Estruturas de repetição

Definição

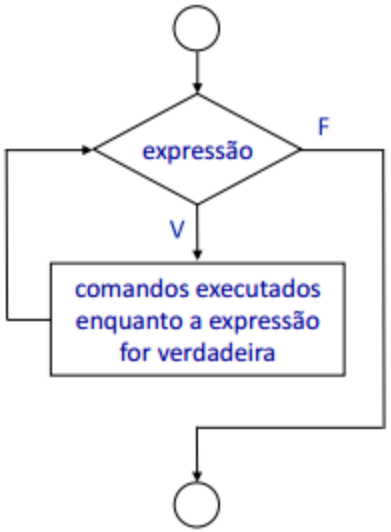
- Uma estrutura de repetição é utilizada quando um trecho de um algoritmo precisa ser executado diversas vezes.
- A cada execução do trecho, algumas variáveis terão seus valores modificados por algum cálculo ou alguma seqüência de instruções.
- O número de repetições pode ser fixo ou pode depender do valor de uma condição.
- Se o número de repetições for atingido ou se a condição de execução for falsa, o fluxo de execução é desviado para o primeiro comando após a estrutura de repetição.

Estruturas de repetição

- Pré teste

Tipos de estruturas de repetição

Teste no início



while()

Estrutura de repetição com teste no início.

Sintaxe:

```
while (expressão)
{
// comandos executados enquanto
// a expressão for verdadeira
}
```

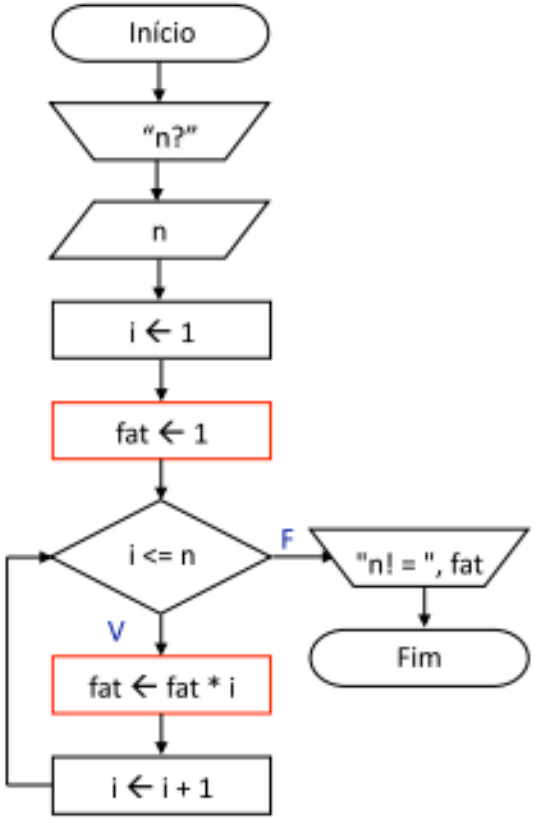
Atenção

Não se usa "**;**" após o **while()**.

Estruturas de repetição

- Pré teste

Escrever um programa para calcular n! = 1 × 2 × 3 × ... × n.



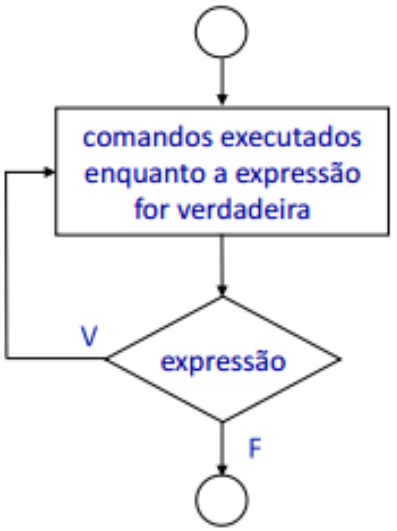

```c
#include <stdio.h>
#include <stdlib.h>

int main()
{
  int i, n;
  double fat;

  printf("Apresentar o fatorial de: ");
  scanf("%d", &n);
  i = 1;
  fat = 1;
  while ( i <= n )
  {
    fat = fat * i;
    i++;
  }
  printf("%d! = %.0f\n", n, fat);
  system("PAUSE");
  return(0);
}
```

Estruturas de repetição

- Pós teste

Tipos de estruturas de repetição

Teste no final



do-while()

Estrutura de repetição com teste no final.

