Introdução a Algoritmos

Aldo Henrique

Quem sou

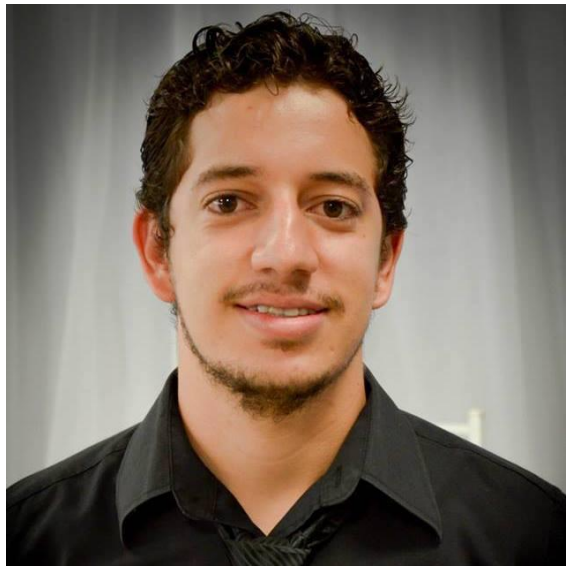
Prof. Aldo Henrique

- **Séries**
- **Filmes**
- **Música**

2011 - 2014
- Graduação em Sistemas de Informação.
    - Universidade Federal de Viçosa, UFV, Brasil.

2015 - 2017
- Mestrado em Informática.
    - Universidade de Brasília, UnB, Brasil.

2019 - 2022
- Doutorando em Informática.
    - Universidade de Brasília, UnB, Brasil.



Atuação Profissional
- Pesquisador;
- Professor;
- Desenvolvedor.

aldohenrique.com.br

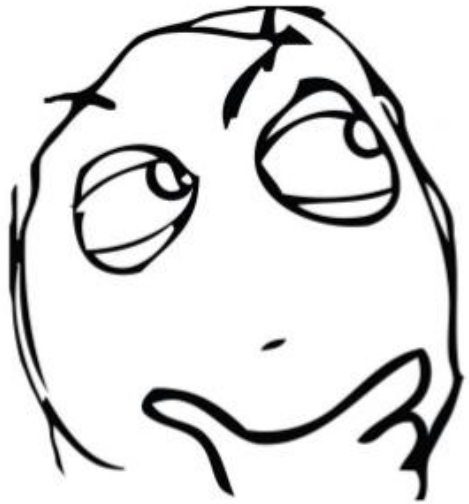
portalprogramando.com.br

youtube.com/PortalProgramando

Portal Programando

YouTube

Quem são vocês e por que estão aqui?



Balcão de Informação

aldohenrique.com.br
portalprogramando.com.br
aldoh.ti@gmail.com

Objetivo básico da computação

- Auxiliar os seres humanos em trabalhos repetitivos e manuais, diminuindo esforços e economizando tempo

- O computador é capaz de auxiliar em qualquer coisa que lhe seja solicitada, mas
  - Não tem iniciativa
  - Não é independente
  - Não é criativo nem inteligente
- É necessário que o computador receba suas instruções nos mínimos detalhes, para que tenha condições de realizar suas tarefas

Finalidade de um computador

- O computador deve receber, manipular e armazenar dados (todas essas operações são realizadas por meio de programas)
- Quando construímos um software para realizar determinado processamento de dados, devemos escrever um programa ou vários programas interligados
- Para que o computador consiga ler o programa e entender o que fazer, este programa deve ser escrito em uma linguagem que o computador entenda → LINGUAGEM DE PROGRAMAÇÃO

Etapas de desenvolvimento de um programa

- Análise
  - estuda-se o enunciado do problema para definir os dados de entrada, processamento e dados de saída
- Algoritmo
  - utiliza-se ferramentas do tipo descrição narrativa, fluxograma ou português estruturado para descrever COMO resolver o problema identificado
- Codificação
  - transforma-se o algoritmo em códigos na linguagem de programação escolhida

Algoritmo - Definição

- É uma sequência de instruções finita e ordenada de forma lógica para a resolução de uma determinada tarefa ou problema

Algoritmo 1 Troca de pneu do carro.

