



Aula 16: Laços aninhados e desvios

Introdução a Programação

Túlio Toffolo & Puca Huachi
<http://www.toffolo.com.br>

Aula Anterior

- Comandos de Repetição (Parte 3)
- Exercícios no *moodle*

Aula de Hoje

- 1 Exercícios
- 2 Laços Aninhados
- 3 Comando `continue`
- 4 Comando `break`
- 5 Exercício
- 6 Próxima aula

Aula de Hoje

- 1 Exercícios
- 2 Laços Aninhados
- 3 Comando `continue`
- 4 Comando `break`
- 5 Exercício
- 6 Próxima aula

Exercício da aula passada

Exercício 1

A Sequência de **Fibonacci** é uma sequência de números inteiros iniciando por 0, seguido por 1 e depois pela soma dos dois anteriores: (0, 1, 1, 2, 3, 5, 8, ...).

Escreva um programa (utilizando o comando de repetição `for`) que imprime os n primeiros números da sequência de Fibonacci. O usuário deve informar o valor de n .

Exemplo:

```
1 Qual o valor de n? 7
2
3 0 1 1 2 3 5 8
```

Exercício da aula passada

Série de Fibonacci

0, 1, 1, 2, 3, 5, 8, 13, 21, 34, 55, 89, 144, 233, 377, 610, 987, 1597, 2584, 4181, 6765, 10946, ...

$$F(n) = \begin{cases} 0 & \text{se } n = 0, \\ 1 & \text{se } n = 1, \\ F(n-1) + F(n-2) & \text{caso contrário.} \end{cases}$$

```
1  #include <stdio.h>
2
3  int main()
4  {
5      int n, termo1 = 0, termo2 = 1, aux;
6      printf("Qual o valor de n?: ");
7      scanf("%d", &n);
8      printf("\n");
9
10     if (n >= 1)
11         printf("%d ", termo1);
12
13     if (n >= 2)
14         printf("%d ", termo2);
15
16     for (int i = 3; i <= n; i++) {
17         aux = termo1 + termo2;
18         printf("%d ", aux);
19         termo1 = termo2;
20         termo2 = aux;
21     }
22
23     printf("\n");
24     return 0;
25 }
```

Aula de Hoje

- 1 Exercícios
- 2 Laços Aninhados**
- 3 Comando `continue`
- 4 Comando `break`
- 5 Exercício
- 6 Próxima aula

Laços Aninhados

```
1 1
2 22
3 333
4 4444
5 55555
6 666666
7 7777777
8 88888888
9 999999999
```

Repetição 1: temos nove repetições de linhas ($1 \leq n \leq 9$).

Repetição 2: temos, em cada linha, a repetição de n caracteres que identificam a própria linha, sendo $1 \leq n \leq 9$. Assim, temos $n = 1$ na linha 1, $n = 2$ na linha 2, e assim sucessivamente, até a linha 9.

- Para obter a saída acima, realizamos a **Repetição 2** dentro da **Repetição 1**.

Laços Aninhados

```
1  #include <stdio.h>
2
3  int main()
4  {
5      // Repetição 1
6      for (int linha = 1; linha <= 8; linha++) {
7
8          // Repetição 2
9          for (int coluna = 1; coluna <= linha; coluna++) {
10             printf("%d", linha);
11         }
12
13         printf("\n");
14     }
15
16     return 0;
17 }
```

Laços Aninhados

```
1 Contador externo (linha): 1
2
3     Contador interno (coluna): 1
4     Contador interno (coluna): 2
5     Contador interno (coluna): 3
6     Contador interno (coluna): 4
7
8 Contador externo (linha): 2
9
10    Contador interno (coluna): 1
11    Contador interno (coluna): 2
12    Contador interno (coluna): 3
13    Contador interno (coluna): 4
14
15 Contador externo (linha): 3
16
17    Contador interno (coluna): 1
18    Contador interno (coluna): 2
19    Contador interno (coluna): 3
20    Contador interno (coluna): 4
```

