Linguagem C

Revisão: Comandos de Repetição

Baseado em slides do Prof. Bruno Travençolo

Estruturas de Repetição

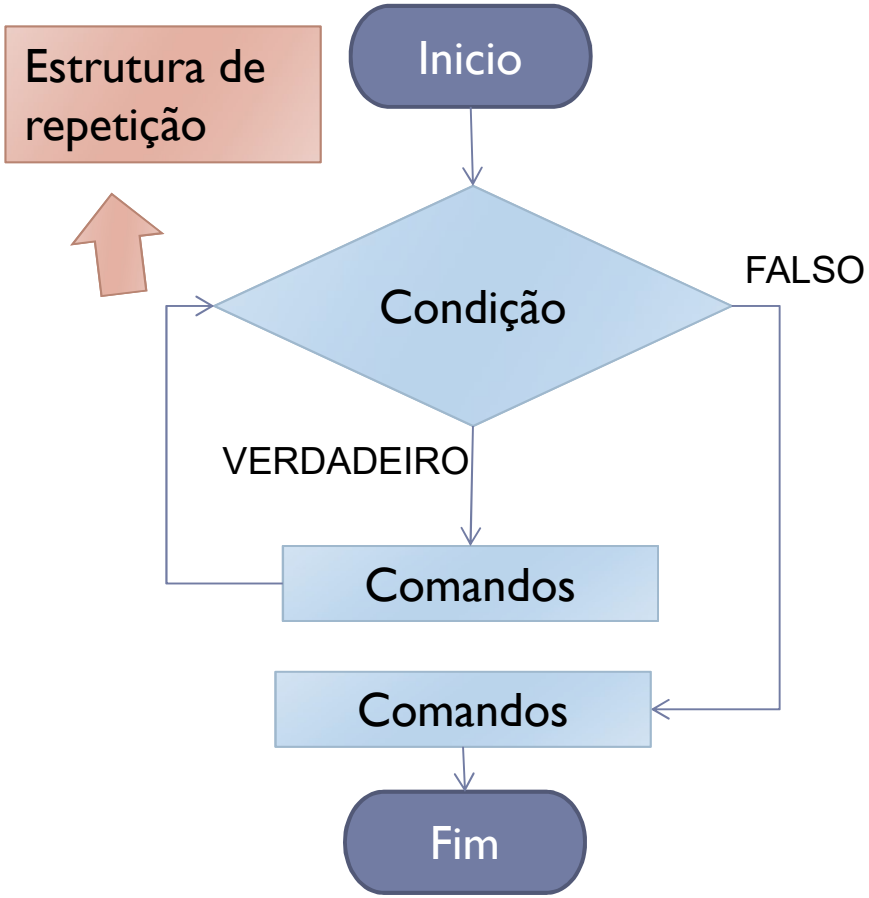
Uma **estrutura de repetição** permite que uma sequência de comandos seja executada **repetidamente**, **enquanto** determinadas condições são satisfeitas. Essas condições são representadas por expressões lógica (como, por exemplo, A>B; C==3; Letra == 'a')

Estruturas de Repetição

- **Repetição com Teste no Início**
- Repetição com Teste no Final
- Repetição Contada

Fluxograma

Repetição com Teste no Início



Comando while

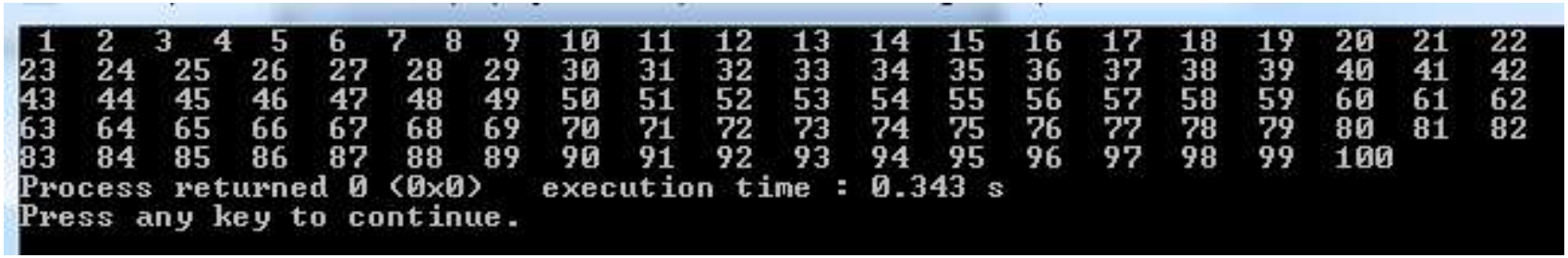
- Repete a seqüência de comandos enquanto a condição for verdadeira.
- Esse comando possui a seguinte forma geral:

```
while (expressão)
    instruções
```