1: desligar o carro
2: pegar as ferramentas (chave e macaco)
3: pegar o estepe
4: suspender o carro com o macaco
5: desenroscar os 4 parafusos do pneu furado
6: colocar o estepe
7: enroscar os 4 parafusos
8: baixar o carro com o macaco
9: guardar as ferramentas

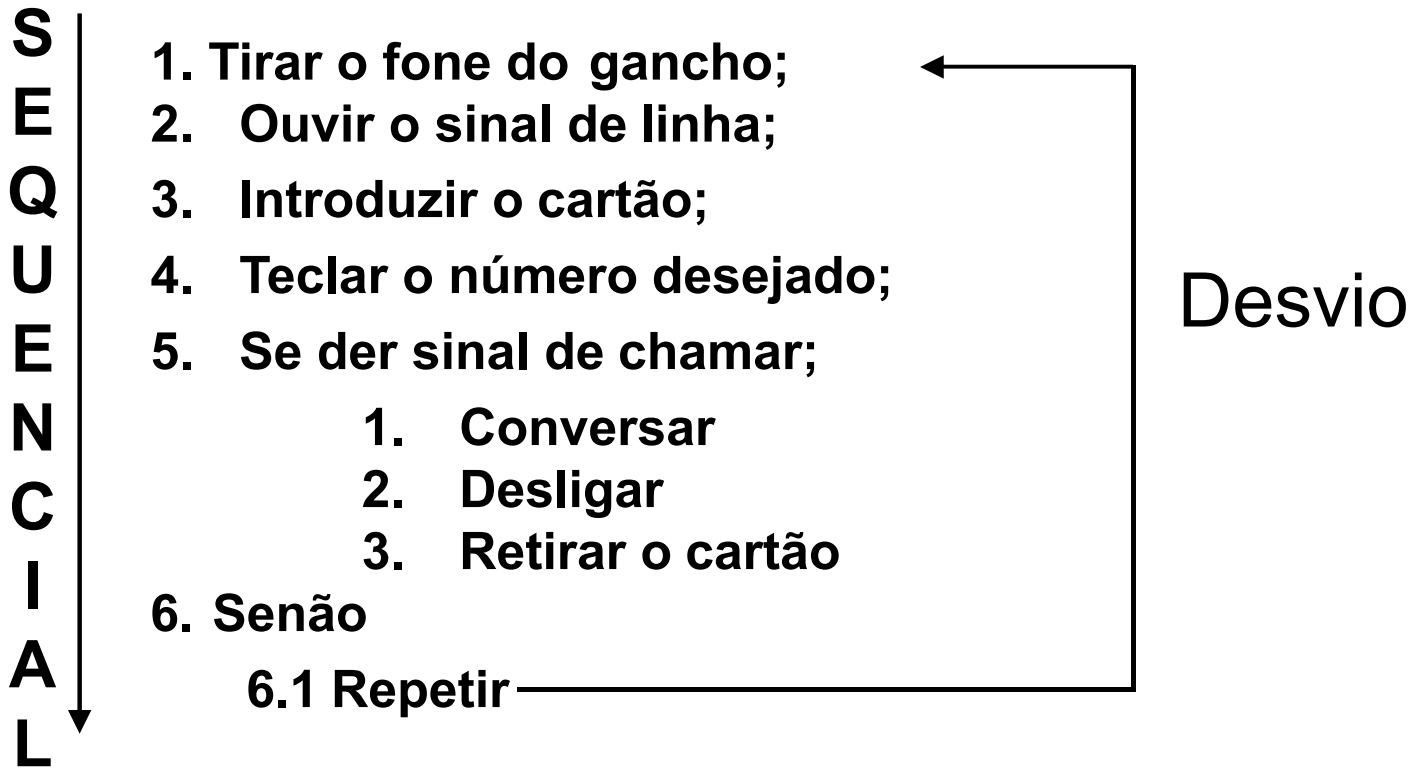
Algoritmo

- Não são operações exclusivas de um computador, pode ser aplicado a qualquer problema cuja solução possa ser decomposta em um grupo de instruções

- Exemplos de algoritmos
  - Instruções para se utilizar um aparelho eletrodoméstico;
  - Uma receita para preparo de algum prato;
  - Guia de preenchimento para declaração do imposto de renda;
  - Fazer um sanduíche
  - Trocar uma lâmpada
  - Sacar dinheiro em um banco 24 horas

Lógica da construção de algoritmos

Exemplo



Lógica da construção de algoritmos

Algoritmo para fritar um ovo

1. Colocar um ovo na frigideira
2. Esperar o ovo ficar frito
3. Remover o ovo da frigideira

Mais detalhado?

