



Aula 14: Repetição (Parte 2)

Introdução a Programação

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

Aula Anterior

- Laços de repetição
- Comando de repetição `while`
- Exercícios

Aula de Hoje

- 1 Exercícios da aula passada
- 2 Comando `do-while`
- 3 Exemplos
- 4 Uso do laço
- 5 Exercícios

Aula de Hoje

- 1 Exercícios da aula passada
- 2 Comando `do-while`
- 3 Exemplos
- 4 Uso do laço
- 5 Exercícios

Exercício 1

A média harmônica é dada pela seguinte equação:

$$H = \frac{n}{\sum_{i=1}^n \frac{1}{x_i}}, \text{ ou seja: } H = \frac{n}{\frac{1}{x_1} + \frac{1}{x_2} + \dots + \frac{1}{x_n}}$$

Faça um programa em C para calcular a média harmônica de n valores digitados pelo usuário. Saída:

```
1  Cálculo da Média Harmônica.  
2  Quanto números você quer digitar? 4  
3  
4  Número 1: 2  
5  Número 2: 4  
6  Número 3: 6  
7  Número 4: 8  
8  
9  Média Harmônica: 3.84
```

Aula de Hoje

- 1 Exercícios da aula passada
- 2 Comando do-while**
- 3 Exemplos
- 4 Uso do laço
- 5 Exercícios

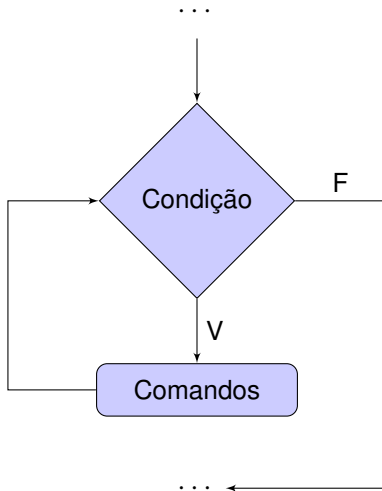
Laços ou Repetições (*loop*)

- Laços são comandos usados sempre que uma ou mais instruções devam ser repetidas enquanto um certa condição estiver sendo satisfeita
- Laços em C++
 - `while`
 - `do - while`
 - `for`

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`
- Denomina-se **iteração** a repetição de um conjunto de comandos: cada execução do corpo do laço, juntamente com a condição de terminação do laço, é uma iteração

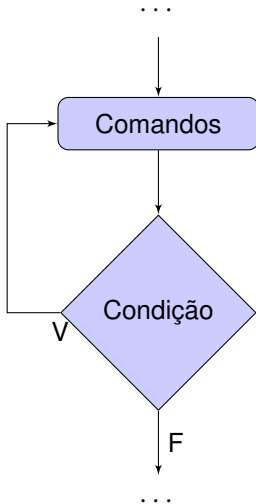
Laço while



Laço while

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

Laço do-while



Comando do-while

```
1 do {  
2     <comando_1>;  
3     ...  
4     <comando_n>;  
5 } while (<expressão>;
```

No comando **do-while**, ao contrário do comando `while`, o teste do laço/loop está no final, isso significa que os `<comandos>` serão executados **no mínimo uma vez**.

Aula de Hoje

- 1 Exercícios da aula passada
- 2 Comando `do-while`
- 3 Exemplos**
- 4 Uso do laço
- 5 Exercícios

Exemplo 1

Faça um programa que leia as notas da primeira prova de BCC201 e calcule e imprima a média das notas.

Considere que o número de alunos é desconhecido. Utilize como critério de parada do programa uma nota negativa.