Exemplo 1

- Faça um programa que mostra na tela os número de 1 a 100

- Saída exemplo:



Exemplo 1

- Faça um programa que mostra na tela os número de 1 a 100

```c
int main()
{
    // programa que mostra na tela números de 1 ate 100
    int numero;

    numero = 1;
    while (numero <= 100){
        printf(" %d ", numero);
        numero = numero + 1;
    }

    return 0;
}
```

Exemplo 2

- Faça um programa para ler 5 números e mostrar o resultado da soma desses números.
- Faça o programa para ler **n** números e mostrar o resultado da soma desses números, onde **n** é uma variável parametrizada.

Exemplo 2

```c
int main()
{
    double val, soma;
    int contador, num=5;

    // inicializando o valor de soma
    soma = 0;

    // inicializando o contador
    contador = 1;

    while (contador <= num){
        printf("\nDigite o %do. numero: ", contador);
        scanf("%lf", &val);
        soma = soma + val;
        contador = contador + 1;
    }

    printf("\nO resultado da soma eh: %.2f", soma);
    return 0;
}
```

Exemplo 3

```
int main(){

  int a,b;

  printf("Digite o valor de a: ");
  scanf("%d",&a);

  printf("Digite o valor de b: ");
  scanf("%d",&b);

  while (a < b) {
        a = a + 1;
        printf("%d \n",a);
  }

}
```

Neste exemplo não há um número determinado de vezes que o loop é executado, como nos casos que usamos contadores explícitos. Quem define isso são os valores de a e b

Exercício

Escreva um programa que dados **n** números inteiros, calcule a media destes números. O valor de **n** é dado pelo usuário. Imprima os números lidos e a média calculada:

$$m = \frac{\sum x}{N}$$

Exercício – em C

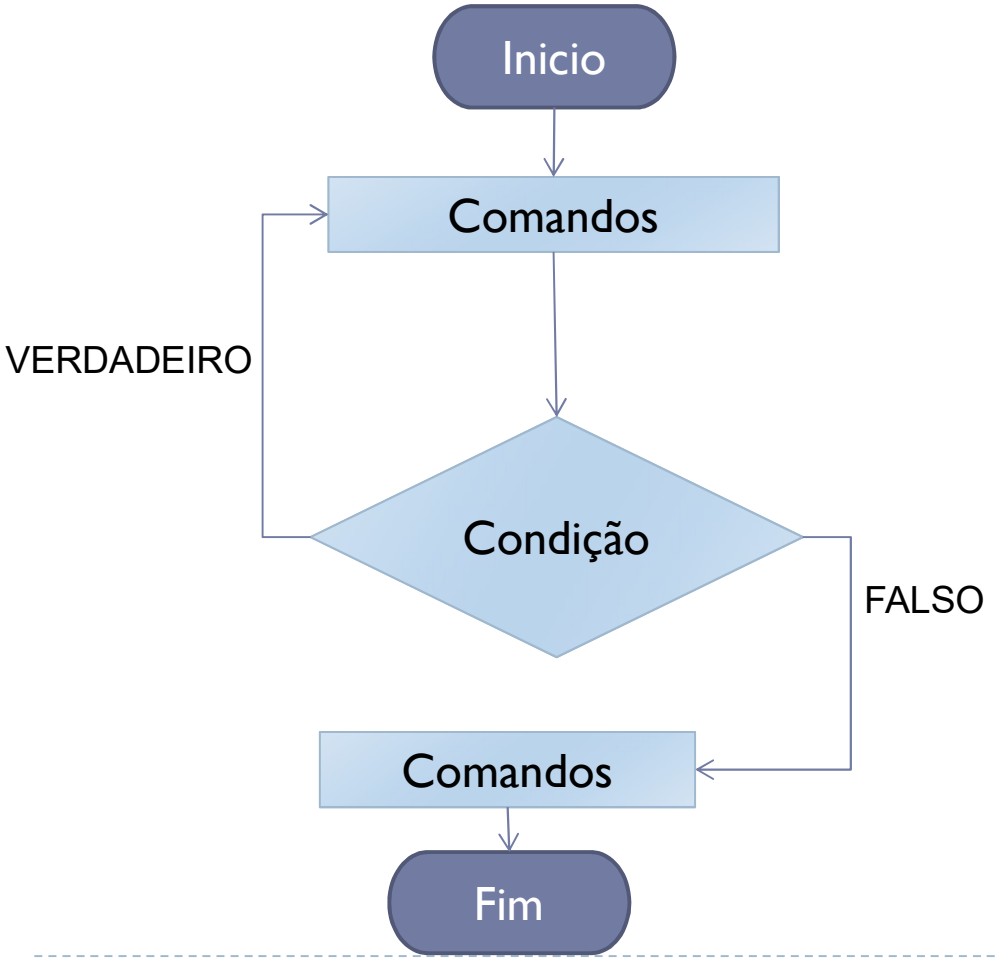
```c
int main()
{
   int n, contador;
   float acumulador, nota, media;

   printf("Quantos números vc deseja digitar: ");
   scanf("%d", &n);

   acumulador = 0;
   contador = 0;

   while (contador < n) {
        printf("Digite a nota %d:", contador+1);
        scanf("%f", &nota);
        acumulador = acumulador + nota;
        contador = contador + 1;
   }

   media = acumulador/n;

   printf("O valor da média é: %f", media);

   return 0;
}
```

