BCC 201 - Introdução à Programação

Controle de Fluxo
Comandos de repetição: while e do-while

Guillermo Cámara-Chávez
UFOP

Comandos de Repetição (Laços) I

- São muito comuns as situações em que se deseja repetir um determinado trecho de um programa um certo número de vezes.
- As estruturas de repetição são muitas vezes chamadas de Laços ou também de *Loops*.

Comandos de Repetição (Laços) II

- Classificação:
  - Laços Contados:
    - Conhecimento previo de quantas vezes o comando no interior da construção será executado;
  - Laços Condicionais:
    - Não se conhece de antemão o número de vezes que o conjunto de comandos no interior do laço será repetido.
    - Amarrado a uma condição sujeita à modificação pelas instruções do interior do laço.

Comandos de Repetição (Laços) I

- O conjunto de comandos em seu interior é executado até que uma determinada condição seja satisfeita.
- Laços condicionais mais comuns nas linguagens de programação modernas:

  Enquanto : laço condicional com teste no início

  Faça-enquanto : laço condicional com teste no final

Comandos de Repetição (Laços) II

- A variável que é testada deve estar sempre associada a um comando que a atualize no interior do laço.
  - Caso isso não ocorra, o programa ficará repetindo indefinidamente este laço, gerando um laço “infinito”.

Laços Condicionais com Teste no Inicio - *while* I

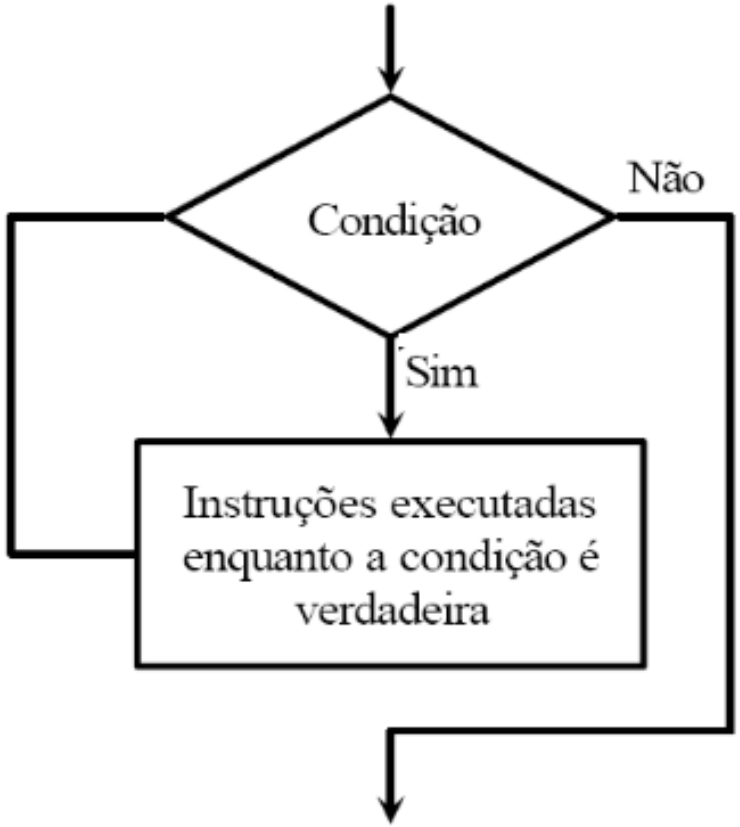
- O comando `while` define uma estrutura de repetição na qual:
  - o número de repetições pode não ser conhecido *a priori*,
  - a repetição será controlada pelo valor de uma expressão.

Laços Condicionais com Teste no Inicio - *while* II

- A sintaxe geral do comando while é a seguinte:

```
while(expressão)
{
        <sequência de comandos>;
}
```

Laços Condicionais com Teste no Inicio - *while* III



Laços Condicionais com Teste no Inicio - *while* IV

- Se a condição for verdadeira (= =1) o corpo do laço é executado e a condição é novamente avaliada.
- Esta operação se repete até que a condição se torne falsa (= =0), encerrando o laço e continuando a executar o programa depois do corpo do laço.
- O laço *while* é mais apropriado para situações que a repetição possa ser encerrada inesperadamente,

Laços Condicionais com Teste no Inicio - *while* V

- Enquanto que o *for* é mais empregado em quantidades de repetições mais conhecidas.
- O *while* também pode ser aninhado, ou seja, possuir um while dentro de outro *while*.

While I

Escreva um programa em C++ para calcular o fatorial de um número inteiro e positivo fornecido pelo usuário do programa

While II

```
#include<iostream>
using namespace std;
int main()
{
    double fat = 1;
    int i, n;
    cout << "Inserir numero \n";
    cin >> n;
    i = 1;
    while(i <= n)
    {
        fat = fat * i;
        i++;
    }
    cout << "O Fatorial de "<< n << "eh " << fat;
    return 0;
}
```

While III

Escreva o algoritmo e o respectivo programa em C++ para um programa que conte a quantidade de números pares e ímpares digitados por um usuário. O usuário pode digitar quantos números quiser, e pode encerrar o programa quando desejar.