Sintaxe:

```
do
{
// comandos executados enquanto
// a expressão for verdadeira
} while (expressão);
```

Atenção

Não se usa "**;**" após o **do**.

Estruturas de repetição

- Pós teste

Repetições com intervenção do usuário

```
int main()
{
  int num;
  char op;

  do
  {
    system("CLS");    // limpa a tela de execução

    printf("Informe um numero inteiro: ");
    scanf("%d", &num);

    if (num % 2 == 0)
      printf("%d é par\n", num);
    else
      printf("%d é ímpar\n", num);

    printf("Deseja executar novamente (S/N)? ");
    op = getchar();
  } while (op == 's' || op == 'S');

  system("PAUSE");
  return(0);
}
```

Repetição e seleção

Determinar o maior de um conjunto de 10 valores informados pelo usuário.

```
int main()
{
  float maior, valor;
  int cont;

  maior = -999999;
  cont = 0;
  while (cont < 10)
  {
    printf("Valor: ");
    scanf("%f", &valor);
    if (valor > maior)
      maior = valor;
    cont++;
  }
  printf("Maior valor informado: %f\n", maior);
  system("PAUSE");
  return(0);
}
```

Repetição dentro de repetição

Imprimir a tabuada do 1 ao 10.

```
int main()
{
  int a, b, valor;

  a = 1;
  while (a <= 10)
  {
    printf("\nTabuada do %d\n", a);
    b = 1;
    while (b <= 10)
    {
      valor = a * b;
      printf("%2d x %2d = %3d\n", a, b, valor);
      b++;
    }
    a++;
  }
  system("PAUSE");
  return(0);
}
```

Alguns Exercícios

O que será feito?

```
int main()
{
  float soma, termo;
  int cont, n;

  printf("Número de termos: ");
  scanf("%d", &n);
  cont = 0;
  soma = 0;
  termo = 0;

  do {
    cont++;
    soma = soma + termo;
    printf("Novo termo: ");
    scanf("%f", &termo);
  } while (condição);

  printf("Soma: %f (%d termos)\n", soma, cont);
  system("PAUSE");
  return(0);
}
```

Se condição for:

soma < 1000

Serão somados termos enquanto a soma não ultrapassar o valor 1000.

termo > 0

A soma considera apenas termos positivos.

cont < n

Serão somados exatamente n termos.

O que será impresso na linha 16

```
1  #include "stdio.h"
2  #include "stdlib.h"
3
4  main(){
5    int a = 1 , b = 2;
6
7    while(a < 16){
8        a += b;
9
10       do{
11         b += a;
12         a++;
13       }while(b < 9);
14   }
15
16   printf("a = %d, b = %d",a,b);
17   system("pause");
18 }
```

Controle do número de repetições

- A cada execução dos comandos de uma estrutura de repetição dá-se o nome de iteração.
- Em alguns problemas o número de iterações é fixo ou determinado em tempo de execução. Nestes casos, usa-se uma variável de controle do número de iterações.
- A variável de controle deve ser inicializada antes do primeiro comando da estrutura de repetição.
- Dentro da estrutura, a variável de controle terá seu valor atualizado de acordo com algum incremento ou decremento.
- A variável de controle será usada na expressão que determina se haverá a execução de uma nova iteração.

