

CURSO DE ALGORITMOS

PO
R JUNIOR CRISTE



INFORMATICODE

WWW.INFORMATICODE.COM.BR

WWW.YOUTUBE.COM/INFORMATICODE

WWW.INSTAGRAM.COM/INFORMATICODE

*Esse material é gratuito e pode ser distribuído apenas no site oficial.
Para maiores informações: www.informaticode.com.br

NOTAS IMPORTANTES

- Apostila criada por Junior Criste.
- Não poderá ser redistribuída, apenas o site oficial tem os direitos.
- Material totalmente gratuito e de apoio as aulas disponíveis em www.youtube.com/informaticode.
- Site oficial: www.informaticode.com.br
- Programa utilizado [VisuAlg 3.0.7](#).

AULA 1

O que são Algoritmos?

Algoritmos são sequências lógicas que tem o objetivo de solucionar algo de forma eficiente.

O que é Linguagem de Máquina?

Linguagem de máquina (ou linguagem binária) é a forma própria de linguagem que a máquina suporta. Ela é formada de "zeros" e "uns", ou seja é binária. Porém é muito complexa, não facilmente entendida por humanos, então para isso existe a linguagem de programação, que é uma *linguagem intermediária*.

O que é Linguagem de Programação?

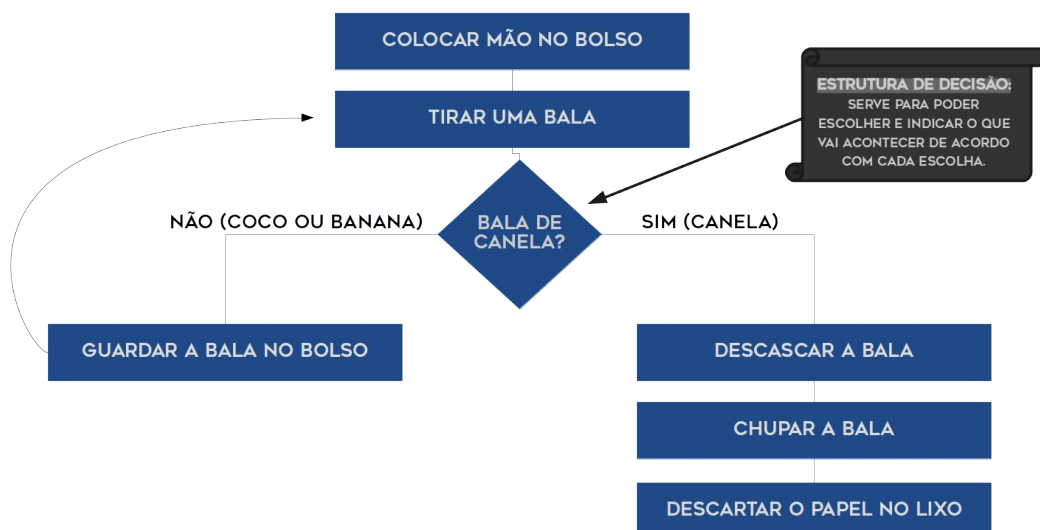
Como a linguagem de máquina era muito complexa para o homem, então criou-se a linguagem de programação. Ela é considerada uma linguagem intermediária, pois é de fácil entendimento do homem e consegue conversar com a máquina. Como exemplo temos as linguagens Pascal, C, C++, C#, Java, Python, Lua, Elixir, JavaScript, PHP, TypeScript, Ruby, Swift, entre outras tantas.

