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, ingles):apresenta um inconveniente: a ambiguidade de alguns termos.
- Em uma Linguagem de Programação:porém apresenta alguns incovenientes: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 determi-nada variável, onde o tipo desse valor tem de ser compa-tível ao tipo da variável.

Ex.: Soma ← 100

Operadores

Aritméticos

+ : Soma
- : Subtração

* : Multiplicação

/ : Divisão

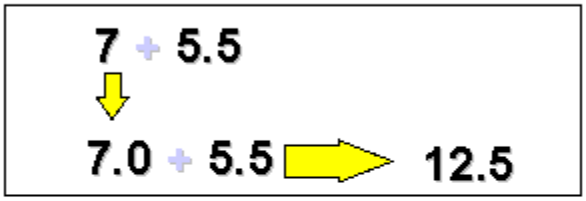
** : Potênciação

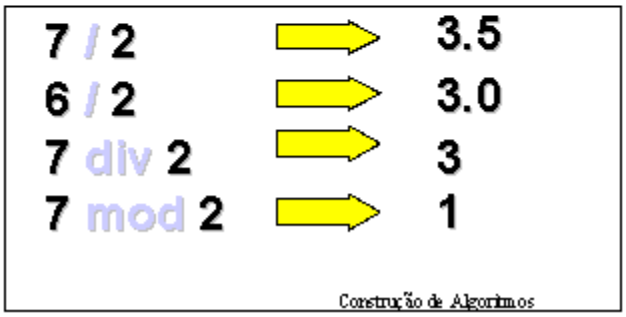
m mod i : resto da divisão de m por i

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

Operadores

Aritméticos - Exemplos





Ex.: 5/2 = 2,5

m MOD i (ou resto) : dá o resto da divisão inteira de m por i.

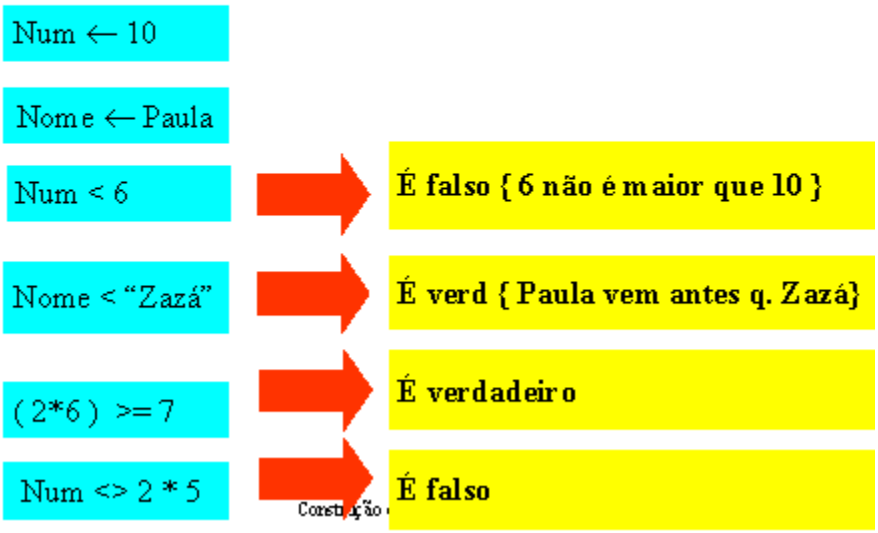
ex.: 10 mod 4 = 2
6 mod 2 = 0

m DIV n (ou quociente) : dá o quociente inteiro da divisão de m por n

ex.: 20 div 6 = 3

Operadores Relacionais

Exemplos:



Operadores Lógicos

Operando		Operando	Resultado
lógico	e ou não	lógico	lógico

Exemplos

Achou ← falso

Num ← 9

(1 > 9) e (20 > 3) → **F e V = F**

Não Achou → **V**

(4 >= 5) ou (5 >= 3) → **F ou V = V**

(Num < 3) ou (Num < 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 gravação podem indicar de onde serão lidos ou onde serão gravados os dados.

