Linguagem de Programação

Comandos de Repetição

ECT2303

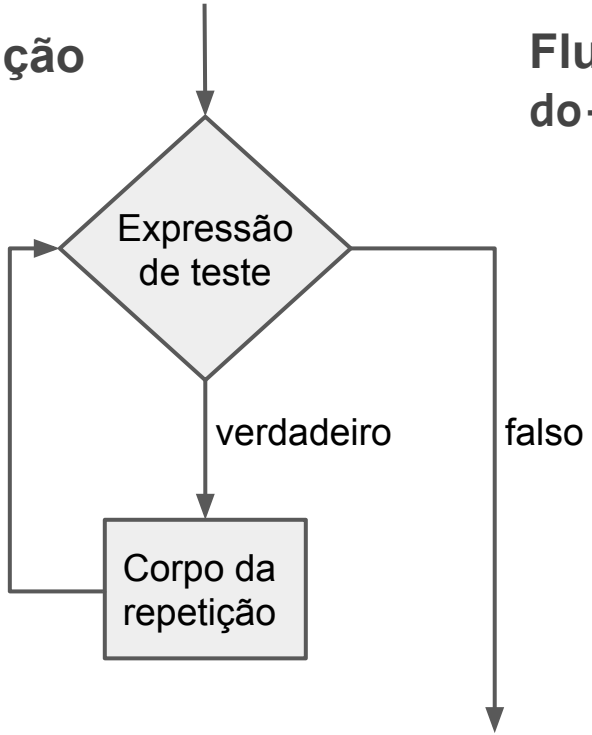
helton.maia@ect.ufrn.br

Comandos de repetição: `while` e `do-while`

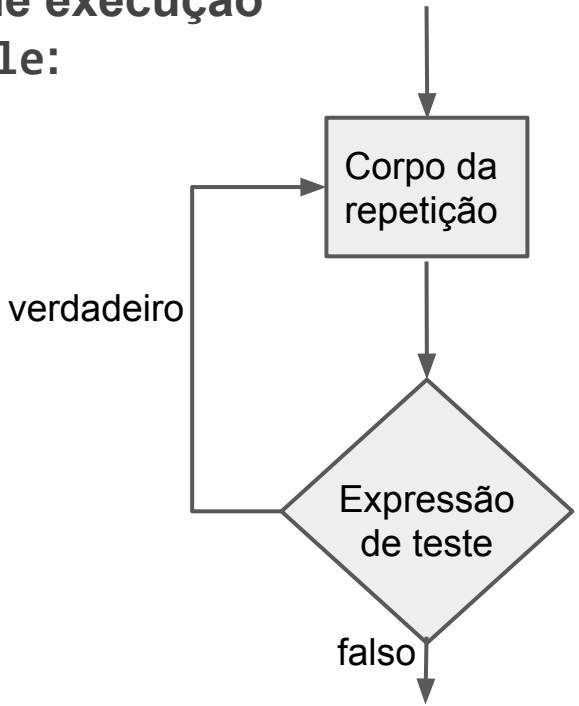
- Problemas complexos podem ser resolvidos utilizando comandos de repetição;
- A estrutura de repetição, garante que o bloco de instruções seja executado uma ou mais vezes, já que a condição que controla o laço(*loop*) é testada a cada repetição;
- O comando de repetição **while** possui duas partes: a expressão de teste e o corpo da repetição que contém os blocos de instruções a serem executados.
- Uma estrutura de repetição permite que seja especificado uma ação na qual deverá ser repetida **enquanto/até** que uma condição seja verdadeira.

Comandos de repetição: `while` e `do-while`

Fluxo de execução `while`:



Fluxo de execução `do-while`:



Comando de repetição `while` (enquato-faça)

while (*expressão teste*)
corpo da repetição

Exemplo:

Enquanto (houver itens na minha lista de compras)
……comprar próximo item
Fim Enquanto

Comando `while`

```
while(condição de teste) {
        comandos / blocos de comandos;
}
```

Se a condição de teste for verdadeira, ou seja, diferente de zero, o comando ou blocos de comandos será executado.