Algoritmos no dia a dia

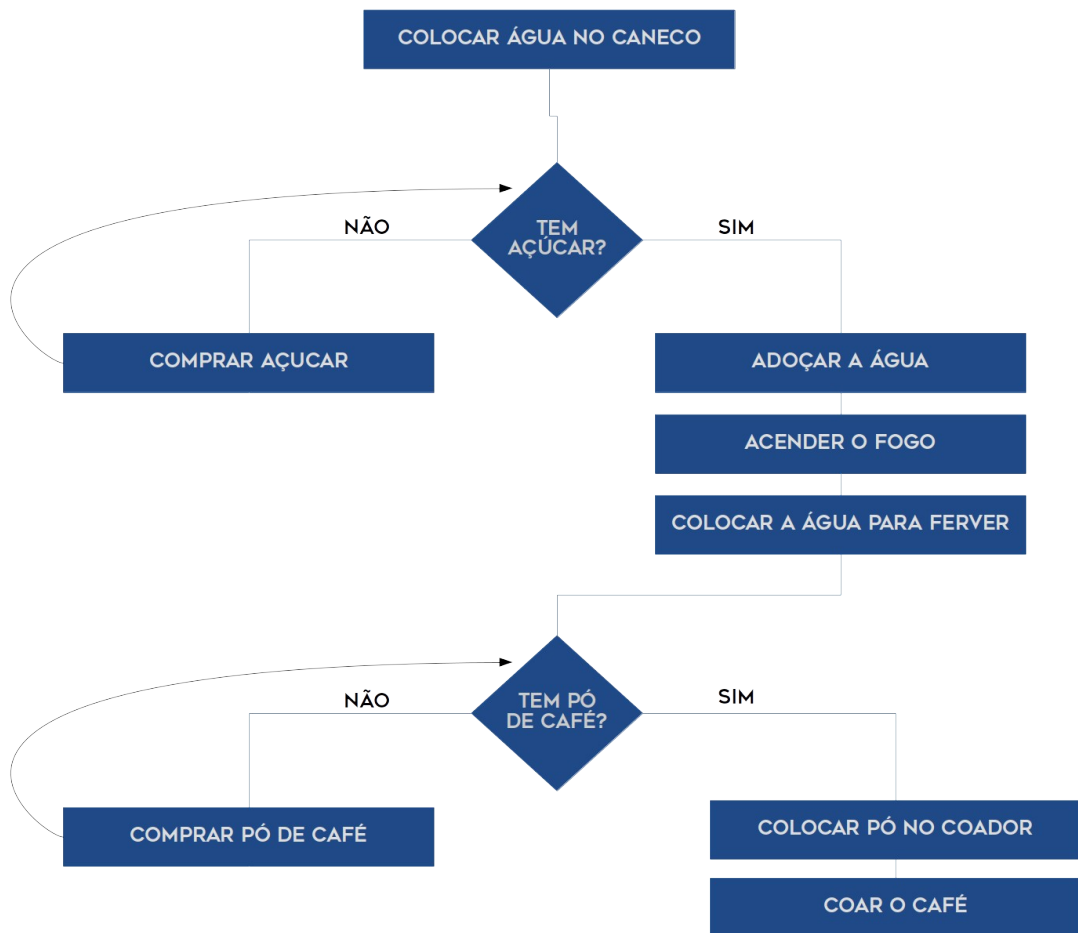
Imagine que no dia a dia exista rotinas, essas rotinas seguem um passo a passo, ou seja, são sequências, sabendo disso podemos considerá-las como algoritmos.

EXEMPLOS

Imagine que você precise pegar uma bala no bolso. Seu bolso está cheio delas, com três sabores diferentes: coco, banana e canela. Você quer chupar uma bala de canela. Faça um algoritmo disso, supondo que você poderá pegar apenas uma bala por vez.



Agora vamos fazer a mesma coisa, porém com uma receita de café. Supondo que o açúcar e o pó de café podem estar em falta, logo sabemos que novamente iremos usar a estrutura de decisão.



Nos dois exemplos acima foram usadas formas geométricas para simbolizar os processos, etapas dos algoritmos. Porém são meramente para exemplificar e não estão seguindo ao pé da letra as regras para criação de fluxogramas, até por que seria muito confuso neste momento estudarmos precocemente fluxogramas. Então não se preocupe ainda com isso.