Estruturas de Repetição

- Repetição com Teste no Início
- **Repetição com Teste no Final**
- Repetição Contada

Fluxograma

Repetição com Teste no Final



Comando do-while

- Comando while: é utilizado para repetir um conjunto de comandos zero ou mais vezes.
- Comando do-while: é utilizado sempre que o bloco de comandos **deve ser executado ao menos uma vez.**

Comando do-while

- executa comandos
- avalia condição:
  - se verdadeiro, re-executa bloco de comandos
  - caso contrário, termina o laço

- Sua forma geral é (**sempre termina com ponto e vírgula ;**)

```
do
    instruções
while (expressão);
```

Comando do-while

```c
#include <stdio.h>
#include <stdlib.h>

int main()
{
    int i;
    do{
        printf("Escolha uma opcao:\n");
        printf("(1) Opcao 1 \n");
        printf("(2) Opcao 2 \n");
        printf("(3) Opcao 3 \n");
        scanf("%d", &i);
    } while ((i<1)||(i>3));
    printf("Opcao escolhida: %d \n",i);
    return 0;
}
```

Exercício

Escreva um programa chamado "ACUMULE", que conta e mostra na tela o número de vezes que o número "7" é digitado pelo usuário. O programa lê vários números inteiros até que o número -1 seja digitado.

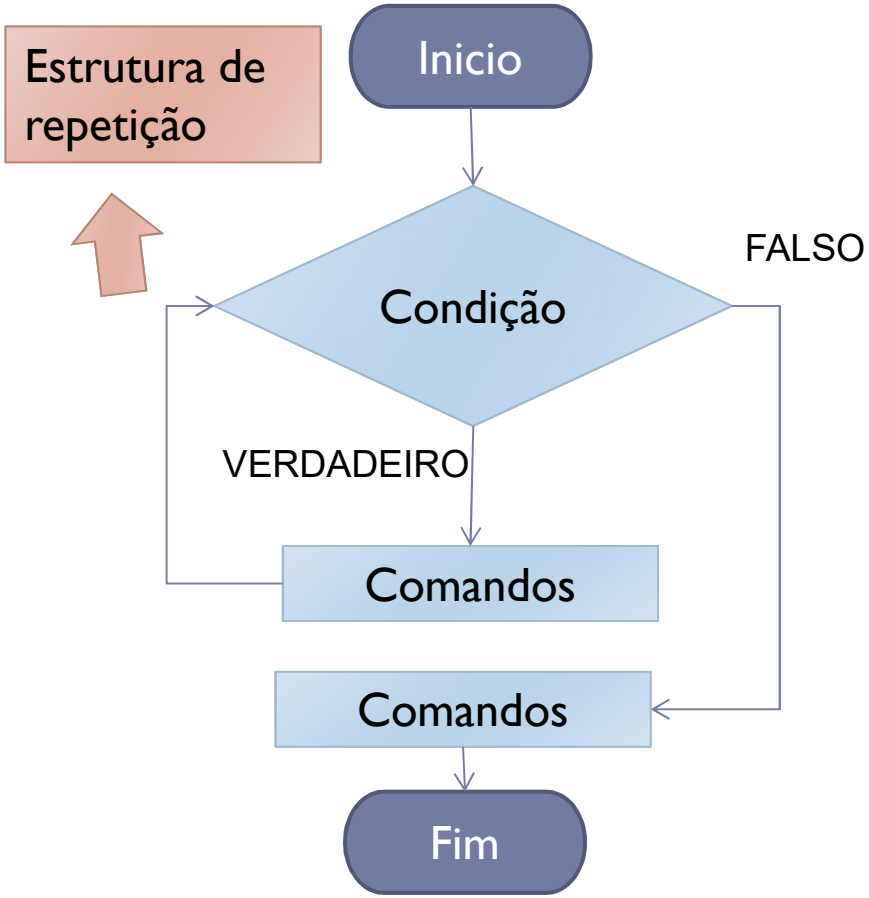
```c
#include <stdio.h>
#include <stdlib.h>

int main()
{
    int acumulador; // guarda a qte que o numero 7 eh digitado
    int x; // leitura do dado

    // inicializando acumulador
    acumulador = 0;

    do{
       printf("\n Digite um valor: ");
       scanf("%d",&x);

       if (x == 7)
        acumulador = acumulador + 1;

    } while (x != -1);

    printf("vc digitou %d vezes o num. 7\n", acumulador);
    return 0;
}
```

