Aula: Repetição - for e laços aninhados

Introdução a Programação

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

Departamento de Computação
Universidade Federal de Ouro Preto

Aula: Repetição - for e laços aninhados

Aula de Hoje

Laços ou Repetições (*loop*)

- Em um laço **controlado logicamente**, os comandos (corpo do laço) são repetidos enquanto uma **expressão lógica for verdadeira**
  - `while`
  - `do - while`
- Em um laço **controlado por contador**, os comandos (corpo do laço) são repetidos um **número predeterminado de vezes**
  - `for`

Laços `for`

Em um laço **controlado por contador**, os comandos (corpo do laço) são repetidos um **número predeterminado de vezes**.

Sintaxe:

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

Exemplo 1

Em matemática, a fórmula de Leibniz para π, estabelece que:

$$\frac{\pi}{4} = 1 - \frac{1}{3} + \frac{1}{5} - \frac{1}{7} + \frac{1}{9} - \cdots$$

Faça um programa em C para calcular o valor aproximado de π. Quanto maior o número de termos melhor será a aproximação. O número de termos deve ser informado pelo usuário.

Dica: note que a soma pode ser escrita como

$$\frac{\pi}{4} = \frac{1}{1+2\cdot 0} - \frac{1}{1+2\cdot 1} + \frac{1}{1+2\cdot 2} - \frac{1}{1+2\cdot 3} + \frac{1}{1+2\cdot 4} - \cdots$$

Qual comando de repetição usar?

Exemplo 1

```c
#include <stdio.h>

int main()
{
    int n;
    printf("Quantos termos quer utilizar? ");
    scanf("%d", &n);

    double pi = 0;
    for (int i = 0; i < n; i++) {
        if (i % 2 == 0)
            pi += 1.0 / (1 + i*2);
        else
            pi -= 1.0 / (1 + i*2);
    }
    pi *= 4;
    printf("Valor de pi calculado: %.6lf", pi);

    return 0;
}
```

Exemplo 1 (solução alternativa)

```c
#include <stdio.h>

int main()
{
    int n;
    printf("Quantos termos quer utilizar? ");
    scanf("%d", &n);

    double pi = 0;
    double sinal = +1;
    for (int i = 0; i < n; i++) {
        pi += sinal * (1.0 / (1 + i*2));
        sinal *= -1;
    }
    pi *= 4;
    printf("Valor de pi calculado: %.6lf", pi);

    return 0;
}
```

Exemplo 2

Faça um programa em C para calcular a média quadrática de n valores digitados pelo usuário. A média quadrática é dada pela seguinte equação:

$$x_q = \sqrt{\frac{x_1^2 + x_2^2 + \ldots + x_n^2}{n}}$$

O programa deve ler o valor de n e os n valores de x. Em seguida, deve imprimir o resultado. Exemplo:

```
1  DIGITE O VALOR DE N: 2
2  DIGITE O VALOR DE X1: 2
3  DIGITE O VALOR DE X2: 2
4
5  MÉDIA QUADRÁTICA: 2
```

Podemos/devemos usar o comando de repetição `for` neste caso?

Exemplo 2

```c
int main()
{
    int n;
    printf("DIGITE O VALOR DE N: ");
    scanf("%d", &n);

    double soma = 0, nro;
    for (int i = 0; i < n; i++) {
        printf("DIGITE O VALOR DE X%d: ", i+1);
        scanf("%lf", &nro);
        soma += nro * nro;
    }

    double media = sqrt(soma / n);
    printf("MÉDIA QUADRÁTICA: %.0lf\n", media);
    return 0;
}
```

Aula de Hoje

Exercício

Apenas para praticar, use o comando `do-while` em uma das questões e o `for` na outra:

Exercício 1

Escreva um programa que imprima o quadrado dos números inteiros, no intervalo fechado de 1 a 20. A seguir, um exemplo de execução do programa.

```
1 4 9 16 25 36 49 64 81 100 121 144 169 196 225 256 289 ... 400
```

Exercício 2

Suponha que exista 50 alunos em uma sala. Faça um programa que determina quantos desses alunos tem idade maior que 20 anos. O usuário (coitado) deve digitar a matrícula e idade de todos os 50 alunos.