1. Retirar um ovo da geladeira
2. Colocar a frigideira no fogo
3. Colocar óleo
4. Esperar até o óleo ficar quente
5. Quebrar o ovo separando a casca
6. Colocar o conteúdo do ovo na frigideira
7. Esperar um minuto
8. Retirar o ovo da frigideira
9. Apagar o fogo

Algoritmo x Programa

- Para a grande maioria dos problemas, é possível haver mais de um algoritmo para solucionar um determinado problema
- Um programa é um conjunto de milhares de instruções que indicam ao computador, passo-a-passo, o que ele tem que fazer
- Um programa nada mais é do que um algoritmo computacional descrito em uma linguagem de programação

Características dos Algoritmos

- Todo algoritmo deve apresentar algumas características básicas
  - ter fim
  - não dar margem à dupla interpretação (não ambíguo)
  - capacidade de receber dado(s) de entrada do mundo exterior
  - poder gerar informações de saída para o mundo externo ao do ambiente do algoritmo
  - ser efetivo (todas as etapas especificadas no algoritmo devem ser alcançáveis em um tempo finito)

Formas de Representação dos Algoritmos

- Descrição Narrativa
- Fluxograma
- Pseudocódigo, também conhecido como Linguagem Estruturada ou Portugol

Descrição Narrativa

- Analisar o enunciado do problema e escrever, utilizando uma linguagem por natural (língua portuguesa, exemplo), os passos a serem seguidos para a resolução do problema

- Vantagem
  - Não é necessário aprender nenhum conceito novo

Descrição Narrativa

- Desvantagem

  Línguas naturais são sempre passíveis de ambiguidades, ou seja, podem gerar múltiplas

  interpretações, e dificultar a posterior transcrição do problema em código

  - Exemplo
    - Uma instrução “afrouxar ligeiramente os parafusos”, em um algoritmo da troca de pneu furado, está sujeita interpretações diferentes por pessoas distintas
    - Uma instrução mais precisa seria: “afrouxar a porca, girando-a de 30º no sentido anti-horário"

Descrição Narrativa

- Desvantagem
  - Extensão, normalmente, escreve-se muito para dizer pouca coisa
  - EXEMPLO: Receita de Bolo
    - Providencie manteiga, ovos, farinha, açúcar, etc.
    - Misture os ingredientes
    - Despeje a mistura na fôrma de bolo
    - Leve a fôrma ao forno
    - Espere 20 minutos
    - Retire a fôrma do forno
    - Deixe esfriar
    - Prove

Fluxograma

- Analisar o enunciado do problema e escrever, utilizando símbolos gráficos predefinidos, os passos a serem seguidos para a resolução dos problemas.

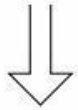
- É uma representação gráfica de algoritmos onde formas geométricas diferentes implicam ações (instruções, comandos) distintas.

Fluxograma

- Vantagem
  - Entendimento de símbolos gráficos é mais fácil que entendimento de textos
  - Desvantagens
    - É necessário aprender a simbologia dos fluxogramas
    - Em alguns casos, o algoritmo resultante não apresenta muitos detalhes, dificultando sua transcrição para um programa
    - Complica-se à medida que o algoritmo cresce

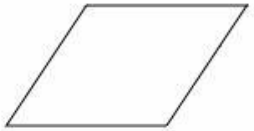
Fluxograma

Início e Fim do algoritmo

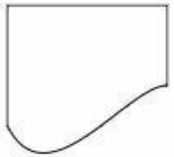


Sentido do fluxo de dados. Conecta símbolos ou blocos existentes

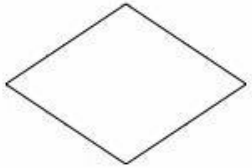
Processos em geral (cálculos ou atribuições de valores)



