

Algoritmo

É uma sequência de ações finitas que descrevem como um problema deve ser resolvido.

Um Algoritmo, precisa:

- Ter início e fim
- Ser escrito em termos de ações (comandos) bem definidas
- Que as ações sigam uma sequência ordenada.

Técnicas de apresentação de Algoritmos

- Em uma língua (português, inglês): apresenta um inconveniente: a ambiguidade de alguns termos.
- Em uma Linguagem de Programação: porém apresenta alguns inconvenientes: utiliza apenas as instruções existentes na LP.
- Representações Gráficas:
 - Fluxograma
 - Diagramas de Nassi-Shneiderman (utilizaremos neste curso)
 - Método de Jackson
 - Diagramas de Warnier-Or

- Pseudo-linguagem(ou pseudo-código) : não tem os inconvenientes da ambigüidade de uma língua,nem os rigores de uma LP.É um português estruturado c/ “frases(comandos)” correspondentes as estruturas básicas de programação.

Vantagem da Utilização de algoritmos:

A partir dele o programador poderá implementá-lo em qualquer linguagem de programação que conheça ou deseje.

Objetivos das Técnicas de Construção de Algoritmos

Estruturados:

- facilita o desenvolvimento e entendimento dos algoritmos;
- facilita a leitura e entendimento dos algoritmos pelas pessoas;
- antecipa a comparação de sua correção;
- facilita manutenção;
- permite que o desenvolvimento de algoritmos possa ser empreendido simultaneamente por uma equipe de pessoas.

Pseudo-Código : Elementos e Comandos Básicos

Identificadores:

Conjunto de caracteres (letras, números ou símbolos) normalmente iniciados por uma letra, que identificam ou nomeiam de forma clara uma constante, variável, tipo, arquivo, módulo, algoritmo etc...

Ex.: Nome, Nota1, Salario, TotAlunos etc

Tipo de Dados:

Constante (Const):

É um identificador que armazena um valor fixo e imutável durante a execução de um algoritmo ou programa.

Variável (var):

É um identificador que possui um conteúdo variável durante a execução de um algoritmo ou programa.

Tipo Básico do Dado :

- Inteiro (“integer”) : uma variável deste tipo poderá armazenar qualquer número não fracionário(inteiro) positivo ou negativo;
- Real (“real”) : uma variável deste tipo poderá armazenar um número fracionário ou não fracionário(real) qualquer;
- Caracter (“char”) : uma variável deste tipo poderá armazenar um caracter qualquer;
- Texto ou cadeia de caracteres (“string”) : uma variável deste tipo poderá armazenar uma cadeia de caracteres de qualquer tamanho.
- Lógico(“boolean”) : uma variável deste tipo poderá armazenar um valor lógico - verdadeiro(1) ou falso(0)

Tipos Básicos de Dados

Inteiro : 2, 3, -1; 0

Real : 2 ; -5,8 ; 9,21 ; 0

Caracter : “ A ” ; “ B ” ; “ * ” ; “ 1 ”

Texto: “ ANA ” ; “ PUC ”

Lógico : Verdadeiro; Falso

Declaração de Variáveis e Constantes no Dicionário de Dados

Identificadores das vars. deste tipo : Tipo da variável

Ex.:

```
NUM, QUANTALUNO: inteiro;  
SALÁRIO, MÉDIA : real;  
NOME, NOMEPROF : texto;  
ACHOU : lógico;  
TIPOCONSULTA: caracter;
```

Palavras Reservadas

São palavras que terão uso específico no nosso pseudo-código e que não deverão ser usadas como identificadores, para não causar confusão na interpretação.

Ex.: Algoritmo, Inteiro, Real, Se, Então, Senão

Comando de Atribuição:

Permite que se forneça ou altere o valor de uma determinada variável, onde o tipo desse valor tem de ser compatível ao tipo da variável.

