

Estruturas de Repetição

Parte 4

Soma acumulada
Repetições Aninhadas

Sumário

- Soma acumulada (controle por leitura)
- Repetições aninhadas

Soma acumulada de sequência terminada com zero

Mostrar sucessor de número lido.
Termina ao digitar **0** (zero).

```
/* Programa 'seqSucessor' */
#include <iostream>
using namespace std;

int main()
{
    int n;

    cin >> n;
    while ( n != 0 )
    {
        cout << n + 1 << endl;

        cin >> n ;
    }

    return 0;
}
```

Obtém 10 valores do usuário e apresenta a soma destes valores.

```
/* Programa 'Soma10Valores' */
#include <iostream>
using namespace std;

int main()
{
    int n, soma, k;

    k = 0;
    soma = 0;
    while ( k < 10 ) {
        cin >> n;
        soma = soma + n;
        k = k + 1;
    }

    cout << soma << endl;
    return 0;
}
```

Soma acumulada de sequência terminada com zero

Escrever programa que obtém valores do usuário e apresenta a soma destes valores. Termina quando ler valor 0 (zero), **que não deve ser somado**.

```
/* Programa 'Soma10Valores' */
#include <iostream>
using namespace std;

int main()
{
    int n, soma;

    soma = 0;

    cin >> n;
    while ( n != 0 ) {

        soma = soma + n;

        cin >> n;
    }

    cout << soma << endl;
    return 0;
}
```

Soma acumulada de sequência terminada com zero

Escrever programa que obtém valores do usuário e apresenta a soma destes valores e a **quantidade** de valores lidos. Termina quando ler valor 0 (zero), **que não deve ser usado na soma e na contagem**.

```
/* Programa 'Soma10Valores' */
#include <iostream>
using namespace std;

int main()
{
    int n, soma, k ;

    soma = 0;
    k = 0; // k: é contador MAS não controla a repetição

    cin >> n; // n: é dado de entrada E também controla a repetição
    while ( n != 0 ) {
        soma = soma + n;
        k = k + 1;
        cin >> n;
    }

    cout << soma << " " << k << endl;

    return 0;
}
```

Exercício

Escrever programa que obtém valores do usuário e apresenta a média aritmética destes valores. Termina quando ler valor 0 (zero), **que não deve ser usado no cálculo da média.**

Exercício (codeblocks)

The screenshot displays the Code::Blocks IDE interface. The main editor window shows a C++ file named `media.cpp` with the following code:

```
1  /* Programa 'mediaAritmetica' */
2
3  #include <iostream>
4  using namespace std;
5
6  int main()
7  {
8      int n, k;
9      float soma;
10
11     soma = 0;
12     k = 0;
13     cin >> n;
14     while ( n != 0 ) {
15         soma = soma + n;
16         k = k + 1;
17         cin >> n;
18     }
19
20     cout << "média = " << soma / k << endl;
21
22     return 0;
23 }
```

Overlaid on the right side of the IDE is a terminal window titled "Running". It shows the output of the program:

```
1
2
3
4
5
6
7 média = 3.5
8
9 Process returned 0 (0x0)   execution time : 25.539 s
10 Press ENTER to continue.
11
```

At the bottom of the IDE, the "Logs & others" panel is visible, showing a message:

```
F: L: Message
=== Build file: "no target" in "no project" (compiler: unknown) ===
=== Build finished: 0 error(s), 0 warning(s) (0 minute(s), 0 second(s)) ===
```

The status bar at the very bottom indicates the current file path, encoding, and cursor position: `/home/nicolui/doc/UFPR/Gradua... C/C++ Unix (LF) UTF-8 Line 20, Col 15, Pos 282`.

Repetições Aninhadas

Da mesma forma que na estrutura **if-else**, podemos ter **repetições aninhadas**

- repetições de repetições
- repetições dentro de repetições.

Repetições Aninhadas

Escrever um programa que leia um conjunto de valores inteiros n (até ser digitado 0) e para cada valor, mostre uma linha n asteriscos ($*$).

Ler inteiros n
até digitar 0 (zero)

Mostrar n asteriscos
em uma mesma linha

Programa desejado:
Ler inteiros n e mostrar
 n asteriscos.
Termina quando digita 0 para n .

Repetições Aninhadas

Ler **n** valores até digitar **0**

```
/* Programa 'mostraNem[a,b]' */
#include <iostream>
using namespace std;

int main()
{
    int n;

    cin >> n;
    while ( n != 0 ) {

        cout << n;

        cout << endl;
        cin >> n;
    }

    return 0;
}
```

Mostrar **n** asteriscos em uma mesma linha

```
/* Programa 'tabuadaN' */
#include <iostream>
using namespace std;

int main()
{
    int n, k;

    n = 5; // exemplo para 5 asteriscos

    k = 0;
    while ( k < n ) {
        cout << "*" ;
        k = k + 1;
    }

    return 0;
}
```

Repetições Aninhadas

Escrever um programa que leia um conjunto de valores inteiros **n** (até ser digitado **0**) e para cada valor, mostre uma linha **n** asteriscos (*).

```
/* Programa 'tabuada' */
#include <iostream>
using namespace std;

int main()
{
    int n, k;

    cin >> n;
    while ( n != 0 ) {

        k = 0;
        while ( k < n ) {
            cout << "*" ;
            k = k + 1;
        }

        cout << endl;
        cin >> n;
    }

    return 0;
}
```

Repetições Aninhadas

Escrever um programa que mostre a tabuada dos inteiros no intervalo $[a, b]$, $a \leq b$ lidos pelo teclado.