CURIOSIDADES

As 20 linguagens de programação mais usadas em 2020 (até março) pela lista Tiobe:

Mar 2020	Mar 2019	Change	Programming Language
1	1		Java
2	2		C
3	3		Python
4	4		C++
5	6	↑	C#
6	5	↓	Visual Basic .NET
7	7		JavaScript
8	8		PHP
9	9		SQL
10	18	⬆	Go
11	14	↑	R
12	12		Assembly language
13	17	⬆	Swift
14	15	↑	Ruby
15	11	↓	MATLAB
16	22	⬆	PL/SQL
17	13	↓	Perl
18	20	↑	Visual Basic
19	10	↓	Objective-C
20	19	↓	Delphi/Object Pascal

Tabela 1: Acessível em: <https://www.tiobe.com/tiobe-index/>

AULA 2

O que é Pseudocódigo?

Um pseudocódigo é uma forma mais simples de converter algoritmos em resultados, utilizado em grande escala para estudo. Aqui usaremos o *Visualg* com o *portugol* como programa de pseudocódigo. Ou seja, não chega a ser uma linguagem de programação, mas é necessária para uma melhor aprendizagem.

O que é Visualg?

Visualg é um programa desenvolvido pelo professor Antônio Carlos Nicolodi, que tem como função trabalhar com pseudocódigos, no caso o popular *portugol*, que simula de forma traduzida para o português o que seria possível fazer em linguagens de programação, como o Pascal ou C.

O que são Variáveis?

Variável é o nome dedicado ao espaço que vai armazenar um dado dentro de um programa, por exemplo um nome ou um valor. No Visualg elas podem ser classificadas como *inteiro*, *real*, *caractere* e *lógico*.*

TIPO	DEFINIÇÃO	EXEMPLOS		
INTEIRO	GUARDA VALORES NUMÉRICOS, SEM CASAS DECIMAIS	8	17	500
REAL	GUARDA VALORES QUEBRADOS, OU SEJA, COM CASAS DECIMAIS	3,5	18,9	10,5
CARACTERE	GUARDA PALAVRAS, OU APENAS LETRAS	A	TIGRE	JUNIOR CRISTE
LÓGICO	GUARDA UM VALOR QUE VERDADEIRO OU FALSO.	VERDADEIRO		FALSO

**Existem mais tipos de variáveis, mas no Visualg são apenas essas. Você verá mais tipos quando começar a estudar alguma linguagem de programação.*

O que são Operadores?

Um operador é um símbolo que na matemática indica uma operação. No Visualg temos operadores aritméticos, relacionais, lógicos e de caractere.

OPERADORES ARITMÉTICOS	
+,-	Operadores unários, isto é, são aplicados a um único operando. São os operadores aritméticos de maior precedência. Exemplos: -3, +x. Enquanto o operador unário - inverte o sinal do seu operando, o operador + não altera o valor em nada o seu valor.
\	Operador de divisão inteira. Por exemplo, $8 \setminus 3 = 2$.
+, -, *, /	Operadores aritméticos tradicionais de adição, subtração, multiplicação e divisão. Por convenção, * e / têm precedência sobre + e -. Para modificar a ordem de avaliação das operações, é necessário usar parênteses como em qualquer expressão aritmética.
MOD ou %	Operador de módulo (isto é, resto da divisão inteira). Por exemplo, $10 \text{ MOD } 3 = 1$.
^	Operador de potenciação. Por exemplo, $5 ^ 2 = 25$.

OPERADORES RELACIONAIS	
=	Igual a
<	Menor que
>	Maior que
<=	Menor ou igual a
>=	Maior ou igual a
<>	Diferente de

OPERADORES LÓGICOS	
nao	Operador de negação. Uma coisa que é "nao VERDADEIRO" é "FALSO".
ou	Operador que resulta VERDADEIRO quando um dos seus dois valores for verdadeiro.
e	Operador que resulta VERDADEIRO somente se seus dois valores forem verdadeiros.
xou	Operador que resulta VERDADEIRO se seus dois operandos lógicos forem diferentes, e FALSO se forem iguais.

OPERADOR DE CARACTERES	
+	Operador de concatenação do tipo "caractere". Por exemplo: "Rio " + " de Janeiro" = "Rio de Janeiro".

AULA 3

Estrutura de um Pseudocódigo

O pseudocódigo, diferente de um algoritmo, tem a obrigação de seguir uma estrutura básica, com início, meio e fim.

O **início** é onde temos nome e declaração das variáveis;

O **meio** é onde acontece todo o processamento, onde acontece todo o programa em si;

E o **fim** é o fechamento apenas.