Ex.: Soma \leftarrow 100

Operadores

Aritméticos

+ : Soma

- : Subtração

*** : Multiplicação**

/ : Divisão

**** : Potênciação**

m mod i : resto da divisão de m por i

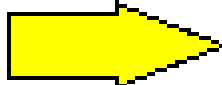
m div n : quociente inteiro da divisão de m por n


Operadores

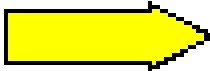
Aritméticos - Exemplos

7 + 5.5



7.0 + 5.5  12.5

7 / 2  3.5

6 / 2  3.0

7 div 2  3

7 mod 2  1

Ex.: $5/2 = 2,5$

$m \text{ MOD } i$ (ou resto) : dá o resto da divisão inteira
de m por i .

$$\text{ex.: } 10 \bmod 4 = 2$$

$$6 \bmod 2 = 0$$

$m \text{ DIV } n$ (ou quociente) : dá o quociente inteiro da
divisão de m por n

$$\text{ex.: } 20 \text{ div } 6 = 3$$

Operadores Relacionais

Operando

Operando

Resultado

qualquer

\neq
 \wedge
 \vee
 \diamond
 \equiv

qualquer

lógico

Exemplos:

Num \leftarrow 10

Nome \leftarrow Paula

Num $<$ 6



É falso { 6 não é maior que 10 }

Nome $<$ "Zazá"



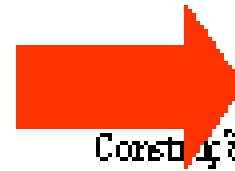
É verd { Paula vem antes q. Zazá }

$(2 * 6) \geq 7$



É verdadeiro

Num $<>$ $2 * 5$



É falso

Construção

Operadores

Lógicos

Operando

Operando

Resultado

lógico

e
ou
não

lógico

lógico

Exemplos

Achou \leftarrow falso

Num \leftarrow 9

$(1 \geq 9)$ e $(20 \geq 3)$



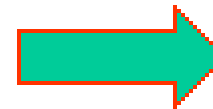
F e V = F

Não Achou



V

$(4 \geq 5)$ ou $(5 \geq 3)$



F ou V = V

$(\text{Num} \leq 3)$ ou $(\text{Num} \leq 8)$



F ou F = F

Prioridade entre os operadores

Nível de prioridade	Operadores
Zero	Parenteses e Funções.
01	“ + “ e “ - “ unitários
02	“ ** “ potenciação
03	“ * “ e “ / “
04	“ + ” e “ - ”
05	relacionais
06	não
07	e
08	ou

Entrada e Saída de Dados

Entrada: (Leitura de Dados)

Comando **Leia** (Identificador da variável, ...)

Ex: Leia(Nome, Salário, Idade)

Obs.: Esses nomes (entre parenteses) serão “**apelidos**” pelos quais os dados serão referenciados no algoritmo. Esses nomes serão explicados (por extenso) no Dicionário de Dados.

Ex.: Sal: salário de empregado (real)

Saída: (Impressão ou gravação de dados)

Comando **Imprima** (Identificador da variável, ...)

Imprima ('texto explicativo', Identificador, ...)

Obs.: Tanto leitura, como a gravação podem indicar de onde serão lidos ou onde serão gravados os dados.

Ex.: Leia(Nome, Idade do Arq.A)

Exemplo de Construção de Um Programa

CA [R.N.M.L.O]

Projeto

Dicionário de Dados

Nota1 : Nota da P1 (Real)

Nota2 : Nota da P2 (Real)

Nomealu: Nome do aluno (Texto)

Média : Média final do aluno (Real)

Algoritmo CalcMed

Início

Leia (NomeAlu,Nota1,Nota2)

$Média \leftarrow (Nota1 + Nota2) / 2$

Imprima (Nomealu,Média)

Fim CalcMed.