While IV

```
#include<iostream>
using namespace std;
int main(){
    int contaPar = 0, contaImpar = 0, num;
    char resp = 's';
    while (resp = 's' || resp == 'S')
    {
        cout << "Inserir numero \n";
        cin >> num;
        if (num % 2 == 0)
            contaPar++;
        else
            contaImpar++;
        continua << "Continua s/n ? \n");
        cin >> resp;
    }
    cout <<"Numero de Pares: "<< contaPar << endl;
    cout <<"Numero de Impares: "<< contaImpar;
    return 0;
}
```

While V

Escreva o programa em C++, para um programa que conte a quantidade de números primos positivos digitados por um usuário. O usuário pode digitar quantos números quiser, e pode encerrar o programa quando desejar.

While VI

```
#include<iostream>
using namespace std;
int main(){
    int cont, num, i;
    char resp = 's';
    while (resp = 's' || resp == 'S')
    {
        cout << "Digite um numero \n";
        cin >> num;
        i = 2; cont = 0;
        while(i <= num)
        {
            if (num % i == 0) cont++;
        }
        if (cont == 1) cout << num << " eh primo \n";
        cout  << "continua s/n?";
        cin >> resp;
    }
    return 0;}
```

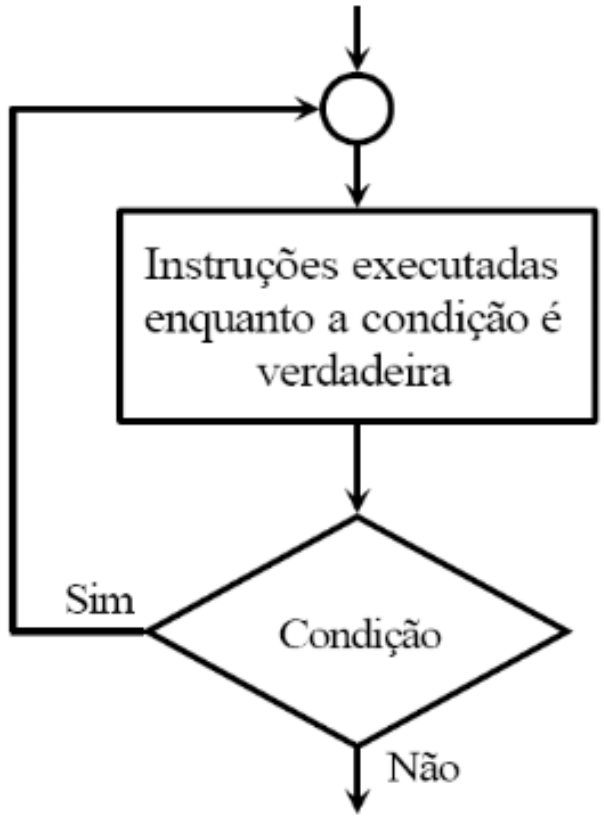
Laços Condicionais com Teste no Inicio - *do-while* I

- Efetua um teste lógico no final de um laço, verificando se é permitido ou não executar novamente o conjunto de comandos no interior do mesmo.
- Na construção *do-while* o comando é executado uma ou mais vezes (pelo menos uma vez).
- Na construção *while* o comando é executado zero ou mais vezes

Laços Condicionais com Teste no Inicio - *do-while* II

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

Laços Condicionais com Teste no Inicio - *do-while* III



Laços Condicionais com Teste no Inicio - *do-while* IV

A conversão de graus Fahrenheit para Centígrados é obtida pela fórmula $C = 5/9(F - 32)$. Escreva um algoritmo que calcule e escreva uma tabela de graus centígrados em função de graus Fahrenheit que variem de 150 até 50 de 0,5 em 0,5.

Laços Condicionais com Teste no Inicio - *do-while* V

```
#include<iostream>
using namespace std;
int main()
{
    double F = 150, C;
    cout << "Fahrenheit   |   Centigrados\n";
    do
    {
        C = 5.0 / 9 * (F - 32);
        cout << setw(10) << fixed << setprecision(2)
             << F <<"   |" << setw(10) <<  C << endl;
        F -= 0.5;
    } while (F >= 50);
    return 0;
}
```

Laços Condicionais com Teste no Inicio - *do-while* VI

Fazer um algoritmo para calcular a média aritmética de um conjunto de valores inteiros positivos (considere o valor zero como finalizador do programa).

Laços Condicionais com Teste no Inicio - *do-while* VII

```
#include<iostream>
using namespace std;
int main()
{
    int soma = 0, cont = 0, num;
    do{
        cout << "Inserir numero \n";
        cin >> num;
        if (num > 0)
        {
            soma += num;
            cont++;
        }
    } while(num != 0);
    if (cont > 0)
        cout << static_cast<double>(soma) / cont;
    return 0;
}
```

Laços Condicionais com Teste no Inicio - *do-while* VIII

Dado um país A, com 5.000.000 habitantes e uma taxa de crescimento de 3% ao ano, e um país B com 7.000.000 habitantes e uma taxa de natalidade de 2% ao ano. Construa um algoritmo que calcule e imprima quanto tempo é necessário para que a população do país A ultrapasse a população do país B.

Laços Condicionais com Teste no Inicio - *do-while* IX

```
#include<iostream>
using namespace std;
int main()
{
    double populacaoA = 5000000, populacaoB = 7000000;
    int cont = 0;
    while(populacaoA < populacaoB)
    {
        cont++;
        populacaoA = populacaoA + 0.03 * populacaoA;
        populacaoB = populacaoB + 0.02 * populacaoB;
    }
    cout << "Eh necessario " << cont << "anos \n";
    return 0;
}
```

FIM