Estruturas de Controle

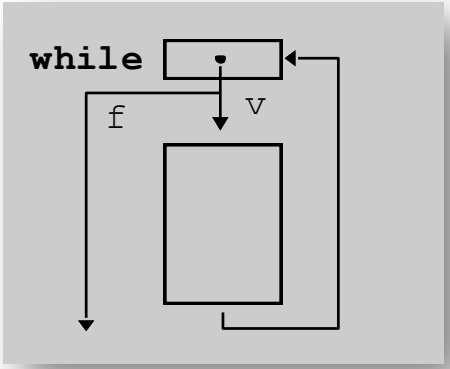
Introdução à Ciência da Computação I

Estruturas de Controle

- ESTRUTURA SEQUENCIAL
- ESTRUTURAS CONDICIONAIS
  - Estrutura Condicional Simples
  - Estrutura Condicional Composta
  - Seleção entre duas ou mais Seqüências de Comandos
- **ESTRUTURA DE REPETIÇÃO**
  - **Repetição com Teste no Início**
  - **Repetição com Teste no Final**
  - **Repetição Contada**

O Comando While

```
while (condição) {
    comandos;
}
```

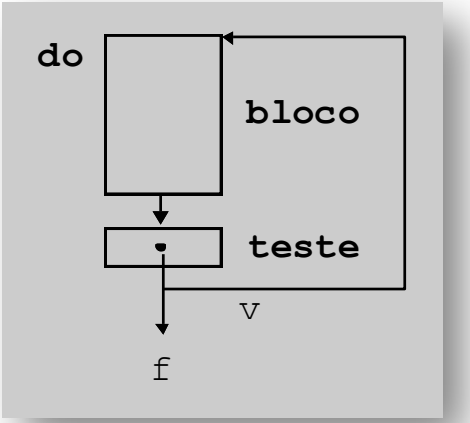


- 1º avalia condição
- se condição é verdadeira, executa comandos do bloco
- ao término do bloco, volta a avaliar condição
- repete o processo até que condição seja falsa

O Comando Do-While

- *do...while* é utilizado sempre que o bloco de comandos deve ser executado ao menos uma vez

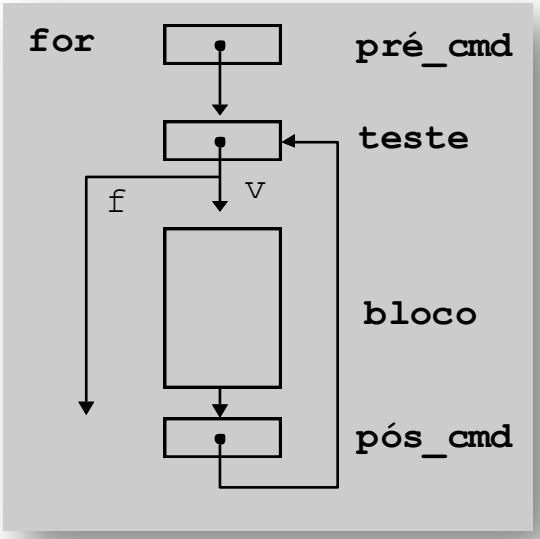
```
do {
 comandos;
} while (condição);
```



- 1º executa comandos
- 2º avalia condição:
  - se verdadeiro, executa novamente os comandos do bloco; senão encerra laço

O Comando For

```
for (pré_cmd; teste; pós_cmd) {
  comandos;
}
```



- em termos de while, equivale a:

```
pré_cmd;
while (teste) {
  comandos;
  pós_cmd;
}
```

O Comando For

- 1º executa *pré_cmd* (*inicialização*), que permite iniciar variáveis
- 2º avalia *teste* (*condição*): se verdadeiro, executa comandos do bloco, senão encerra laço
- ao término do bloco, executa *pós_cmd* (*incremento*)
- reavalia teste
- repete o processo até que teste seja falso

O Comando For

- O loop for é usado para repetir um comando, ou bloco de comandos, diversas vezes, de maneira que se possa ter um bom controle sobre o loop.

```
for (inicialização; condição; incremento) {
    seqüência_de_comandos;
}
```

O Comando For

- Exemplo: Escreve a tabuada do 2 na tela

```
#include <stdio.h>
int main( ) {
      int i;
      for ( i=1; i <=10; i++)
         printf("2 x %d = %d\n", i, 2*i);
      return 0;
}
```

O Comando For

- Podemos omitir qualquer um dos elementos (inicialização, condição ou incremento) do for.
- Por exemplo:

```
for (inicialização; ;incremento) {
  seqüência de comandos;
}
```

  - Este é um loop infinito porque será executado para sempre (não existindo a condição, ela será sempre considerada verdadeira), a não ser que ele seja interrompido.
  - Para interromper um loop como este usamos o comando break.

Exercícios

1) Ler x e y e calcular x^y sem usar funções de potência da linguagem C.

2) Faça um programa para calcular o valor da seguinte série:

$$S = \frac{1}{1} + \frac{3}{2} + \frac{5}{3} + \frac{7}{4} + \ldots + \frac{99}{50}$$

Exercícios

3) Crie um programa que receba um valor positivo inteiro e imprime os divisores desse valor.

Exercícios

3) Crie um programa que receba um valor positivo inteiro e imprime os divisores desse valor.

4) Criar um programa em C que determina os números de 1 a 5000 que são quadrados perfeitos (a raiz é um número inteiro). Não é permitido o uso de comandos do C para potência e raiz.

Exercícios

1) Calcule o máximo divisor comum (m.d.c.) e o mínimo múltiplo comum (m.m.c.) de 3 números fornecidos pelo usuário..

Exercícios

5) Desenvolver um programa para calcular o somatório dado abaixo, sabendo que N e P devem ser informado pelo usuário.

$$S = \sum_{i=1}^{N} \sqrt{\frac{P+i}{i}}$$

Exercícios

6) Crie uma função que recebe um inteiro e exibe uma “seta”. Obs: A linha central da seta deverá ter tamanho n.

   Ex. para n = 3:

```
*

*  *

*  *  *

*  *

*
```

Exercícios

1) Faça um programa que leia um valor inteiro e calcule o seu fatorial.

2) Calcular e escrever o valor do número π , com n termos, usando a série π= 4 – 4/3 + 4/5 - 4/7 + 4/9 - 4/11...

Exercícios

3) Criar uma função que receba como argumento um número real e um número inteiro e retorne a raiz quadrada do número real através de Newton:

$R_{n+1} = (R_n + (E/R_n))/2$

e $R_1 = E/2$

para E = entrada, R = raiz quadrada

e n = número de iterações