Laços Aninhados

```
1  #include <stdio.h>
2
3  int main()
4  {
5      // Repetição variando a <linha>
6      for (int linha = 1; linha <= 3; linha++) {
7          printf("Contador externo (linha): %d\n\n", linha);
8
9          // Repetição variando a <coluna>
10         for (int coluna = 1; coluna <= 4; coluna++) {
11             printf("\t\tContador interno (coluna): %d\n", coluna);
12         }
13
14         printf("\n");
15     }
16
17     return 0;
18 }
```

Exemplo 1

Faça um programa que imprime a tabuada de x até y (valores de x e y devem ser digitados pelo usuário).

```
1  Digite os valores para x e y: 5 15
2
3  Tabuada de multiplicação!
4
5      |  5   6   7   8   9  10  11  12  13  14  15
6  -----
7  5 |  25  30  35  40  45  50  55  60  65  70  75
8  6 |  30  36  42  48  54  60  66  72  78  84  90
9  7 |  35  42  49  56  63  70  77  84  91  98 105
10 8 |  40  48  56  64  72  80  88  96 104 112 120
11 9 |  45  54  63  72  81  90  99 108 117 126 135
12 10 | 50  60  70  80  90 100 110 120 130 140 150
13 11 | 55  66  77  88  99 110 121 132 143 154 165
14 12 | 60  72  84  96 108 120 132 144 156 168 180
15 13 | 65  78  91 104 117 130 143 156 169 182 195
16 14 | 70  84  98 112 126 140 154 168 182 196 210
17 15 | 75  90 105 120 135 150 165 180 195 210 225
```

```

1  int main()
2  {
3      int x, y;
4      printf("Digite os valores para x e y: ");
5      scanf("%d %d", &x, &y);
6
7      // imprimindo o cabeçalho
8      printf("\nTabuada de multiplicação!\n\n");
9      printf("    | ");
10     for (int j = x; j <= y; j++)
11         printf("%3d ", j);
12     printf("\n----");
13     for (int j = x; j <= y; j++)
14         printf("----");
15     printf("\n");
16
17     // calculando (e imprimindo) a tabuada
18     for (int i = x; i <= y; i++) {
19         printf("%2d | ", i);
20         for (int j = x; j <= y; j++)
21             printf("%3d ", i*j);
22         printf("\n");
23     }
24     return 0;
25 }

```

Exemplo 2

Bart Simpson está aprendendo a jogar xadrez, mas tem dificuldade em saber para qual direção ele pode mover sua **Torre**.

Sabemos que um tabuleiro de xadrez é composto por 8 linhas e 8 colunas, e que a **Torre** se move ortogonalmente, ou seja, pelas linhas (horizontais) e pelas colunas (verticais).

- Escreva um programa que solicite ao *Bart* o número da linha e da coluna que indicam a posição de sua **Torre**. O programa deve imprimir quais são os possíveis movimentos da **Torre**.
- Utilize "**—**" para indicar uma casa para a qual a Torre não pode ser movida e "**x**" para indicar uma casa para a qual ela pode ser movida.

Exemplo 2

Exemplo de saída:

```
1  Movimentos de uma Torre no xadrez!
2  Digite a linha em que a Torre se encontra: 6
3  Digite a coluna em que a Torre se encontra: 3
4
5  Movimentos possíveis:
6
7      1  2  3  4  5  6  7  8
8  -----
9  1 | - - x - - - - -
10 2 | - - x - - - - -
11 3 | - - x - - - - -
12 4 | - - x - - - - -
13 5 | - - x - - - - -
14 6 | x x x x x x x x
15 7 | - - x - - - - -
16 8 | - - x - - - - -
```



```

1  int main()
2  {
3      int linha, coluna;

4
5      printf("Movimentos de uma Torre no xadrez!\n");
6      printf("Digite a linha em que a Torre se encontra: ");
7      scanf("%d", &linha);
8      printf("Digite a coluna em que a Torre se encontra: ");
9      scanf("%d", &coluna);

10
11     // Imprime o cabeçalho da tabela antes do loop
12     printf("\nMovimentos possíveis:\n\n");
13     printf("      1  2  3  4  5  6  7  8  \n");
14     printf("      -----\n");

15
16     // Imprime a tabela
17     for (int l = 1; l <= 8; l++) {
18         printf(" %d | ", l);
19         for (int c = 1; c <= 8; c++) {
20             if (l == linha || c == coluna) {
21                 printf(" x ");
22             }
23             else {
24                 printf(" - ");
25             }
26         }
27         printf("\n");
28     }
29     return 0;
30 }