Entrada de dados

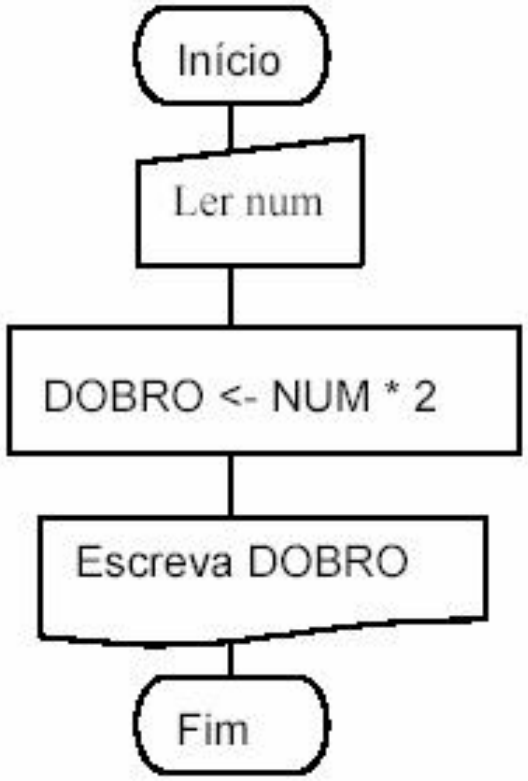


Saída de dados



Tomada de decisão, indicando a possibilidade de desvios

Fluxograma

EXEMPLO	EXPLICAÇÃO
	Início do algoritmo
	Entrada do número
	Cálculo do dobro do número
	Apresentação do resultado
	Fim do algoritmo

Pseudocódigo ou Portugol

- Analisar o enunciado do problema e escrever, por meio de regras predefinidas, os passos a serem seguidos para a resolução do problema
- Forma de representação de algoritmos rica em detalhes, apresenta a definição dos tipos das variáveis usadas no algoritmo
- Assemelhar-se bastante à forma em que os programas são escritos → bastante aceita

Pseudocódigo ou Portugol

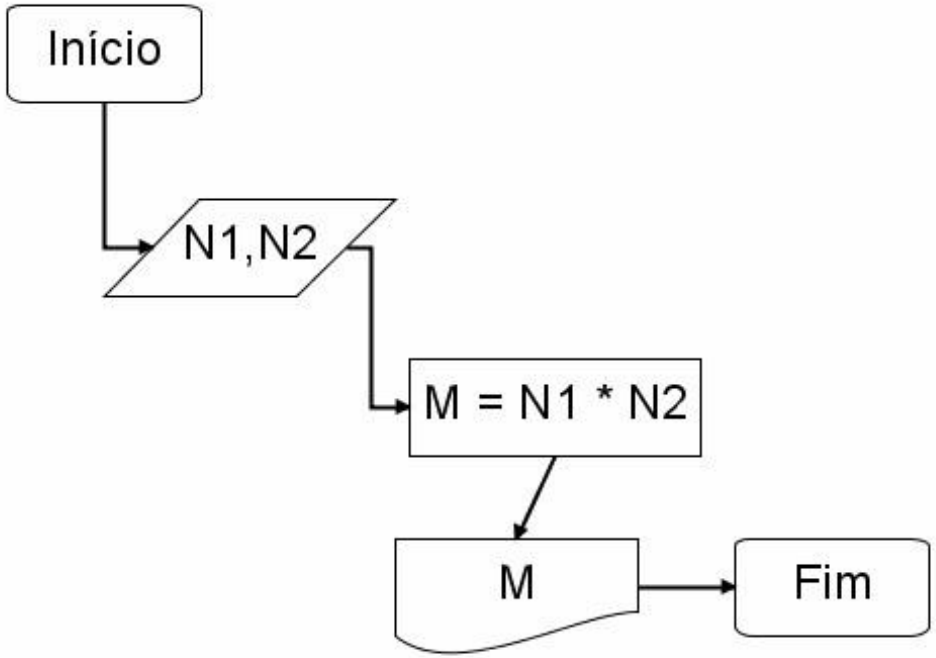
- Vantagem
  - A passagem para o código em linguagem de programação é quase imediata
- Desvantagem
  - Exige o aprendizado das regras do pseudocódigo

