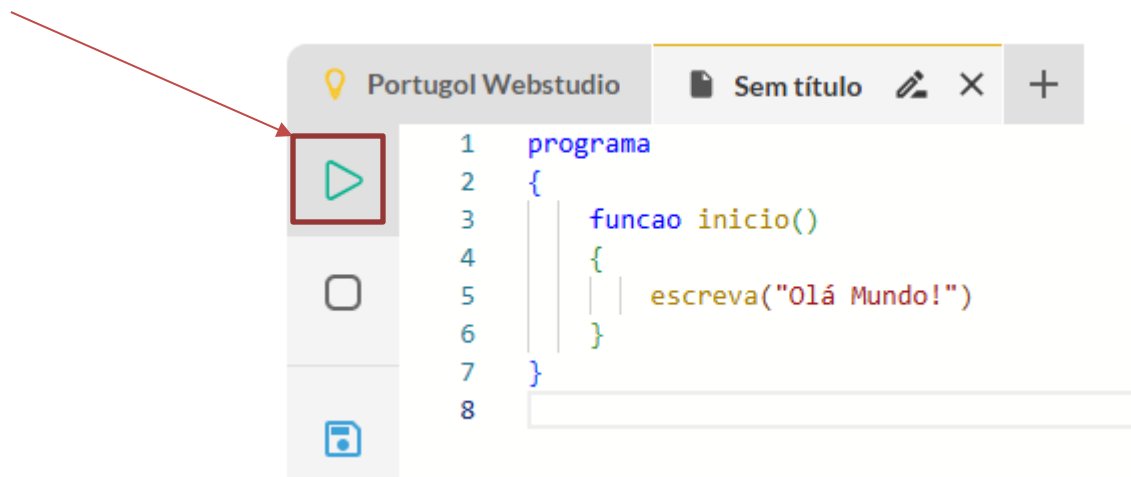
The screenshot shows the Portugol Webstudio interface with the following structured pseudocode:

```

1  programa
2  {
3      funcao inicio()
4      {
5          escreva("Olá Mundo!")
6      }
7  }
8

```

Clique no botão Executar disponível ao lado do código:



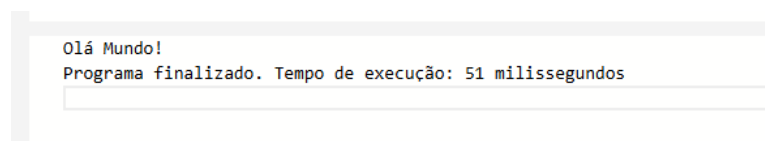
The screenshot shows the Portugol Webstudio interface with the same pseudocode as before. A red arrow points to the 'Executar' button (a green play icon) in the left sidebar. The code editor contains the following structured pseudocode:

```

1  programa
2  {
3      funcao inicio()
4      {
5          escreva("Olá Mundo!")
6      }
7  }
8

```

O resultado será exibido no Console:

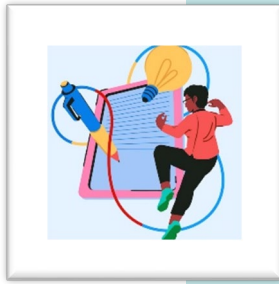


The screenshot shows the console output of the program execution:

```

Olá Mundo!
Programa finalizado. Tempo de execução: 51 milissegundos

```

**Na prática!**

crie um programa que solicite ao usuário seu nome e depois exiba uma mensagem personalizada. Use os seguintes comandos:

```
programa
{
    funcao inicio()
    {
        cadeia nome

        escreva("Digite seu nome: ")
        leia(nome)

        escreva("Olá, ", nome, "! Seja bem-vindo ao mundo da
programação!")
    }
}
```

Experimente modificar o código para perguntar também a idade do usuário e exibir uma mensagem como:

"Olá, Luana! Você tem 20 anos!"



Em linguagens de programação, o tipo utilizado para armazenar textos pode variar. No Portugol, usamos o tipo cadeia, enquanto em linguagens como Java, utilizamos String.

No Portugol Studio, o tipo cadeia é utilizado para armazenar textos, ou seja, sequências de caracteres. Podemos atribuir um valor a uma variável do tipo cadeia utilizando o operador igual (=). Exemplo: nome = "João".

Vamos a mais um exemplo prático no PortugolStudio:

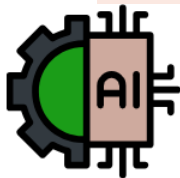
```
1  programa
2  {
3      funcao inicio()
4      {
5          cadeia nome, curso
6
7          escreva("Digite seu nome: ")
8          leia(nome)
9
10         escreva("Digite seu curso favorito: ")
11         leia(curso)
12
13         escreva("Olá, ", nome, "! Seu curso favorito é ", curso, ".")
14     }
15 }
```

Após Executar, confira o resultado no Console.

Confira os exemplos disponíveis na página inicial do PortugolStudio, em Mais Exemplos.



[Abrir Exemplo](#)



Utilize ferramentas de Inteligência Artificial para pesquisar mais sobre entrada e saída de dados em algoritmos. Solicite a apresentação de exemplos utilizando o PortugolStudio. Utilize o Fórum de Colaboração e Apoio disponibilizado pela professor tutor para compartilhar os exemplos encontrados.