```

Exemplo 3

O valor de π pode ser calculado pelo seguinte algoritmo:

Passo 1) Calcula-se a série com n termos:

$$S = \frac{1}{1^3} - \frac{1}{3^3} + \frac{1}{5^3} - \frac{1}{7^3} + \dots$$

Passo 2) O valor aproximado de π é dado pela expressão:

$$\pi = \sqrt[3]{S \times 32}$$

Faça um programa para calcular e imprimir o valor aproximado de π (use 20 casas decimais). O programa deve ler o número de termos. Exemplo:

```
1 Digite o número de termos: 10
2 Valor aproximado de pi: 3.14152608792950616134
```

```
1  #include <stdio.h>
2  #include <math.h>
3
4  int main()
5  {
6      // lendo o nro de termos (n)
7      int n;
8      printf("Digite o número de termos: ");
9      scanf("%d", &n);
10
11     // calculando o valor de s
12     double s = 0, sinal = +1;
13     for (int i = 0; i < n; i++) {
14         s += sinal / pow(1.0 + 2*i, 3);
15         sinal *= -1;
16     }
17
18     // calculando e imprimindo o valor de pi
19     double pi = cbrt(s * 32);
20     printf("Valor aproximado de pi: %.20lf\n", pi);
21
22     return 0;
23 }
```

Exemplo 4

Crie um programa que calcula o valor da expressão a seguir. Considere que os valores de n e m serão fornecidos pelo usuário.

$$x = \sum_{i=1}^n \sum_{j=1}^m (i + j)$$

Exemplo de execução:

```
1 Digite os valores de n e m: 10 5
2
3 x = 425.000000
```

Exemplo 4

```
1  #include <stdio.h>
2
3  int main()
4  {
5      // lendo os valores de n e m
6      int n, m;
7      printf("Digite os valores de n e m: ");
8      scanf("%d %d", &n, &m);
9
10     // calculando o valor de x
11     double x = 0;
12     for (int i = 1; i <= n; i++) {
13         for (int j = 1; j <= m; j++) {
14             x += i + j;
15         }
16     }
17
18     // imprimindo o valor de x na saída
19     printf("\nx = %lf\n", x);
20     return 0;
21 }
```

Aula de Hoje

- 1 Exercícios
- 2 Laços Aninhados
- 3 Comando continue**
- 4 Comando break
- 5 Exercício
- 6 Próxima aula

Desvio em laço

Em diversos momentos queremos **alterar o fluxo** ou mesmo encerrar a execução de um laço de repetição.

Um dos comandos utilizados para isso é o `continue`

- Este comando permite alterar o fluxo do laço, fazendo-o retornar ao início.
- É particularmente útil para evitar `if` aninhados em alguns casos.

Comando `continue`

Exemplo de uso em laço `while`:

```
1 while (<condição>) {  
2     <comando_1>;  
3     ...  
4     continue;  
5     ...  
6     <comando_n>;  
7 }
```


Comando `continue`

Exemplo de uso em laço `do-while`:

```
1 do {  
2     <comando_1>;  
3     ...  
4     continue;  
5     ...  
6     <comando_n>;  
7 } while (<condição>);
```

Comando `continue`

Exemplo de uso em laço `for`:

```
1  for (<inicialização>; <condição>; <incremento/decremento>) {  
2      <comando_1>;  
3      ...  
4      continue;  
5      ...  
6      <comando_n>;  
7  }
```

Exemplo

Faça um aplicativo em C que some todos os números, de 1 até 100, exceto os múltiplos de 5.