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

Exemplo de Contrução de Um Programa

[P.N.M.A.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 ← (Nota1+ Nota2)/2
Imprima (Nomealu,Média)
Fim CalcMed.

Algoritmo: Estrutura e Procedimento

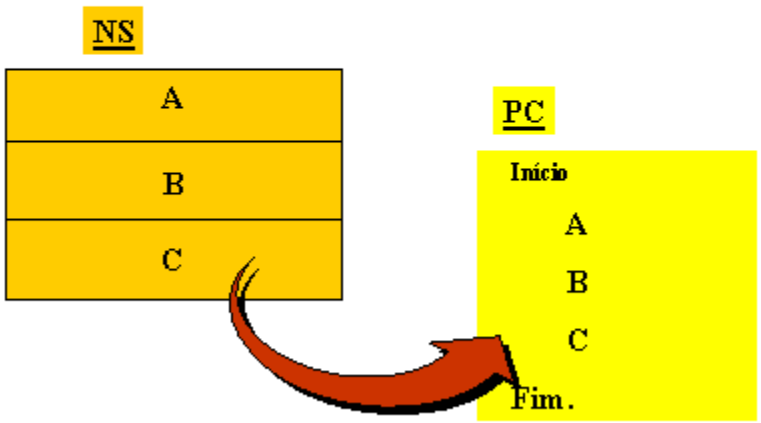
Leia (NomeAlu,Nota1,Nota2)

Média ← (Nota1 + Nota2) / 2

Imprima (Nomealu, Média)

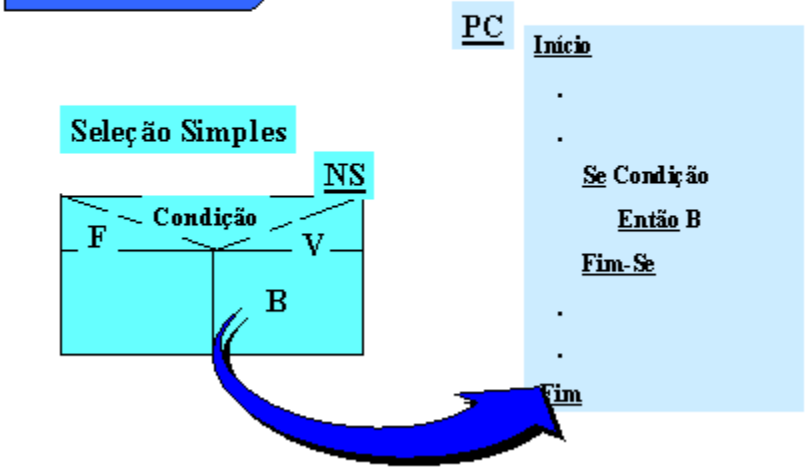
Sequência

É indicada pelas linhas de comando de cima para baixo



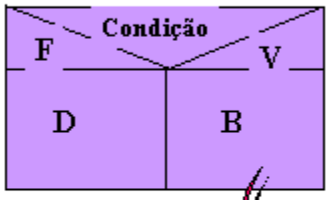
Seleção

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



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.

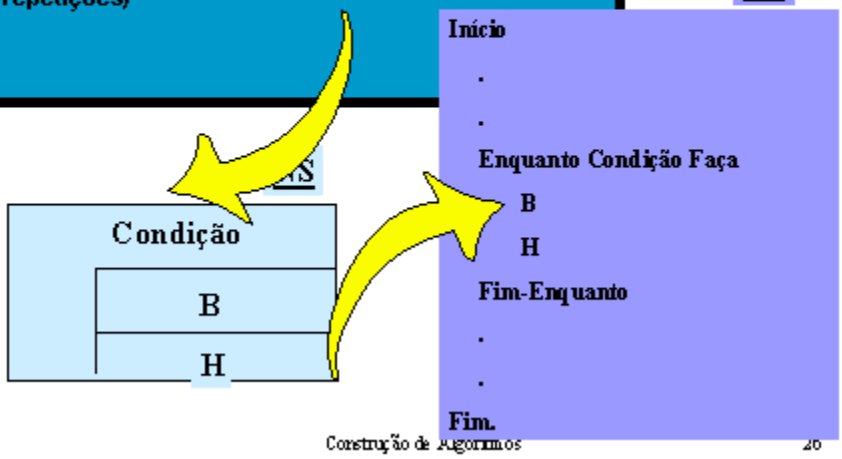
A estrutura de repetição, assim como a estrutura de seleção,envolve sempre a avaliação de uma condição