Observe abaixo a estrutura padrão, que já vem quando abrimos o Visualg.

```
1 Algoritmo "se nome"
2 //
3 //
4 // Descrição : Aqui você descreve o que o programa faz!
5 // Autor(a) : Nome do(a) aluno(a)
6 // Data atual : 12/3/2020
7 Var
8 // Seção de Declarações das variáveis
9
10
11 Início
12 // Seção de Comandos, procedimento, funções, operadores, etc...
13
14
15 Fimalgoritmo
```

Agora observe a mesma estrutura com as cores rosa, vermelha e amarela, que separam o início (rosa), meio (vermelho) e fim (amarelo).

```
1 Algoritmo "se nome"
2 //
3 //
4 // Descrição : Aqui você descreve o que o programa faz!
5 // Autor(a) : Nome do(a) aluno(a)
6 // Data atual : 12/3/2020
7 Var
8 // Seção de Declarações das variáveis
9
10
11 Início
12 // Seção de Comandos, procedimento, funções, operadores, etc...
13
14
15 Fimalgoritmo
```

Como você pode observar a estrutura tem algumas *palavras reservadas*.

Palavras Reservadas é o nome que damos a comandos de uma determinada linguagem. Essas palavras não podem ser usadas em nomes de variáveis, apenas serão usadas em suas respectivas funções.

A parte importante do programa começa realmente na declaração das variáveis. Primeiro declara-se o nome da variável, depois o tipo dela, por exemplo: "NOME: CARACTERE", "IDADE: INTEIRO", "NOTA1: REAL" ou "APROVADO: LOGICO".

```
7 Var
8 // Seção de Declarações das variáveis
9 nome: caractere
10 idade: inteiro
11 nota1: real
12 aprovado: logico
13
```

Uma regra é que nunca um nome de variável se comece com número.

O texto em verde recebe o nome de comentário, ele não será executado, serve apenas para comentar algo dentro do código, ou seja o usuário não tem acesso a isso. Para

comentar basta escrever duas barras e em seguida o que se deseja, por exemplo: "// esse é um comentário".

O visualg aceita tanto letras maiúsculas, quanto minúsculas. Já outras linguagens tem suas próprias regras quanto a isso.

Entrada, Processamento e Saída

Já falamos de início, meio e fim. Agora seremos ainda mais específico, a entrada, o meio e a saída ocorrem no meio do programa.

Entrada é onde recebemos informações, por exemplo quando alguém digita um nome, nota ou qualquer outro tipo de dado.

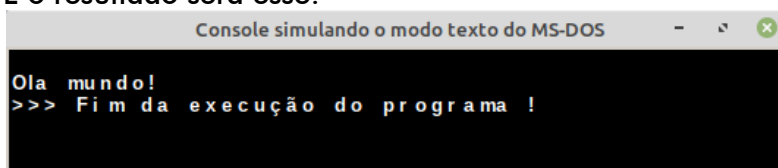
Processamento é a parte em que vamos trabalhar com esses dados, por exemplo, se eu for fazer uma conta, é no processamento que ficarão os comandos dessa operação matemática. Mas não se limita apenas a isso. Processamento é qualquer alteração que eu faça em valores de variáveis, independente do tipo dela.

E por fim temos a **Saída**, essa é responsável para apresentar a solução do problema, no caso da soma é o resultado. Qualquer solução apresentada é uma saída.

Para escrever alguma coisa na tela para que o programa possa se comunicar com o usuário é necessário usar o comando "ESCREVA" seguido de parênteses, e dentro desses parênteses existem aspas duplas, dentro dessas abas o texto que vai aparecer na tela. Observe:

```
13
14 Início
15 // Seção de Comandos, procedimento, funções, operadores, etc...
16 escreva("Ola mundo!")
17
18 Fim algoritmo
```

E o resultado será esse:



```
Console simulando o modo texto do MS-DOS
Ola mundo!
>>> Fim da execução do programa !
```

Agora para que haja um retorno, por exemplo, quando você solicitar que o usuário digite o nome dele é preciso salvar isso em uma variável, para isso usamos o comando "LEIA" seguido de parênteses e o nome da variável dentro. Observe:

```
13 escreva("Digite seu nome ")
14 leia (nome)
```

Acabamos de fazer um processo de **entrada**!

