**Fundamentos de Algoritmos**

**Tipologia e Variáveis**

Qual a função do computador?

**Informação** = composta por **dados** e **instruções**

**Dados**

Tratados e processados = numéricos, caracteres e lógicos

**Numéricos**

* Inteiros: 0, 1, 5, 50, 800, -50, -32, -888...
* Reais: 5.95, 9.54, -8.8, -0.555, 10, 8, -7, -55...

**Caracteres**

Ex: Programação, PROGRAMAÇÃO, KU\*&NH53!g, Fone: (51)991919441, Rua Jacinto Gomes

**Lógico**

* **Verdadeiro – 1**
* **Falso – 0**

Português estruturado:

* .Verdadeiro .V ou .S
* .Falso .F ou .N

**Variável**

É uma estrutura que irá receber um tipo de dado, mas não tem certeza do seu valor

* É mutável
* Possui variações
* Inconstante
* Incerto
* Instável

Pode assumir qualquer um dos valores de um determinado conjunto de valores, mas está restrita ao seu tipo. Se é do tipo numérica, vai receber numérica. Se é do tipo string, vai receber string.

**Vai identificar qual um determinado dado e o tipo daquele dado**

**Regras** (boas práticas)

* Atribuição de um ou mais caracteres
* Primeira letra – não número
* Sem espaços em branco
* Vedado (utilização de palavras reservadas)
* Caracteres e números

**Papéis de uma variável**

* Ação – modificação o estado do algoritmo, o estado do programa
* Controle – vigiada, controle de alguma estrutura

Variável é tudo aquilo que é fixo ou estável.

**Instruções primitivas**

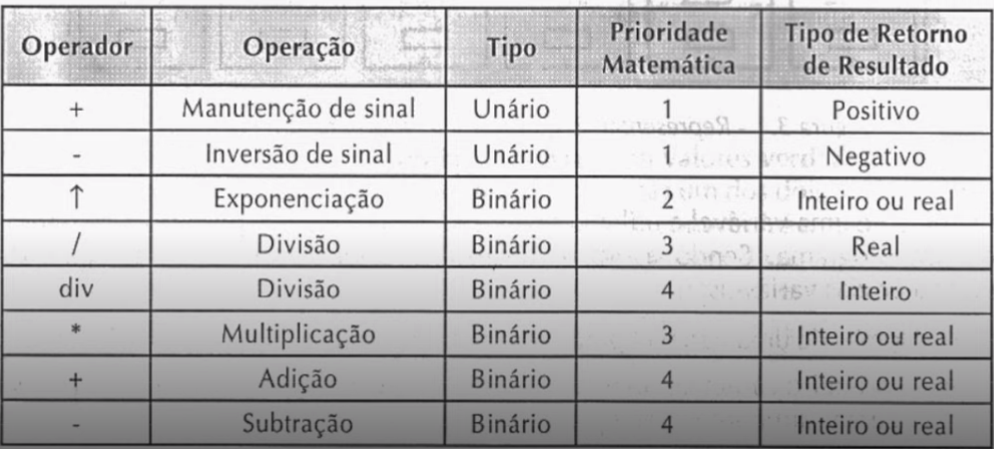
As instruções vão determinar as ações que iremos executar em cima dos nossos dados

**Cálculos matemáticos**

* Variáveis
* Constantes

**Operadores**

* Binário
* Unário



**Definição formal:** instruções são linguagens de **palavras-chave** de uma determinada programação que tem por finalidade comandar um computador que irá **tratar os dados**

**Linguagens de programação**

* janela – português
* window – inglês
* ventana – espanhol

**Exemplo:**

Média escolar

*Início programa:*

*Nota1* **(variável)** *= 5*

*Nota = 8*

*Resultado = 0*

*Resultado = (Nota1 + Nota2) / 2* **(constante)**

*Escreve resultado*

*Fim programa*

*Saída: 6.5*

**Estrutura Condicional**

Estado de uma pessoa ou coisa – **Condição**

**Condicional:**

* que expressa uma condição ou suposição
* contém ou implica uma suposição ou hipótese

Condição: **Ok?** **Operação**

Condição: **Não?** **Exceção**

**Estrutura Condicional:**

**Simples:**

* Condição > Operação

**Composta:**

* Condição > Exceção OU Operação

**Encadeada:** se e senão um dentro do outro

* Condição > Condição ou Operação
* Condição ou Exceção

**Operadores Relacionais:**

* = igual a
* <> diferente de
* > maior que
* < menor que
* >= maior ou igual a
* <= menor ou igual a

**Condicional simples:**

Se (<condição>) então

<instruções para condição verdadeira>

fim\_se

*Início Programa:*

*A = 0*

*B = 0*

*X = 0*

*Leia A*

*Leia B*

*X = A + B*

*Se (X > 10)*

*Escreva X*

*Fim se*

*Fim programa*

**Condicional Composta**

Se (<codição>) então

<instruções para condição verdadeira>

Senão

<instruções para condição falsa>

fim\_se

*Início programa:*

*Nota1* **(variável)** *= 5*

*Nota = 8*

*Resultado = 0*

*Resultado = (Nota1 + Nota2) / 2* **(constante)**

*Escreve resultado*

*Fim programa*

*Saída: 6.5*

**Condicional Encadeado**

Se (<condição 1>) então

<instruções para condição verdadeira>

Senão

Se (<condição 2>) então

<instruções para: condição 2 verdadeira e condição 1 falsa>

Senão

<instruções para condição 1 e 2 falsas>

fim\_se

*Início programa:*

*A = 0*

*B = 0*

*X = 0*

*Leia A*

*Leia B*

*X = A + B*

*Se (X >= 10)*

*X = X-7*

*Senão*

*X = X+5*

*Fim se*

*Fim programa*

**Operadores Lógicos:** and, or e not

Quando utilizar?