Pseudocódigo ou Portugol – Exemplo 1

- Faça um algoritmo para mostrar o resultado da multiplicação de dois números
  - Algoritmo em descrição narrativa
    - Passo1: Receber os dois números que serão multiplicados
    - Passo2: Multiplicar os números
    - Passo3: Mostrar o resultado obtido na multiplicação

Pseudocódigo ou Portugol – Exemplo 1

- Fluxograma



Pseudocódigo ou Portugol – Exemplo 1

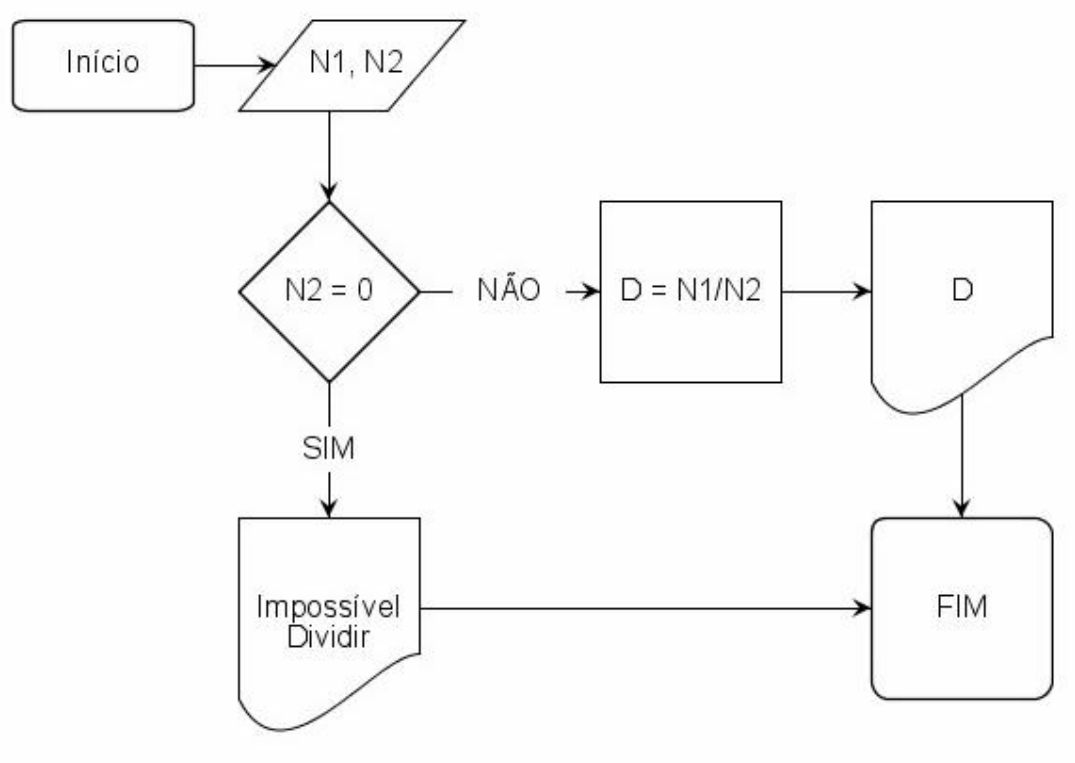
```
algoritmo "Multiplicação"

var
        n1,n2,m: inteiro
inicio
    escreva("Digite dois números:")
    leia(n1)
    leia(n2)
    m ← n1 * n2
    escreva("Multiplicação = ", m)
fimalgoritmo
```

Pseudocódigo ou Portugol – Exemplo 2

- Faça um algoritmo para mostrar o resultado da divisão de dois números
  - Algoritmo em descrição narrativa
    - Passo 1: Receber os dois números que serão divididos
    - Passo 2: Se o segundo número for igual a zero, não poderá haver divisão, pois não existe divisão por zero; caso contrário, dividir os números e mostrar o resultado da divisão