Algoritmo: Estrutura e Procedimento

Leia (NomeAlu,Nota1,Nota2)

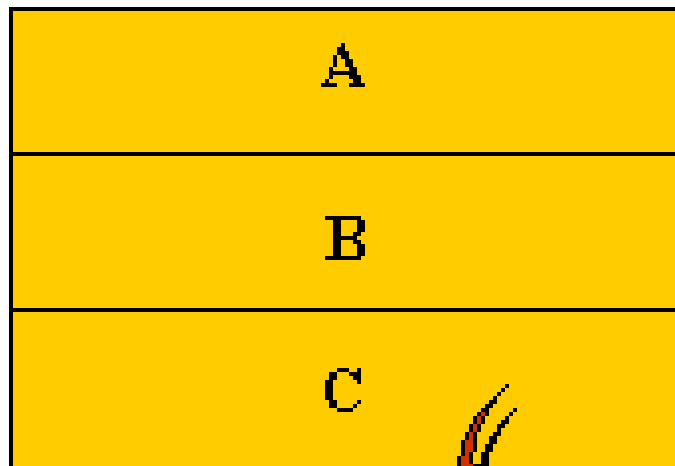
$Média \leftarrow (Nota1 + Nota2) / 2$

Imprima (Nomealu, Média)

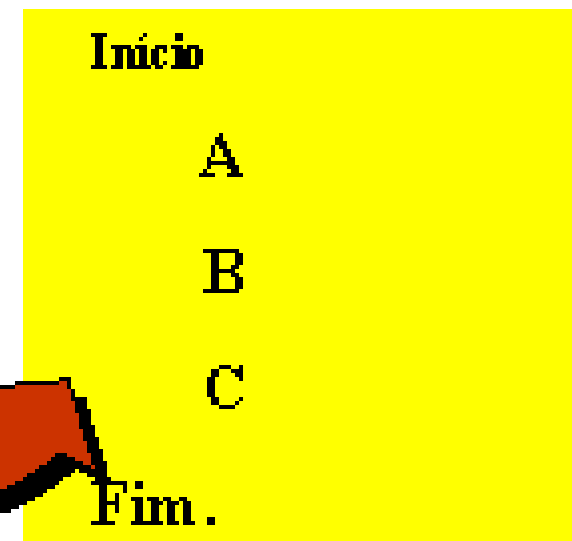
Sequência

É indicada pelas linhas de comando de cima para baixo

NS



PC

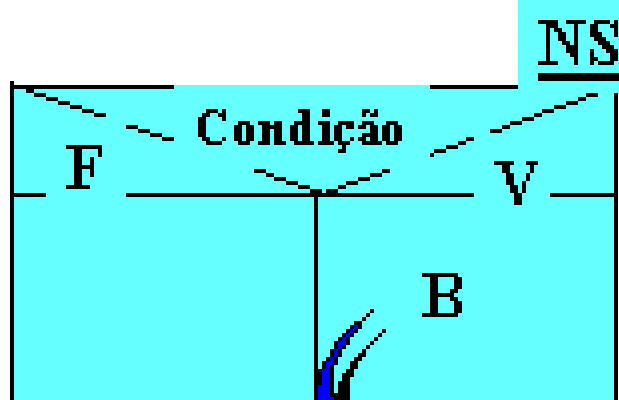


Seleção

Quando uma ação a ser executada depender de um teste, tem-se uma seleção, que pode ser Simple ou Composta.

PC

Seleção Simples



Início

.

.

Se Condição

Então B

Fim-Se

.

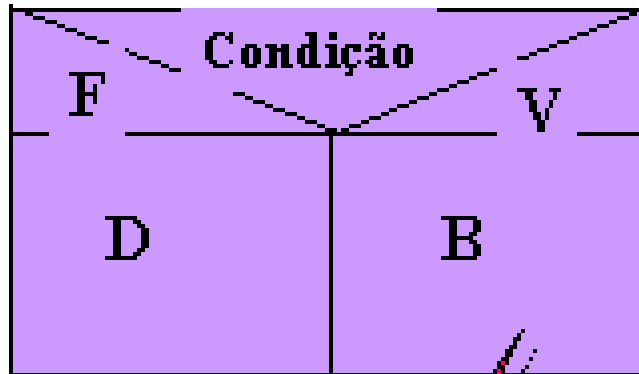
.