Aula de Hoje

Laços Aninhados

```
1
22
333
4444
55555
666666
7777777
88888888
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

```c
#include <stdio.h>

int main()
{
    // Repetição 1
    for (int linha = 1; linha <= 9; linha++) {

        // Repetição 2
        for (int coluna = 1; coluna <= linha; coluna++) {
            printf("%d", linha);
        }

        printf("\n");
    }

    return 0;
}
```

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

```c
#include <stdio.h>

int main()
{
// Repetição variando a <linha>
for (int linha = 1; linha <= 3; linha++) {
    printf("Contador externo (linha): %d\n\n", linha);

    // Repetição variando a <coluna>
    for (int coluna = 1; coluna <= 4; coluna++) {
        printf("\t\tContador interno (coluna): %d\n", coluna);
    }

    printf("\n");
}

return 0;
}
```

Exemplo 1

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

```
Digite os valores para x e y: 5 15

Tabuada de multiplicação!

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

```c
int main()
{
int x, y;
printf("Digite os valores para x e y: ");
scanf("%d %d", &x, &y);

// imprimindo o cabecalho
printf("\nTabuada de multiplicação!\n\n");
printf("   | ");
for (int j = x; j <= y; j++)
printf("%3d ", j);
printf("\n----");
for (int j = x; j <= y; j++)
printf("----");
printf("\n");

// calculando (e imprimindo) a tabuada
for (int i = x; i <= y; i++) {
printf("%2d | ", i);
for (int j = x; j <= y; j++)
printf("%3d ", i*j);
printf("\n");
}
return 0;
}
```

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:

```
Movimentos de uma Torre no xadrez!
Digite a linha  em que a Torre se encontra: 6
Digite a coluna em que a Torre se encontra: 3

Movimentos possíveis:

1  2  3  4  5  6  7  8
------------------------
1 |  -  -  x  -  -  -  -  -
2 |  -  -  x  -  -  -  -  -
3 |  -  -  x  -  -  -  -  -
4 |  -  -  x  -  -  -  -  -
5 |  -  -  x  -  -  -  -  -
6 |  x  x  x  x  x  x  x  x
7 |  -  -  x  -  -  -  -  -
8 |  -  -  x  -  -  -  -  -
```

```c
int main()
{
int linha, coluna;
printf("Movimentos de uma Torre no xadrez!\n");
printf("Digite a linha  em que a Torre se encontra: ");
scanf("%d", &linha);
printf("Digite a coluna em que a Torre se encontra: ");
scanf("%d", &coluna);
// Imprime o cabeçalho da tabela antes do loop
printf("\nMovimentos possíveis:\n\n");
printf("      1  2  3  4  5  6  7  8  \n");
printf("    ------------------------\n");

// Imprime a tabela
for (int l = 1; l <= 8; l++) {
printf(" %d | ", l);
for (int c = 1; c <= 8; c++) {
if (l == linha || c == coluna) {
printf(" x ");
}
else {
printf(" - ");
}
}
printf("\n");
}
return 0;
}
```

Exemplo 3

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 3

```c
#include <stdio.h>

int main()
{
// lendo os valores de n e m
int n, m;
printf("Digite os valores de n e m: ");
scanf("%d %d", &n, &m);

// calculando o valor de x
double x = 0;
for (int i = 1; i <= n; i++) {
for (int j = 1; j <= m; j++) {
x += i + j;
}
}

// imprimindo o valor de x na saída
printf("\nx = %lf\n", x);
return 0;
}
```

Aula de Hoje

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`:

```c
#include <stdio.h>

int main()
{
int soma = 0;

for (int cont = 1; cont <= 100; cont++) {
if (cont % 5 == 0)
continue;
soma += cont;
}
printf("Soma = %d\n", soma);
return 0;
}
```

Exemplo

Alternativa sem o `continue`:

```c
#include <stdio.h>

int main()
{
int soma = 0;

for (int cont = 1; cont <= 100; cont++) {
if (cont % 5 != 0) {
soma += cont;
}
}
printf("Soma = %d\n", soma);
return 0;
}
```

Aula de Hoje

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

```c
#include <stdio.h>

int main()
{
for (int cont = 1; cont <= 1000000; cont++) {
if (cont % 11 == 0 && cont % 13 == 0 && cont % 17 == 0) {
printf("O número é %d\n", cont);
break;
}
}
return 0;
}
```

Aula de Hoje

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    3      ...      19
-----------------------------
1 |    1    3               19
3 |    3    9      ...      57
5 |    5   15      ...      95
...
19 |   19   57       ...     361
```

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

Perguntas?