Pseudocódigo ou Portugol – Exemplo 2

- Fluxograma



Pseudocódigo ou Portugol – Exemplo 2

```
algoritmo "Divisão"
var
        n1, n2, D: real
inicio
        escreva( "Digite dois números:")
        leia(n1)
        leia(n2)
        se n2 = 0 então
            escreva("Impossível dividir.")
        senao
           D ← n1/n2
           escreva("Divisão = ",D)
        fimse
        fimalgoritmo
```

Variável

- Um algoritmo, e posteriormente um programa, recebe dados
- Os dados precisam ser armazenados no computador, para que possam ser utilizados no processamento
- O armazenamento dos dados é feito na memória

Variável

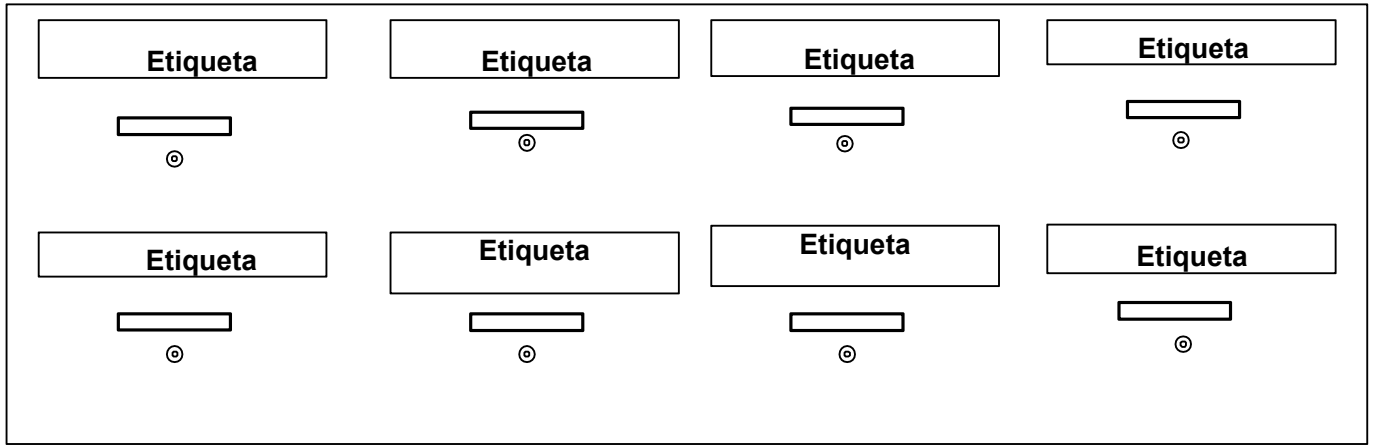
Os dados são armazenados em locais específicos da memória, denominados

ENDEREÇOS DE MEMÓRIA

- Quando uma operação aritmética recebe dois operandos, cada operando é armazenado em um endereço de memória diferente, para ser utilizado no cálculo

Variável

- Memória de um computador: um grande arquivo de gavetas



Variável

- Como cada endereço de memória pode armazenar dados várias vezes, ou seja, seu conteúdo pode variar, chamamos estas posições de variáveis
- Cada variável representa uma posição de memória, e possui um nome e um tipo
- O conteúdo da variável pode variar ao longo do tempo, durante a execução de um programa

Tipos de dados

- Os tipos de dados determinam quantos bytes de memória uma determinada variável ocupará
  - Numérico
  - Lógico
  - Literal ou caractere

Tipos de dados

- Numérico (inteiro ou real)
  - Podem representar números inteiros ou reais
  - Os números inteiros podem ser negativos ou positivos, e ocupam 2 bytes no computador
  - Desta forma, variam entre -32.767 e +32.768
  - Os números reais podem ser negativos ou positivos, e ocupam 4 bytes
- Lógico
  - Também chamados de booleanos
  - Podem assumir os valores VERDADEIRO e FALSO
  - Ocupam apenas 1 byte de memória

Tipos de dados

- Caractere
  - Dados formados por um caractere, ou uma cadeia de caracteres
  - Ocupam um byte de memória por caractere