Mostrar os n inteiros no intervalo $[a, b]$

Mostrar a tabuada de um valor n

Programa desejado:
Mostrar a tabuada de cada inteiro n no intervalo $[a, b]$

Repetições Aninhadas

Mostrar os n inteiros no intervalo $[a, b]$

```
/* Programa 'mostraNem[a,b]' */
#include <iostream>
using namespace std;

int main()
{
    int a, b, n;

    cin >> a >> b;

    n = a;
    while ( n <= b ) {
        cout << n << endl;

        n = n + 1;
    }

    return 0;
}
```

Mostrar a tabuada de um valor n

```
/* Programa 'tabuadaN' */
#include <iostream>
using namespace std;

int main()
{
    int n, k;

    n = 3; // tabuada do 3.

    k = 1;
    while ( k <= 10 ) {
        cout << n << " x " << k << " = " << n * k << endl;
        k = k + 1;
    }

    return 0;
}
```

Repetições Aninhadas

Escrever um programa que mostre a tabuada dos inteiros no intervalo [a, b], $a \leq b$ lidos pelo teclado.

```
/* Programa 'tabuada' */
#include <iostream>
using namespace std;

int main()
{
    int a, b, n, k;

    cin >> a >> b;

    n = a;
    while (n <= b) {
        k = 1;
        while (k <= 10) {
            cout << n << " x " << k << " = " << n * k << endl;
            k = k + 1;
        }
        n = n + 1;
    }

    return 0;
}
```

Repetições Aninhadas

Dado um valor n lido pelo teclado, exibir n linhas em que cada linha contém tantos '*' quanto for a ordem da linha, conforme padrão abaixo:

```
*  
**  
***  
****  
*****  
.....  
** .....* (N asteriscos)
```

Mostrar n asteriscos,
um em cada linha

Mostrar i asteriscos
seguidos, sem mudar
de linha

Programa desejado:
Mostrar o padrão de asteriscos

Repetições Aninhadas

Mostrar n asteriscos, um em cada linha

```
/* Programa 'mostraNem[a,b]' */
#include <iostream>
using namespace std;

int main()
{
    int n, i;

    cin >> n;

    i = 1;
    while ( i <= n ) {
        cout << "*" ;

        cout << endl;
        i = i + 1;
    }

    return 0;
}
```

Mostrar i asteriscos seguidos, sem mudar de linha

```
/* Programa 'tabuadaN' */
#include <iostream>
using namespace std;

int main()
{
    int j, i;

    i = 5; // por exemplo, linha i = 5.

    j = 0;
    while ( j < i ) {
        cout << "**";
        j = j + 1;
    }

    return 0;
}
```


Repetições Aninhadas

Dado um valor n lido pelo teclado, exibir n linhas em que cada linha contém tantos "*" quanto for a ordem da linha, conforme padrão abaixo:

```
/* Programa 'tabuada' */
#include <iostream>
using namespace std;

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

    cin >> n;
    i = 1;
    while (i <= n) {
        j = 0;
        while (j < i) {
            cout << "*";
            j = j + 1;
        }
        cout << endl;
        i = i + 1;
    }
    return 0;
}
```

// i: "percorre" as linhas (de linha 1 a linha n)

// j: "percorre" os caracteres em cada linha → "colunas"
// i é usada como quantidade de asteriscos em uma linha

Exercícios para aula *online*

Após assistir todas as vídeo-aulas da semana, procure trabalhar na **Lista de exercícios** do Tópico **Estruturas de Repetição**, na sala virtual da disciplina na UFPR Virtual.

Estes exercícios serão usados nas aulas *online* para esclarecer e consolidar os conceitos abordados até aqui.

Leitura complementar

Acesse as **Leituras complementares** do Tópico **Estruturas de Repetição**, na sala virtual da disciplina da UFPR Virtual.

Elas são importantes e auxiliam na compreensão dos temas abordados até aqui.

Créditos: O conteúdo original deste documento é de autoria do Prof. Armando L.N. Delgado (DINF/ET) para uso na disciplina *Programação de Computadores* (CI208, CI180, CI183)
Compartilhe este documento de acordo com a licença abaixo



Este documento está licenciado com uma Licença *Creative Commons Atribuição-NãoComercial-SemDerivações* 4.0 Internacional.
<https://creativecommons.org/licenses/by-nc-sa/4.0/>