* Verificação de V ou F
* Substituição
  + Encadeamento de condições

**AND – operador lógico**

Condição verdadeira

* Todas devem ser satisfeitas

Ex: Curso de inglês

Se (gramatica .e .conversacao) então

Escreva “aprovado”

Senão

Escreva “reprovado”

**OR – operador lógico** - UNIÃO

Condição verdadeira

* Apenas uma das condições deve ser verdadeira

Ex:

Se (choveu .ou .grama\_molhada) então

Escreva “Plantas regadas”

Senão

Escreva “Regar plantas”

**NOT – operador lógico**

* Operador de negação
* Inversão do resultado lógico

Ex: Not B -> tudo que não está em B

**Estruturas de Repetição**

**Trecho de um programa:**

* Laços
* Controle de fluxo
* Malhas de repetição
* Repetição
* Loop

**Condições de parada:**

* Número de repetições pré-fixada
* Condição a ser satisfeita

**Vantagens:**

* Redução de linhas
* Compreensão facilitada
* Redução de erro

Ex:

Enquanto ... faça

Repita ... até

Para ... de ... até ... faça

Ex: **cortar grama alta**

Enquanto

Teste lógico

* Início

Número de repetições

* Indefinidas

**Loop: “aparar grama” e “analisar grama”**

*Grama = falso*

*Enquanto (grama == falso) faça*

*<instrução de cortar grama>*

*<atualiza grama>*

*Fim enquanto*

Ex: **procurar artigo**

Repita

Teste lógico

* Final

Número de repetições

* Indefinidas

**Loop: “virar página” e “analisar conteúdo”**

Ex:

Para ... de ... até

Teste

* Início

Número de repetições

* Definidas

*somatorio = 0*

*para inicio = 1 até 10*

*somatorio = somatorio + inidio*

*escreva somatorio*

*fim para*

**Mesclagem de estruturas:**

*Enquanto (<condição>)*

*Se (<condição2>)*

*<instruções>*

*Fimse*

*...*

*Fim enquanto*

**Vetores e Matrizes**

“Um vetor é caracterizado por uma variável dimensionada com tamanho pré-fixado”

“Matrix é uma tabela organizada em linhas e colunas no formato M x N, onde M representa o número de linhas (horizontal) e N o número de colunas (vertical)”

**Matrizes são:**

* Coleção de variáveis
* Contiguas em memória
* Índices

**Ex:** média escolar

*Nota11 = 10*

*Nota 21 = 5*

*Nota 12 = 7*

*Nota 22 = 8*

*notas\_aluno1 = [10,5]*

*notas\_aluno2 = [7,8]*

*notas\_alunos = [10,5,7,8] //a cada duas posições um novo aluno*

**Ex: menor quantidade de linhas**

*Matriz\_alunos [6] [6] //onde está a média do aluno*

*Escreva “ID aluno”*

*Escreva matriz\_alunos [0] [0] //onde está o número do aluno*

*Escreva “média”*

*Escreva matriz\_alunos [0] [6]*

**Funções**

Podem também ser chamadas de: **subalgoritmo, função, bloco, subprograma, método, sub-rotina**

É similar ao conceito de função matemática

“As **funções**, ou sub-rotinas, são blocos de instruções que **realizam tarefas específicas**”

* Modularização do programa
* Código mais claro e conciso
* Reutilização de instruções

“São blocos de instruções (código), identificados por **nomes e parâmetros**”

* Definição
* Nome
* Invocação
* Variável local //são destruídas ao encerrar a função

**Os dados são enviados para a função e a função retorna com o resultado**

A função altera o estado do programa.

Ex: média escolar

*funcao mediaescolar(nota1,nota2)*

*resultado = 0*

*resultado = (nota1 + nota2)/2*

*retorne resultado*

*fim funcao*

*aluno1 = mediaescolar(nota11,nota21)*

*aluno2 = mediaescolar (nota12,nota22)*

*aluno3 = mediaescolar(nota13,nota23)*

**Instruções de entrada/saída**

Eu entrego os dados (input/entrada) para o computador, que irá processá-los e dar o resultado (output/saída)

**Entrada:**

“Consiste na inserção e recebimento de dados do mundo real por meio de ação de alguma interface, seja teclado, mouse, arquivos, entre outros.”

**Saída:**

“Consiste na impressão dos dados do mundo abstrato, digital por meio de ação de alguma interface.

Os formatos podem variar desde simples arquivos binários até complexas querys de banco de dados”

**Tipos de saída:**

* Saída programada = **condicional** (aguarda o dispositivo) ou **incondicional**
* Saída por interrupção = definida pelos periféricos

**Casos de saída na programação:**

* Bem sucedida
* Erro de sintaxe ou outro
* Erros de programação
* Problemas com a interface