Identificadores

- Representam os nomes das variáveis, dos programas, etc.
- Regras básicas para formação
  - Caracteres permitidos → números, letras maiúsculas ou minúsculas e o caractere sublinhado "_"
  - O primeiro caractere dever ser sempre uma letra ou o caractere sublinhado
  - Não são permitidos caracteres em branco e caracteres especiais (@,$,+,-,%,!)
  - Não é possível utilizar palavras reservadas nos identificadores, ou seja, palavras que pertencem à linguagem de programação utilizada
  - Exemplos
    - A, a, Nota, nota, nota_1, dia

Estrutura Sequencial

- Estrutura básica para algoritmos em pseudocódigo
- Os comandos do algoritmo fazem parte de uma sequência, onde é relevante a ordem na qual se encontram os mesmos, pois serão executados um de cada vez, estritamente, de acordo com essa ordem

Estrutura Sequencial

```
algoritmo “Nome_do_Algoritmo”
var
      declaração de variáveis
inicio
     Comando-1
     Comando-2
      :
      Comando-N
fimalgoritmo
```

Declaração de Variáveis

- Variáveis são declaradas logo após VAR e os tipos mais utilizados são inteiro, real, caractere e lógico

- Exemplo

```
x: inteiro
y: real
 c: caracter
d: logico
```

Atribuição

- Utilizado para atribuir um determinado valor ou operação à uma variável
- Representado pelo símbolo ←
- Exemplos
  - X ← 4
  - Y ← Y + 2
  - Z ← “aula”
  - Teste ← FALSO

Atribuição

```
algoritmo "soma"
var
     x, y, soma: inteiro
inicio

     x <- 5
     y <- 7
     soma <- x + y

fimalgoritmo
```

Atribuição

```
algoritmo "Simbolos"
var
simb1, simb2: caractere
inicio

    simb1 <- "$"
    simb2 <- "@"

    fimalgoritmo
```

Entrada, Processamento, Saída

- Passos para criar um programa em um computador
  - Entrada
  - Processamento
  - Saída

Entrada e Saída de Dados

- Entrada
  - O comando de entrada é utilizado para receber dados informados pelo usuário
  - Representado pela palavra **LEIA**
  - Exemplo
    - LEIA (x) (os dados informados pelo usuário serão armazenados na variável x)

Entrada de dados - Exemplo

```
algoritmo "soma"
var
     x, y, soma: inteiro
inicio

     leia(x)

     leia(y)
     soma <- x + y

fimalgoritmo
```

Entrada e Saída de Dados

- Saída
  - O comando de saída é utilizado para mostrar dados para o usuário, na tela do monitor, ou na impressora, entre outros
  - Representado pela palavra ESCREVA
  - Exemplo
    - ESCREVA(y) (mostra o valor armazenado na variável y)
    - ESCREVA(“Conteúdo de x é: ”,x)

Entrada de dados - Exemplo

```
algoritmo "soma"
var
     x, y, soma: inteiro
inicio


     leia(x)

     leia(y)
     soma <- x + y

     escreva("Resultado da soma: ", soma)

fimalgoritmo
```

