

Algoritmos

1

Algoritmo

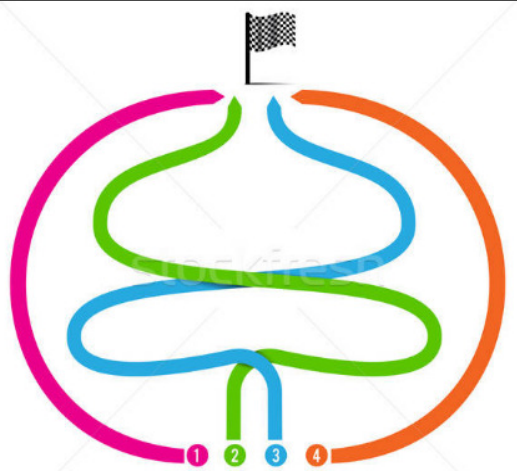
- ▶ É uma **sequência ordenada, finita e repetível de passos, que deve ser seguida para a realização de uma tarefa** (solução de um problema).



▶ 2

▶ Algoritmo não é a solução de um problema, pois, se assim fosse, cada problema teria um único algoritmo.

▶ Algoritmo é um caminho para a solução de um problema.



▶ 3

Importante! Lembre-se

O aprendizado de algoritmos/programação não se consegue a não ser através de muitos exercícios.

Algoritmos não se aprende:

- * Copiando Algoritmos
- * Estudando Algoritmos

Algoritmos só se aprendem:

- * Construindo Algoritmos
- * Testando Algoritmos



▶ 4

Características de um algoritmo

- ▶ Ter **início e fim**;
- ▶ Não dar margem a **dupla interpretação**;
- ▶ Deve **gerar informações de saída** para o mundo externo ao do ambiente do algoritmo;
- ▶ Ser **efetivo** (*ou seja, todas as etapas especificadas no algoritmo devem ser alcançáveis em um tempo finito*)

▶ 5

Vantagens do uso de algoritmos

- ▶ Um algoritmo pode ser implementado em qualquer linguagem de programação.
- ▶ Se representarmos **algoritmicamente** a solução de um problema, e depois o traduzirmos para uma **linguagem de programação**, teremos como resultado um **programa** de computador.

Algoritmo + Ling. Programação = Programa

▶ 6

Formas de representação de algoritmos

1. Descrição Narrativa;
2. Fluxograma Convencional;
3. Pseudocódigo, Linguagem Estruturada ou Portugol.

► 7

Formas de representação de algoritmos

I. Descrição Narrativa ou Linguagem Natural (LN)

► Uso do idioma/LN (Pt):

- + bastante conhecido;
- Impreciso;
- Más interpretações;
- Ambíguo;
- Extenso;

“Querido, vá ao mercado para mim e traga 2 ovos, se tiver leite, traga 6.”



► 8

Formas de representação de algoritmos

2. Fluxogramas

- ▶ É uma representação gráfica de algoritmos;
- ▶ Formas geométricas diferentes implicam ações (*instruções, comandos*) distintos.
- ▶ É considerada uma forma de representação intermediária

+ Ferramenta conhecida mundialmente;

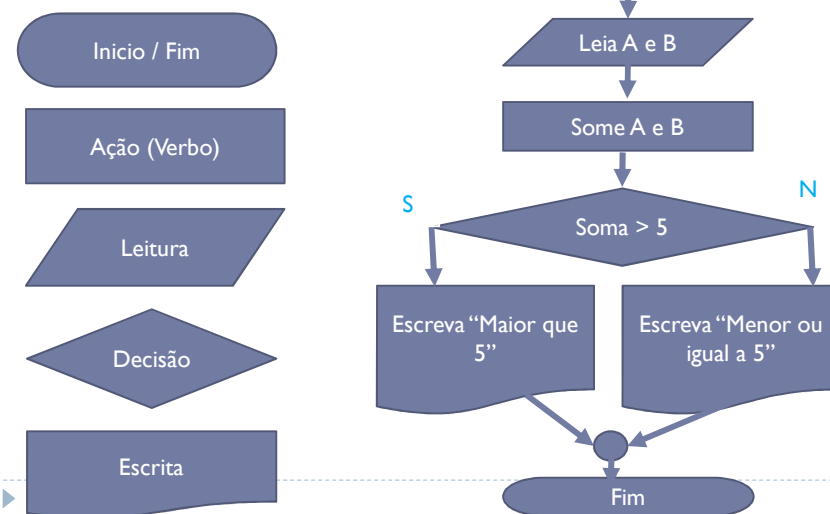
- Não se preocupa com detalhes de implementação do programa (p.ex.: tipos de dados).