Exemplo:

```
#include <iostream>
using namespace std;

int main () {
   // Declarando variável
   int n = 1;

   // Executando o while loop
   while( n < 10 ) {
      cout << "valor de n: " << n << endl;
      n++;
   }
   return 0;
}
```

Saída: ?

Comando do-while

```
do {
   comandos / blocos de comando;
}
while(condição de teste);
```

Se a condição de teste (realizado no fim) for verdadeira, ou seja, diferente de zero, o comando ou blocos de comandos será executado.

Exemplo:

```
#include <iostream>
using namespace std;

int main () {
   // Declarando variável
   int n = 1;

   // do loop
    do {
      cout << "valor n: " << n << endl;
      n = n + 1;
   } while( n < 10 );
return 0;
}
```

Saída: ?

Observação:

O comando *break* quando inserido em uma estrutura de repetição, causa a saída imediata desta estrutura. A execução do programa é então direcionada para a primeira instrução após o final do comando/bloco de código do laço(*loop*) de repetição.

Exemplo:

```
int main(){
    int x = 1;

    while(x<=10){
        x++;
        if(x==9){
            break;
        }
        cout << x << endl;
    }
    cout << “x = “ << x << endl;
}
```

Observação:

O comando *continue* dentro de uma estrutura de repetição serve para “pular” as instruções restantes contidas no corpo e realiza a próxima iteração do loop.

Exemplo:

```
int main(){
    int x = 1;

    while(x<=10){
        x++;
        if(x==9){
            continue;
        }
        cout << x << endl;
        x++;
    }
    cout << “x = “ << x << endl;
}
```

Saída: ?

Exercício

- Dado um número inteiro x, crie um programa para verificar se x é um número primo. Imprima na saída o resultado.

Obs: um número primo é aquele que é divisível somente por 1 e por ele mesmo.

Exercício

```
int main(){
  int n, i;
  bool isPrime = true;
  cout << "Entre com um inteiro positivo: ";
  cin >> n;

  for(i = 2; i <= n / 2; ++i){
      if(n % i == 0){
          isPrime = false;
          break;
      }
  }
  if (isPrime)
      cout << "Primo";
  else
      cout << "Não Primo";
  return 0;
}
```

Comandos de repetição `for`

- É controlada por um contador, sabemos quantas vezes o *loop* será executado;
- Uma variável de controle, que faz parte do próprio laço é utilizada para contar as repetições;
- Possui na variável de controle(contador) um valor inicial, podendo ser incrementada/decrementada até que se alcance uma condição de parada.

Comandos de repetição `for`

Estrutura geral:

```
for (exp1; exp2; exp3)
        Instrução/bloco de instruções
```

Obs: exp1 e 3 podem ser listas de expressões separadas por vírgula. Desta forma, pode-se ter uma ou mais variáveis de controle.

Comandos de repetição `for`

Exercício: Implemente um laço de repetição para imprimir os números de 0 até 10 na tela.

Comandos de repetição for

```
#include <iostream>
using namespace std;

int main() {
    for(int n = 0; n <= 9; n++){
        cout << n << endl;
    }

    return 0;
}
```

Comandos de repetição `for`

Exercício: Implemente um laço de repetição para calcular o fatorial de um dado número *x* positivo.

Obs: Sabe-se que o fatorial de *x* é o produto de todos os números de 1 a *x* (inclusive).

Comandos de repetição for

```
int main(){
    int i, n, fatorial = 1;

    cout << "Entre com um numero inteiro positivo: ";
    cin >> n;

    for (i = 1; i <= n; ++i) {
        fatorial *= i;   // fatorial = fatorial * i;
    }
    cout<< "Fatorial de "<< n <<"! = "<< fatorial;
    return 0;
}
```

Exercício: Escreva um programa para calcular e exibir todos os números primos, dentro de um intervalo dado por dois números.

Obs: Este problema pode ser resolvido utilizando comandos de decisão if...else e laços de de repetição

Saída desejada:

```
Entre com dois numeros(intervalo): 20 50
Primos dentro do intervalo de 20 até 50 são: 23 29 31 37 41 43 47
```