Precisamos do comando `continue` para criar tal programa?

- Definitivamente **não**!
- Mas o `continue` é uma alternativa válida que (em alguns casos) simplifica o código.

Exemplo

Usando o `continue`:

```
1  #include <stdio.h>
2
3  int main()
4  {
5      int soma = 0;
6
7      for (int cont = 1; cont <= 100; cont++) {
8          if (cont % 5 == 0)
9              continue;
10         soma += cont;
11     }
12     printf("Soma = %d\n", soma);
13     return 0;
14 }
```

Exemplo

Alternativa sem o `continue`:

```
1  #include <stdio.h>
2
3  int main()
4  {
5      int soma = 0;
6
7      for (int cont = 1; cont <= 100; cont++) {
8          if (cont % 5 != 0) {
9              soma += cont;
10         }
11     }
12     printf("Soma = %d\n", soma);
13     return 0;
14 }
```

Aula de Hoje

- 1 Exercícios
- 2 Laços Aninhados
- 3 Comando `continue`
- 4 Comando `break`**
- 5 Exercício
- 6 Próxima aula

Desvio em laço

Em diversos momentos queremos alterar o fluxo ou mesmo **encerrar** a execução de um laço de repetição.

Um dos comandos utilizados para isso é o `break`

- Este comando permite encerrar o laço imediatamente.
- Assim como o `continue`, é particularmente útil para evitar uma quantidade excessiva de `if` aninhados.

Comando break

Exemplo de uso em laço `while`:

```
1 while (<condição>) {  
2     <comando_1>;  
3     ...  
4     break;  
5     ...  
6     <comando_n>;  
7 }
```


Comando break

Exemplo de uso em laço `do-while`:

```
1 do {  
2     <comando_1>;  
3     ...  
4     break;  
5     ...  
6     <comando_n>;  
7 } while (<condição>);
```

Comando break

Exemplo de uso em laço **for**:

```
1  for (<inicialização>; <condição>; <incremento/decremento>) {  
2      <comando_1>;  
3      ...  
4      break;  
5      ...  
6      <comando_n>;  
7  }
```

Exemplo

Faça um programa que imprime o primeiro número, entre 1 e 1 milhão, que é divisível por 11, 13 e 17.

Precisamos do comando `break` para criar tal programa?

- Definitivamente **não**!
- Mas o `break` neste caso simplifica o código.

Exemplo

```
1  #include <stdio.h>
2
3  int main()
4  {
5      for (int cont = 1; cont <= 1000000; cont++) {
6          if (cont % 11 == 0 && cont % 13 == 0 && cont % 17 == 0) {
7              printf("O número é %d\n", cont);
8              break;
9          }
10     }
11     return 0;
12 }
```

Aula de Hoje

- 1 Exercícios
- 2 Laços Aninhados
- 3 Comando `continue`
- 4 Comando `break`
- 5 Exercício**
- 6 Próxima aula

Exercício

Exercício 1

Apresente um programa em C que imprime uma tabela contendo a tabuada de multiplicação de 1 a 20 **ignorando os números pares**.

Exemplo:

1		1	3	...	19
2	-----				
3	1	1	3		19
4	3	3	9	...	57
5	5	5	15	...	95
6				...	
7	19	19	57	...	361

Dica: use "%3d " para ficar bonito! :)

Aula de Hoje

- 1 Exercícios
- 2 Laços Aninhados
- 3 Comando `continue`
- 4 Comando `break`
- 5 Exercício
- 6 **Próxima aula**

Próxima Aula

Aula prática:

- Laços de repetição

Aula teórica:

- Aula mais curta: assembleia do DECOM marcada para 11h30
- Estruturas de dados homogêneas unidimensionais



Perguntas?