Fim

Seleção Composta

NS

PC



Início

.

.

Se Condição

Então B

Senão D

Fim-se

.

.

Fim

Repetição

Quando um conjunto de ações é executado repetidas vezes, tem-se uma repetição.

A estrutura de repetição, assim como a estrutura de seleção, envolve sempre a avaliação de uma condição

São três os tipos de estruturas de repetição:

Enquanto
Variando e
Repita até

● Enquanto

É usada para repetir n vezes uma ou mais instruções, tendo como vantagem o fato de não ser necessário o conhecimento prévio do valor de n (isto é, não é preciso saber-se o número de repetições)

PC

Início

.

.

Enquanto Condição Faça

B

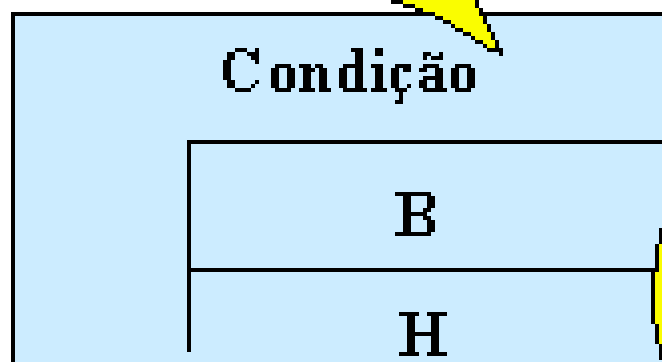
H

Fim-Enquanto

.

.

Fim.

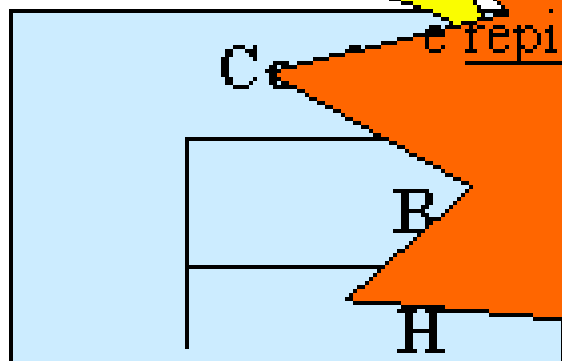


● Enquanto

É usada para repetir n vezes uma ou mais instruções, tendo como vantagem o fato de não ser necessário o conhecimento prévio do valor de n (isto é, não é preciso saber-se o número de repetições)

PC

○ *enquanto* é a estrutura mais geral, isto é, as estruturas variando e repita até podem ser representadas pelo *enquanto*.



Fim.

Problema:



Calcular a média final de um aluno. As informações conhecidas de cada aluno são: Nome, NotaP1 e NotaP2. Imprimir a média e os dados do aluno.

Obs.: A média final é dada por: $0.4 p1 + 0.6 p2$



Diagrama NS

Leia (Nome,NotaP1,NotaP2)

Mf \leftarrow (0.4*NotaP1 + 0.6*NotaP2)

Imprima (Nome,NotaP1,NotaP2,mf)

PC

Algoritmo EX1

Início

Leia (Nome, NotaP1,NotaP2)

mf \leftarrow (0.4*NotaP1 + 0.6*NotaP2)

Imprima (Nome, NotaP1,NotaP2,mf)

Fim.

Dicionário de Dados

Nome- nome do aluno :(texto)

NotaP1,NotaP2- notas de cada prova do aluno:(inteiro)

mf- média do aluno :(real)