São três os tipos de estruturas de repetição:

- **Enquanto**
- **Variando e**
- **Repita até**

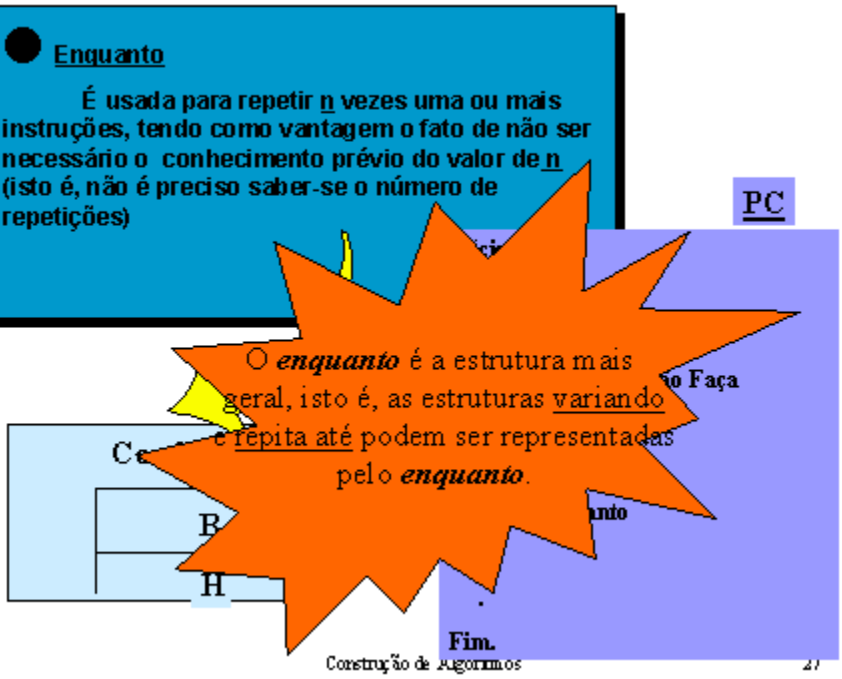
Enquanto

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)



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)



Problema:

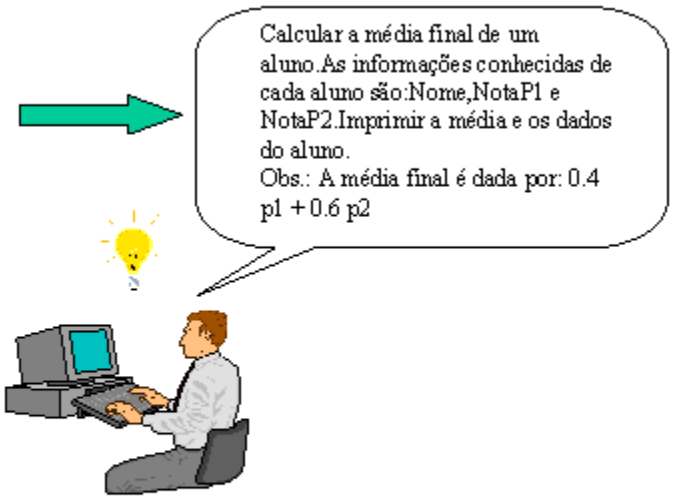


Diagrama NS

Leia (Nome,NotaP1,NotaP2)
Mf ← (0.4*NotaP1 + 0.6*NotaP2)
Imprima (Nome,Notap1,NotaP2,mf)

PC

```
Algoritmo EX1
Início
   Leia (Nome, NotaP1,NotaP2)
   mf ← (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)