▶ 9

Formas de representação de algoritmos

2. Fluxograma



Formas de representação de algoritmos

3. Pseudocódigo

- ▶ Linguagem intermediária entre linguagem natural e linguagens de programação para descrição de algoritmos.
- ▶ Utiliza a estrutura de linguagens de programação porém visa a interpretação por humanos e não por computadores.

```

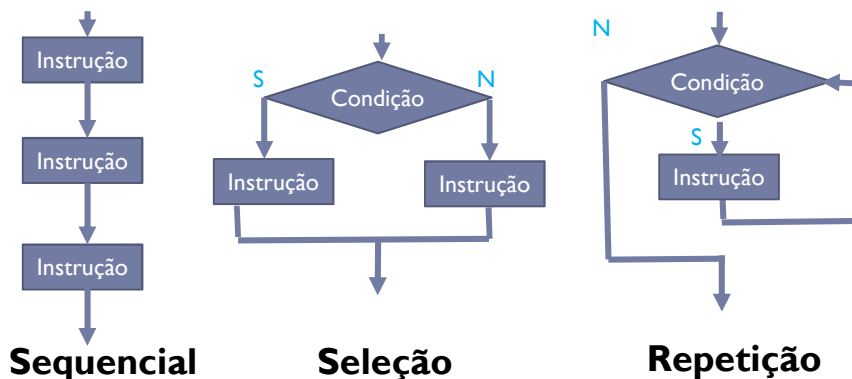
ALGORITMO SOMA
Entradas: A, B
Saídas: Soma
Declarar A, B: Inteiros
LEIA A
LEIA B
Soma ← N1 + N2
SE (Soma > 5)
    ESCREVA "Maior que 5"
SENÃO
    ESCREVA "Menor ou igual a 5"
FIM ALGORITMO
  
```

▶ 11

Estrutura de algoritmos

Teorema do Programa Estruturado:

Três estruturas são suficientes para representar qualquer função computável. (Fonte: Bohm, Jacoplini – Fluxogramas e Máquinas de Turing e linguagens ... <http://dl.acm.org/citation.cfm?id=365646>)



▶ 12

Raciocínio lógico na construção de algoritmos

Lógica

“A lógica nos ensina a colocar **ordem no pensamento**”



Pensar em Ordem

Lógica de programação

Possibilita a construção de algoritmos para solução de problemas pela ordenação do pensamento.

Lógica Matemática

Prova de declarações matemáticas

► 13

Raciocínio lógico na construção de algoritmos

Lógica de Programação:
ordem de instruções para
resolver um problema.

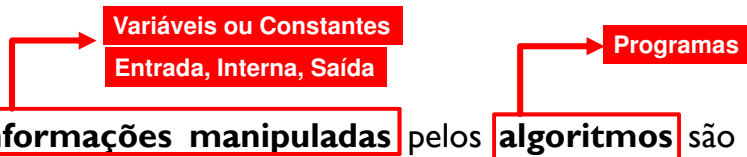
Lógica Matemática:
expressões que
podem ser avaliadas
como verdadeiras ou
falsas

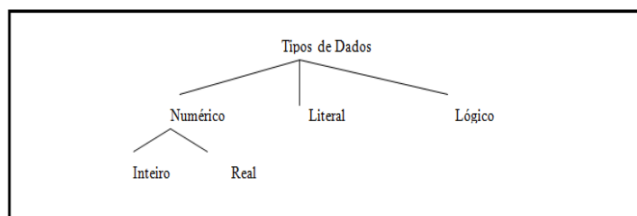
```
int main(int argc, char** argv)
{
    int a = 0, b = 0, soma = 0;
    scanf("%d, %d", &a, &b);
    soma = a + b;

    if (soma > 5)
        printf("%d é maior que 5\n", soma);
    else
        printf("%d é menor que 5\n", soma);
    return soma;
}
```

► 14

Tipos de Dados

- 
- ▶ As **informações manipuladas** pelos **algoritmos** são armazenadas de diferentes maneiras, dependendo do tipo de dados que ela representa.
 - ▶ Existem basicamente 3 **tipos de dados** primitivos:



▶ 15

Tipos de Dados

▶ TIPO NUMERICO

- ▶ **INTEIRO:** Toda e qualquer informação numérica que pertença ao conjunto dos números inteiros relativos;
 - ▶ Ex.: a idade de uma pessoa, o número de degraus de uma escada, etc.;
- ▶ **REAL:** Toda e qualquer informação numérica que pertença ao conjunto dos números inteiros reais;
 - ▶ Ex.: a altura de uma pessoa (em metros), o valor (em reais) do salário de um funcionário, etc.;
- ▶ **LITERAL:** Toda e qualquer informação composta por um conjunto de caracteres alfanuméricos;
 - ▶ Ex.: nome de uma pessoa, placa de um veículo, etc..