Exemplo 1

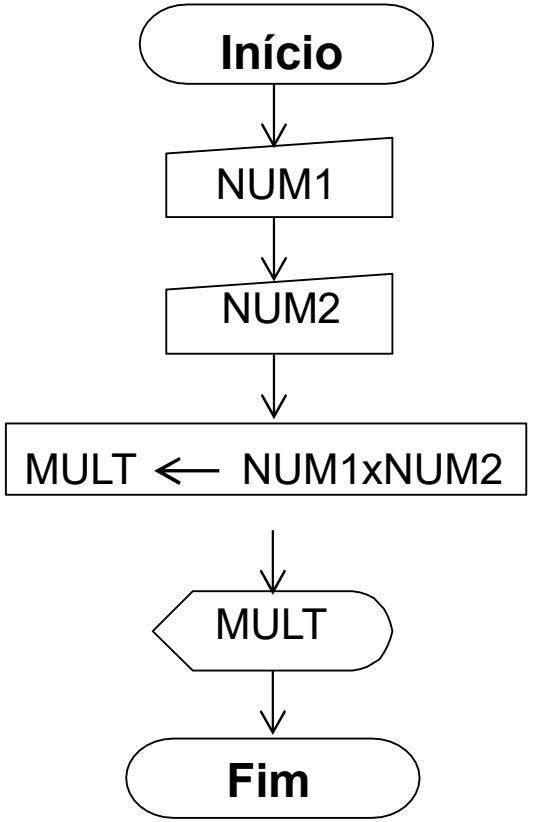
- **Enunciado:**

  “Deverá ser criado um programa que efetue a leitura de dois valores numéricos. Faça a operação de multiplicação entre os dois valores e apresente o resultado obtido.”

- **Descrição narrativa**:

  1- Criar variáveis: NUM1, NUM2, MULT
  2- Ler dois valores: NUM1 e NUM2;
  3 Efetuar a multiplicação das variáveis NUM1 e NUM2, atribuir o resultado a variável MULT;
  4 Exibir o valor da variável MULT.

Exemplo 1 - Fluxograma



Descrição narrativa:

1 Criar três variáveis: NUM1, NUM2 e MULT;
2 Ler dois valores, no caso variáveis NUM1 e NUM2;
3 Efetuar a multiplicação das variáveis NUM1 e NUM2, atribuir o resultado a variável MULT;
4 Exibir o valor da variável MULT.

Exemplo 1 - Portugol

- Descrição narrativa:
  1. Criar três variáveis: NUM1, NUM2, MULT;
  2. Ler dois valores, no caso variáveis NUM1 e NUM2;
  3. Efetuar a multiplicação das variáveis NUM1 e NUM2, atribuir o resultado a variável MULT.
  4- Exibir o valor da variável MULT.

```
algoritmo "multiplicação"
var
     num1, num2, mult: real
inicio
     leia(num1)
     leia(num2)
     mult <- num1 * num2
     escreva(mult)
fimalgoritmo
```

Exemplo 2

- Faça um programa que receba quatro números inteiros, calcule e mostre a soma desses números.

Exemplo 2

```
algoritmo “SomaInteiros”
var
      n1, n2, n3, n4, soma: inteiro
      inicio
            escreva( “Digite quatro números inteiros: ”)
            leia(n1)
            leia(n2)
            leia(n3)
            leia(n4)
            soma ← n1 + n2 + n3 + n4
           escreva(“A soma dos quatro números é: “, soma)
fimalgoritmo
```

Exemplo 3

- Faça um programa que receba três notas, calcule e mostre a média aritmética entre elas.

Exemplo 2

```
algoritmo “Media_Notas”
var
      n1, n2, n3, soma, media: real
      inicio
            escreva( “Digite as três notas: ”)
            leia(n1)
            leia(n2)
            leia(n3)
            soma← n1 + n2 + n3
            media ← soma/3
            escreva(“A média das notas é: “, media)
            fimalgoritmo
```

Exercícios

1) Resolva os problemas abaixo usando as 3 formas de representação de algoritmos: narrativa, fluxograma e português estruturado.

a)Escreva um algoritmo que receba três notas e seus respectivos pesos, calcule e mostre a média ponderada entre essas notas

a)Escreva um algoritmo que receba o salário de um funcionário, calcule e mostre o novo salário, sabendo-se que este sofreu um aumento de 25%

Exercícios

c) Escreva um algoritmo que receba o salário-base de um funcionário, calcule e mostre o salário a receber, sabendo-se que esse funcionário tem gratificação de 5% sobre o salário-base, e paga imposto de 7% sobre o salário-base.

d) Escreva um algoritmo que receba uma hora formada por hora e minutos, e calcule a hora digitada apenas em minutos.

Exercícios

e) Escreva um algoritmo que receba o ano de nascimento de uma pessoa e o ano atual, calcule e mostre a idade desta pessoa, e quantos anos essa pessoa terá em 2020

f) Cada degrau de uma escada tem X de altura. Escreva um algoritmo que receba essa altura e a altura que o usuário deseja alcançar subindo a escada. Calcule e mostre quantos degraus o usuário deverá subir para atingir seu objetivo, sem se preocupar com a altura do usuário