Estruturas de Repetição

- Repetição com Teste no Início
- Repetição com Teste no Final
- **Repetição Contada**

Fluxograma

Repetição Contada



Comando **para** (for)

- Para utilizar o comando para é preciso ter
  - Uma variável para realizar a contagem. Exemplos: 1

    contador : inteiro

    i : inteiro
  - Inicializar a variável contador com um valor. 2
  - Especificar uma condição para continuar no loop 3
  - Incrementar a variável contador 4

1 2 3 4

para <*variável*> ***de*** <*valor-inicial*> ***ate*** <*valor-limite*> [***passo*** <*incremento*>]
faca
    <*seqüência-de-comandos*>
fimpara

Comando *for* (para) em C

- O loop ou laço *for* é usado para repetir um comando, ou bloco de comandos, diversas vezes
  - Maior controle sobre o loop.
- Sua forma geral em C é um pouco distinta do pseudocódigo, mas possui os mesmo elementos

```
for (inicialização; condição; incremento)
        instruções
```

Comando for

```
for (expression1; expression2; expression3)
        statement
```

1. `expression1`: inicialização: iniciar variáveis (contador).

2. `expression2`: condição: avalia a condição. Se verdadeiro, executa comandos do bloco, senão encerra laço.

3. `expression3`: incremento: ao término do bloco de comandos, incrementa o valor do contador

4. repete o processo até que a condição (`expression2`) seja falsa.

Exemplo for

```c
int main()
{
  // mostra os valores de 1 até 100
  int i;
  for (i = 1; i <= 100; i = i+1){
    printf("%d ",i);
  }
}
```

Operador Incremento

- Em C, existe um operador de incremento cujo símbolo é ++
- Ele serve para incrementar em uma unidade o valor de uma variável.
  - Exemplo i++ tem o mesmo efeito que i = i + 1
- Esse operador é muito comum em loops *for*

```
int main()
{
  // mostra os valores de 1 até 100
  int i;

  for (i = 1; i <= 100; i++){
    printf("%d ",i);
  }
}
```

Exercício

- Escreva, usando for, um algoritmo para calcular a soma dos números inteiros de 1 a 10.

Exercício

```
int n;
int soma = 0;
for (n = 1; n <= 10; n++){
      soma = soma + n;
}
printf("%d",soma);
```

Exemplo for

```
01  #include <stdio.h>
02  #include <stdlib.h>
03  int main(){
04    int a,b,c;
05    printf("Digite o valor de a: ");
06    scanf("%d",&a);
07    printf("Digite o valor de b: ");
08    scanf("%d",&b);
09    for (c = a; c <= b; c++){
10       printf("%d \n",c);
11    }
12    system("pause");
13    return 0;
14  }
```

Comando for

- Podemos omitir qualquer um de seus elementos
  - inicialização, condição ou incremento.
- Ex.: **for** sem inicialização

```
01  #include <stdio.h>
02  #include <stdlib.h>
03  int main(){
04    int a,b,c;
05    printf("Digite o valor de a: ");
06    scanf("%d",&a);
07    printf("Digite o valor de b: ");
08    scanf("%d",&b);
09    for (; a <= b; a++){
10       printf("%d \n",a);
11    }
12    system("pause");
13    return 0;
14  }
```

Comando for

- Cuidado: for sem condição
  - omitir a condição cria um laço infinito;
  - condição será sempre verdadeira.

- Cuidado: for sem incremento
  - omitir o incremento pode criar um laço infinito;
  - Incremento pode ser feito nos comandos.