STRING

▶ 16

Tipos de Dados

▶ TIPO NUMÉRICO

- ▶ INTEIRO
- ▶ REAL

▶ LITERAL

- ▶ STRING

▶ LÓGICO

- ▶ Estes dados são chamados de **booleanos**, devido a significativa contribuição do BOOLE à área da lógica matemática;
- ▶ O tipo **lógico** é usado para representar dois únicos valores lógicos possíveis: **verdadeiro** ou **falso**.

▶ 17

Variáveis e Constantes

- ▶ **Variáveis:** elementos da computação que possuem nome, tipo, valor e endereço de memória. Seu valor pode ser alterado durante a execução de um programa.
- ▶ **Constantes:** semelhantes às variáveis, estas não permitem que seu valor seja alterado durante a execução de um programa.

▶ 18

Nomes das Variáveis

- ▶ Devem começar com uma letra ou um sublinhado “_”, mas **não** com um dígito numérico.
- ▶ Pode conter letras, dígitos e “_”
- ▶ **Não** são permitidos espaços
- ▶ Diferença entre maiúsculo e maiúsculo
- ▶ **Não** pode ser um nome reservado (veremos isso depois):
int, float, double, main...

▶ 19

Nomes das Variáveis

- ▶ **Dicas:**
 - ▶ Manter uma padronização na nomenclatura das variáveis
 - ▶ caixa baixa: idade, volume
 - ▶ Usar letras maiúsculas para separar múltipalavras:
 - ▶ volumeAgua, idadeMedia
 - ▶ Seja consistente: manter um padrão claro de nomenclatura de modo que outros entendam
 - ▶ Usar substantivos que digam o que a variável representa
 - ▶ Usar terminologia do domínio da aplicação

▶ 20

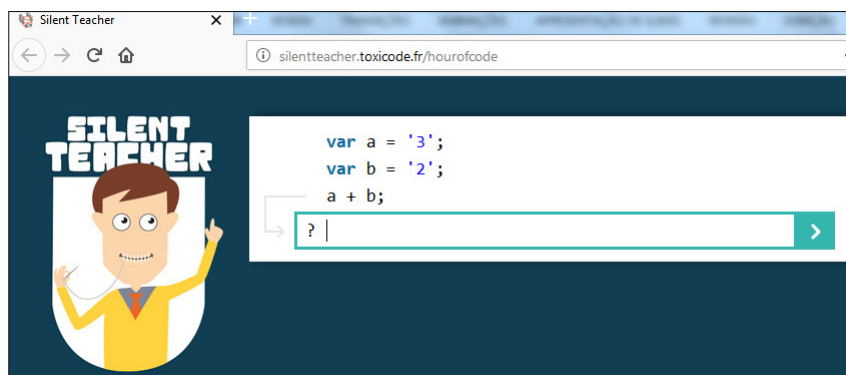
Variáveis



► 21

Silent Teacher – Hour of Code

► <http://silentteacher.toxicode.fr/hourofcode>



► 22

Como verificar se o algoritmo está correto?

► 23

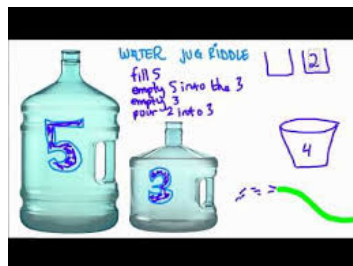
Teste de Mesa

- O teste de mesa simula a execução de um algoritmo manualmente. Para isso, você deve orientar-se por meio dos passos descritos a seguir:
 1. Crie uma tabela, onde cada coluna representa uma das variáveis do seu algoritmo;
 2. Execute manualmente/mentalmente seu algoritmo e preencha as linhas da tabela com os valores que as variáveis devem assumir.
 3. No final da execução do algoritmo a variável deve possuir o valor esperado.

► 24

Exemplo

- Elabore uma sequência de passos para obter 4L de água a partir de um galão de 5L e outro galão de 3L.



► 25

Teste de mesa

1. Inicie com os galões vazios.	G5L = ?	G3L = ?
2. Encha o galão de 5L.	G5L = ?	G3L = ?
3. Transfira o conteúdo do galão de 5L para o de 3L.	G5L = ?	G3L = ?
4. Descarte a água do galão de 3L.	G5L = ?	G3L = ?
5. Transfira o conteúdo do galão de 5L para o de 3L.	G5L = ?	G3L = ?
6. Encha o galão de 5L.	G5L = ?	G3L = ?
7. Transfira o conteúdo do galão de 5L para o de 3L.	G5L = ?	G3L = ?

► 26