Exemplo 1 (usando while)

```
1  #include <stdio.h>
2
3  int main()
4  {
5      int alunos = 0;
6      double nota, soma = 0;
7
8      printf("Digite um nro negativo para sair ou a nota do aluno: ");
9      scanf("%lf", &nota);
10
11     while (nota >= 0) {
12         soma += nota;
13         alunos++;
14         printf("Digite um nro negativo para sair ou a nota do aluno: ");
15         scanf("%lf", &nota);
16     }
17     double media = soma / alunos;
18     printf("A média das notas é: %.2lf\n", soma);
19
20     return 0;
21 }
```

Exemplo 1 (usando do-while)

```
1  #include <stdio.h>
2
3  int main()
4  {
5      int alunos = 0;
6      double nota, soma = 0;
7
8      do {
9          printf("Digite um nro negativo para sair ou a nota do aluno: ");
10         scanf("%lf", &nota);
11         if (nota >= 0) {
12             soma += nota;
13             alunos++;
14         }
15     } while(nota >= 0);
16
17     double media = soma / alunos;
18     printf("A média das notas é: %.2lf\n", soma);
19
20     return 0;
21 }
```


Exemplo 2: Jogo de Adivinhar

Faça um jogo onde o usuário tem que acertar um número selecionada pelo programa. O jogo termina quando o número digitado estiver correto. Imprimir o número de tentativas até o acerto.

Exemplo 2: Jogo de Adivinhar

```
1 // Jogo adivinha usando do-while
2 #include <stdio.h>
3
4 int main()
5 {
6     int resp, adivinha = 10;
7     int tentativas = 0;
8
9     do {
10         printf("Digite um nro: ");
11         scanf("%d", &resp);
12         if (resp != adivinha)
13             printf("%d é incorreto, tente novamente!\n", resp);
14         tentativas++;
15     } while (resp != adivinha);
16
17     printf("%d é o nro CORRETO!\n", resp);
18     printf("Número de tentativas: %d\n", tentativas);
19     printf("FIM DO JOGO...\n");
20     return 0;
21 }
```

Exemplo 2: Jogo de Adivinhar

```
1 // Jogo adivinha usando while
2 #include <stdio.h>
3
4 int main()
5 {
6     int resp, adivinha = 10;
7     int tentativas = 1;
8
9     printf("Digite um nro: ");
10    scanf("%d", &resp);
11
12    while (resp != adivinha); {
13        printf("Digite um nro: ");
14        scanf("%d", &resp);
15        if (resp != adivinha)
16            printf("%d é incorreto, tente novamente!\n\n", resp);
17        tentativas++;
18    }
19    printf("%d é o nro CORRETO!\n", resp);
20    printf("Número de tentativas: %d\n", tentativas);
21    printf("FIM DO JOGO...\n");
22    return 0;
23 }
```

Exemplo 3: Jogo de Adivinhar

E se o usuário quiser jogar mais de uma vez?

```

1  #include <stdio.h>
2  #include <stdlib.h> // para incluir a função rand
3
4  int main()
5  {
6      int tentativas, adivinha, resp, novamente;
7
8      do {
9          tentativas = 0;
10         adivinha = rand() % 10;
11
12         do {
13             printf("Digite um nro: ");
14             scanf("%d", &resp);
15             if (resp != adivinha)
16                 printf("%d é incorreto, tente novamente!\n\n", resp);
17             tentativas++;
18         } while (resp != adivinha);
19
20         printf("%d é o nro CORRETO!\n", resp);
21         printf("Número de tentativas: %d\n", tentativas);
22         printf("Deseja jogar novamente? (0=NAO, 1=SIM)\n");
23         scanf("%d", &novamente);
24     } while (novamente == 1);
25
26     return 0;
27 }

```

Aula de Hoje

- 1 Exercícios da aula passada
- 2 Comando `do-while`
- 3 Exemplos
- 4 Uso do laço**
- 5 Exercícios

Validação de entrada

Exemplo:

- 1 O usuário deve digitar um valor inteiro, positivo e par.
- 2 O programa continua após a digitação de um valor que atende essas restrições.
- 3 Observe que o usuário pode errar na digitação do valor..

Deve ser utilizado o `if`, `while` ou `do-while`?

Validando entrada

```
1  ...
2  printf("\nDigite um valor: ");
3  scanf("%d", &x);
4
5  while (x <= 0 || x % 2 != 0)
6  {
7      printf("\nERRO: Valor Inválido !");
8      printf("\nDigite um valor: ");
9      scanf("%d", &x);
10 }
11
12 // o programa prossegue com um número que
13 // atende a restrição
```

Observação:

- Não se pode prever quantas vezes o usuário entrará com um valor incorreto;
- O comando `if` não nos atende neste caso...