EXEMPLOS

Agora vamos solicitar o nome e sobrenome dessa pessoa. Salvaremos isso em variáveis. Também será necessário outra variável para concatenar (juntar) os dois valores (nome e sobrenome) em apenas um valor.

Para fazermos isso, será necessário então três variáveis, vamos chamá-las de "nome", "snome" e "ncompleto", sendo referentes ao nome do usuário, sobrenome e o nome junto com o sobrenome.

Após capturar o *nome* e o *snome* iremos atribuir isso ao *ncompleto*.

Mas como faremos isso?

Usaremos a atribuição. Atribuir é dizer o que será feito com aqueles dados, ou seja, entramos na parte de **processamento**.

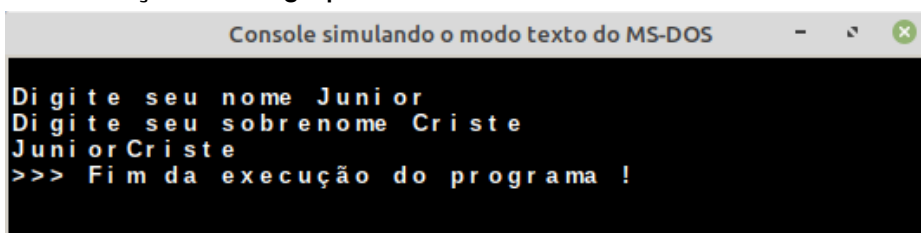
Para atribuir o valor de *ncompleto* devemos seguir uma estrutura. Primeiro coloca-se o nome da variável, depois o símbolo de atribuído que é " \leftarrow ", basicamente desenhar uma seta voltada a variável que vai receber o valor. E em seguida colocamos o valor. No caso vai ser a concatenação de nome com *snome*, ficará assim: "*ncompleto* \leftarrow nome + *snome*". O que acabamos de fazer foi o processamento, onde indicamos que o nome completo do usuário é a junção do nome com o sobrenome.

Agora que já temos a solução, precisamos de mostrá-la ao usuário, isso é o processo de **saída**. Para mostrar um valor na tela é preciso usar novamente o comando "ESCREVA" seguido dos parênteses, porém dessa vez SEM as aspas duplas e com o nome da variável na tela. As aspas duplas não são usadas nesse caso, pois não está sendo apresentado um texto fixo, e sim um valor de variável. Lembrando que a variável apresentada aqui será a solução, ou seja, a *ncompleto* que acabou de receber um valor na etapa de processamento.

Vejamos como isso fica na prática:

```
1 Algoritmo "Nome Completo"
2 //
3 //
4 // Descrição : Aqui você descreve o que o programa faz!
5 // Autor(a) : Nome do(a) aluno(a)
6 // Data atual : 12/3/2020
7 Var
8 // Seção de Declarações das variáveis
9 nome, snome, ncompleto: caractere
10
11 Inicio
12 // Seção de Comandos, procedimento, funções, operadores, etc..
13 escreva("Digite seu nome ")
14 leia (nome)
15
16 escreva("Digite seu sobrenome ")
17 leia (snome)
18
19 ncompleto  $\leftarrow$  nome + snome
20
21 escreva(ncompleto)
22
23 Fimalgoritmo
```

E a execução será algo parecido com isso:



```
Console simulando o modo texto do MS-DOS
Digite seu nome Junior
Digite seu sobrenome Criste
JuniorCriste
>>> Fim da execução do programa !
```

Observe que na declaração das três variáveis, todas ficaram em apenas uma linha, separadas por vírgulas. Isso devido o fato de serem do mesmo tipo, então podem ficar juntas, porém poderiam estar separadas também, é apenas uma questão de prática e estética. Elas foram declaradas do tipo "CARACTERE", pois são nomes, e no Visualg nomes são considerados caracteres, em outras linguagens usamos o tipo "STRING" para isso.