Controle no número de iterações

Exibir os *n* primeiros números ímpares positivos.

```
int main()
{
  int num, n, cont;

  printf("Quantos números ímpares deseja exibir? ");
  scanf("%d", &n);
  cont = 1;
  num = 1;

  while ( cont <= n )
  {
    printf("%d ", num);
    num = num + 2;
    cont++;
  }

  system("PAUSE");
  return(0);
}
```

variável de controle

inicialização

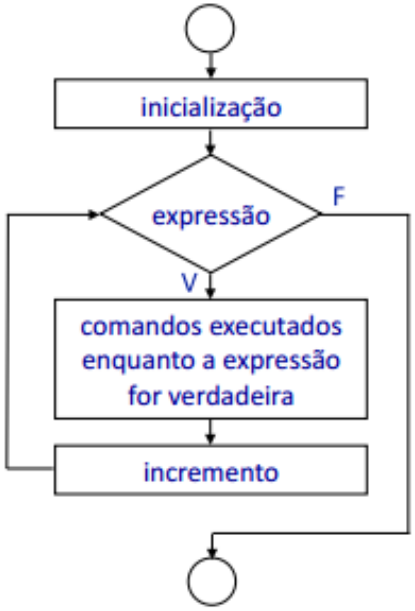
expressão

incremento

- Pré teste

Tipos de estruturas de repetição

Controle no número de repetições



for()

Estrutura de repetição com controle do número de repetições.

Sintaxe:

```
for(inicialização; expressão; incremento)
{
// comandos executados enquanto
// a expressão for verdadeira
}
```

Atenção

Não se usa "**;**" após o **for()**.

- Pré teste

Exemplo: estrutura de repetição `for( )`

Exibir os *n* primeiros números ímpares positivos.

```
int main()
{
  int num, n, cont;

  printf("Quantos números ímpares deseja exibir? ");
  scanf("%d", &n);
  num = 1;

  for ( cont = 1; cont <= n; cont++ )
  {
    printf("%d ", num);
    num = num + 2;
  }

  system("PAUSE");
  return(0);
}
```

variável de controle

inicialização

incremento

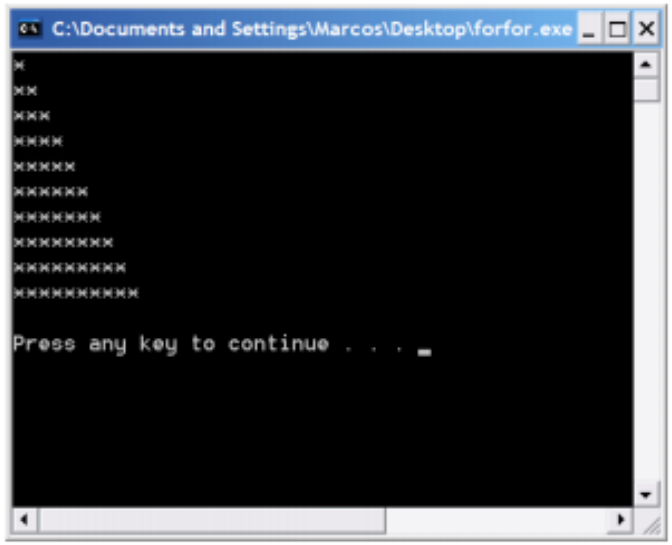
expressão

- Pré teste

Aninhamento: `for( )` dentro de `for( )` dentro de...

No aninhamento, todas as iterações do laço mais interno serão executadas antes da próxima iteração do laço mais externo.

```
int main()
{
  int i, j;

  for ( i = 1; i <= 10; i++ )
  {
    for ( j = 0; j < i; j++ )
    {
      printf("*");
    }
    printf("\n");
  }

  system("PAUSE");
  return(0);
}
```



Neste exemplo, a condição de execução do laço interno depende do valor da variável de controle do laço externo.