Validando entrada

```
1  ...
2  do {
3      printf("\nDigite um valor: ");
4      scanf("%d", &x);
5
6      if (x <= 0 || x % 2 != 0)
7          printf("\nERRO: Valor Inválido !");
8  } while (x <= 0 || x % 2 != 0);
9
10 // o programa prossegue com um número que
11 // atende a restrição
```

Observação:

- Não se pode prever quantas vezes o usuário entrará com um valor incorreto;
- O comando `if` não nos atende neste caso...

Repetição do programa

O comando `do-while` é útil para implementar repetição do programa:

```
1  char op;  
2  do {  
3  
4      /* o código do programa entra aqui... */  
5  
6      printf("\nRepetir a execução (s/n)?");  
7      scanf("%d", &op);  
8  } while (op == 's' || op == 'S') // o programa aceita S maiúsculo tb
```

Em geral, o comando `do-while` é usado quando os <comandos> devem ser executados **no mínimo uma vez**.

Exemplo 4: Algoritmo de Euclides

Calcula o MDC (máximo divisor comum) entre dois inteiros.

Efetuar várias divisões até chegar a uma divisão exata. O divisor desta divisão é o MDC. Ex.: cálculo do $\text{mdc}(48,30)$. Regra prática:

1º) dividimos o número maior pelo número menor:

$$48 / 30 = 1 \text{ (com resto 18)}$$

2º) dividimos o divisor 30, que é divisor da divisão anterior, por 18, que é o resto da divisão, e assim sucessivamente:

$$30 / 18 = 1 \text{ (com resto 12)}$$

$$18 / 12 = 1 \text{ (com resto 6)}$$

$$12 / 6 = 2 \text{ (com resto zero - divisão exata)}$$

3º) O divisor da divisão exata é 6.

Então **$\text{mdc}(48, 30) = 6$** .

Exemplo 4: Algoritmo de Euclides

Deve ser utilizado `do-while` ou `while` para calcular o MDC?

Exemplo 4: Algoritmo de Euclides (I)

```
1  /*
2  * Calcula o MDC (máximo divisor comum) entre dois inteiros.
3  */
4  int mdc(int x, int y)
5  {
6      // criando variável para armazenar o resto
7      int resto;
8
9      // garantindo que x >= y
10     if (x < y)
11         troca(&x, &y); // função que troca os valores de x e y
12
13     resto = x % y; // calcula o resto
14     while (resto != 0) {
15         x = y; // x é atualizado
16         y = resto; // y armazena o MDC
17         resto = x % y; // recalcula o resto
18     }
19
20     return y;
21 }
```

Exemplo 4: Algoritmo de Euclides (II)

```
1  /*
2  * Calcula o MDC (máximo divisor comum) entre dois inteiros.
3  */
4  int mdc(int x, int y)
5  {
6      // criando variável para armazenar o resto
7      int resto;
8
9      // garantindo que x >= y
10     if (x < y)
11         troca(&x, &y); // função que troca os valores de x e y
12
13     do {
14         resto = x % y; // calcula o resto
15         x = y; // x armazena o MDC
16         y = resto; // atualiza o valor de y para o resto
17     } while (resto != 0);
18
19     return x;
20 }
```

Exemplo 4: Algoritmo de Euclides

Usando `while`

```
1  int mdc(int x, int y)
2  {
3      if (x < y)
4          troca(&x, &y);
5
6      int resto = x % y;
7
8      while (resto != 0) {
9          x = y;
10         y = resto;
11         resto = x % y;
12     }
13
14     return y;
15 }
```

Usando `do-while`

```
1  int mdc(int x, int y)
2  {
3      if (x < y)
4          troca(&x, &y);
5
6      int resto;
7
8      do {
9          resto = x % y;
10         x = y;
11         y = resto;
12     } while (resto != 0);
13
14     return x;
15 }
```

Aula de Hoje

- 1 Exercícios da aula passada
- 2 Comando `do-while`
- 3 Exemplos
- 4 Uso do laço
- 5 Exercícios**

Exercício

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

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  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.



Perguntas?