O laço infinito

Uma estrutura de repetição pode apresentar execução infinita se:

- A condição de execução for mal formulada: a expressão realiza comparações exatas (== ou !=) ou envolve variáveis de controle cujo valor não se altera.
- A atualização da variável de controle não produz os valores esperados, devido às limitações na representação de números reais pelo computador.

```
float x, y;

x = 1.0;
y = 0.1;  // atenção: este valor não admite representação exata!
while( x != 0.0 )
{
  x = x - y;
  printf("%.1f\n", x);
}
```

O operador ? : (ternário)

- É uma simplificação do comando if-else, ou seja, faz uma atribuição condicional com apenas um comando e não 2 ou mais blocos de comandos:
  - A forma geral do operador ? : é:
  - var = (expressão condicional **?** expressão1 **:** expressão2);
  - O funcionamento do operador ? é idêntico ao do comando if-else:

    primeiramente, a expressão condicional será avaliada e:
    - Se essa condição for verdadeira, o valor da expressão1 será o resultado da expressão condicional.
    - Se essa condição for falsa, o valor da expressão2 será o resultado da expressão condicional.
  - Ex: int maior = (a>b ? a : b);

Comandos de controle de fluxo

Permitem alterar o fluxo de execução de uma estrutura de repetição.

`break;`

Ignora os comandos restantes do bloco, desviando imediatamente o fluxo de execução do programa para o primeiro comando após o final do bloco.

`continue;`

Ignora os comandos restantes do bloco e força a próxima iteração do laço, dependendo do valor da expressão de controle.

`exit(0);`

Encerra a execução do programa.

- Ex: break

Comandos de controle de fluxo: `break;`

Exemplo: Adivinhe o número que estou pensando...

```c
#include <time.h>

int main()
{
  int cont = 0, num, chute;

  srand( time(0) );           // inicializa o gerador de números aleatórios
  num = 1 + rand() % 100;     // num assume um valor inteiro entre 1 e 100;

  while ( 1 )                 // laço infinito
  {
    cont++;
    system("CLS");
    printf("Advinhe o numero que estou pensando... [1, 100]: ");
    scanf("%d", &chute);
    if (chute == num)
      break;
  }
  printf("Acertou em %d tentativas\n", cont);
  system("PAUSE");
  return(0);
}
```

Exercício 3 – Ling. C (Repetição)

1. Elabore um algoritmo em que o usuário entre com um número inteiro qualquer, e o software imprima os 20 números subsequentes ao que foi digitado pelo usuário.

2. Elabore um algoritmo que solicite que o usuário entre com dois números (inicial e final). Ao final o algoritmo deverá apresentar o valor total da soma de todos os números do intervalo digitado pelo usuário.

3. Elabore um algoritmo que apresente os números pares maiores que 10 no intervalo fechado [A, B]. Sendo que A e B serão números inteiros escolhidos pelo usuário. Um número é par quando este satisfaz a seguinte condição: (NÚMERO mod 2 = 0)

4. Escreva um algoritmo que solicite que o usuário entre com valores inteiros quaisquer. Ao final imprima a quantidade de números digitados, o somatório dos valores digitados, e a média aritmética do somatório. O valor “0” (zero) indica parar de ler valores

5. Elabore um algoritmo para fazer cálculo de potenciação. Ou seja, x^y. (Exemplo: 3^4 = 3 x 3 x 3 x 3). Seu algoritmo deverá solicitar que o usuário entre com o valor da base (x) e do expoente (y) e apresentar o resultado do cálculo sem utilizar os operadores ** ou ^. Para resolver o problema utilize estrutura de repetição.

Exercício 3 – Ling. C (Repetição)

6. Suponha que seu computador consiga executar somente operações de soma e subtração. Escreva programas em C que, dados dois números inteiros positivos a e b, calculem:

- O produto a * b.
- O quociente e o resto da divisão de a por b.
- A potência a ^ b (ab): Dica use o primeiro programa (a * b) para fazer a potência.

FIM