

Profa. Michele Fúlvia Angelo

**PGCA 028 – Tópicos Especiais em Tecnologia  
Computacional I – Introdução à Programação de  
Computadores  
Aula 4**

Universidade Estadual de Feira de Santana

# Sumário

- Estruturas de Repetição (*Looping*) em C
- Simulação de Programas

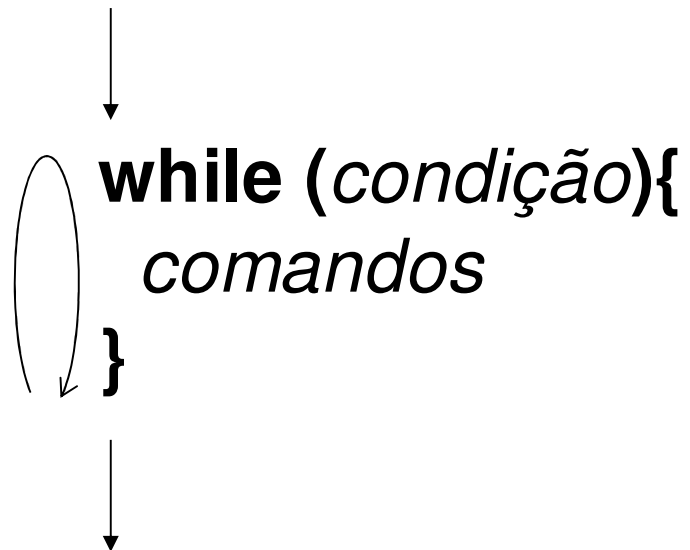
# Estruturas de Repetição (*Looping*)

- Estruturas utilizadas para repetir comandos.
- Repetição controlada por condições.
- "Vamos jogar quantas partidas de futebol pudermos até que chova ou que sejam seis da tarde"
  - Quantas partidas serão?
- O que vai ser repetido?
- Qual é a condição para parar de repetir?
- Qual é a condição para continuar a repetir?

# Estruturas de Repetição (*Looping*)

- **While**

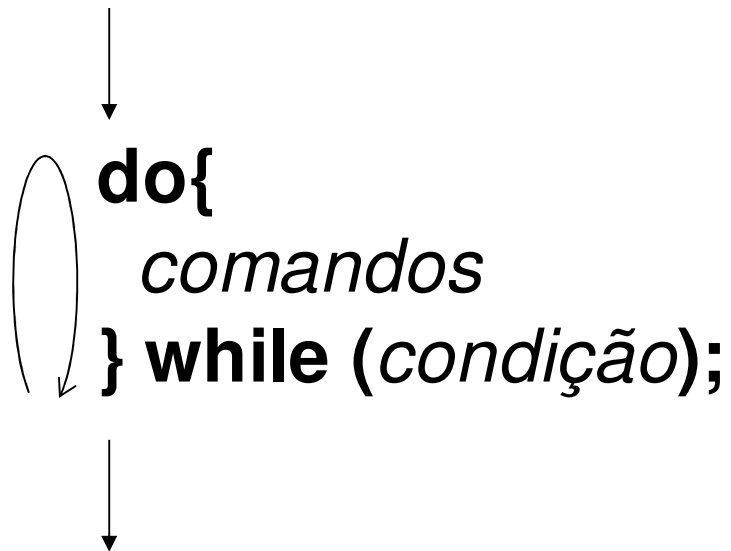
- A condição é de continuação e não de parada.



# Estruturas de Repetição (*Looping*)

- **Do ... while**

- A condição também é de continuação.



# Estruturas de Repetição (*Looping*)

## ■ While *versus* Do...while

Ambos fazem um número de repetições não previsto, descobrindo a partir de uma condição se deve ou não continuar:

- No while, a condição é verificada no começo e podemos ter zero ou mais repetições.
- No do-while, a condição é verificada no final e podemos ter 1 ou mais repetições.

# Estruturas de Repetição (*Looping*)

- Exemplo:

```
int main(){
    int partidas; /* número de partidas */
    char tempo; /* 's': sol; 'c': chuva */

    tempo='s'; /* Tempo inicialmente de sol */
    partidas=0;
    while (tempo != 'c'){
        partidas++;
        printf("Jogamos uma partida.\n");
        printf("Como esta o tempo?\n");
        scanf("%c",&tempo);
    }
    printf("Jogamos %d partidas.\n", partidas);}
```

# Estruturas de Repetição (*Looping*)

- Exemplo:

```
int main(){
    int partidas; /* número de partidas */
    char tempo; /* 's': sol; 'c': chuva */

    partidas=0;
    printf("Como está o tempo?\n");
    scanf("%c",&tempo);
    while (tempo != 'c'){
        partidas++;
        printf("Jogamos uma partida.\n");
        printf("Como está o tempo?\n");
        scanf("%c",&tempo);
    }
    printf("Jogamos %d partidas.\n", partidas);
}
```



# Estruturas de Repetição (*Looping*)

- Exemplo:

```
int main(){
    int partidas; /* número de partidas */
    char tempo; /* 's': sol; 'c': chuva */

    partidas=0;
    do {
        printf("Como está o tempo?\n");
        scanf("%c",&tempo);
        if(tempo!='c'){
            partidas++;
            printf("Jogamos uma partida.\n");
        }
    } while (tempo != 'c');
    printf("Jogamos %d partidas. \n", partidas);
}
```

# Estruturas de Repetição (*Looping*)

- Controlando a repetição:

```
int main(){
    int n;
    do {
        printf("Entre com um valor positivo ou 0 para sair:");
        scanf("%d", &n);
        if( (n % 2)==1 ){
            printf("%d eh impar\n",n);
        }
        else{
            printf("%d eh par\n",n);
        }
    } while(n !=0);
}
```

# Estruturas de Repetição (*Looping*)

- Validação:

```
int main(){
    int n;

    printf("Entre com um número no intervalo [10,50]: ");
    scanf("%d", &n);
    while ((n<10) || (n>50)){
        printf("ERRO: Número inválido.\n");
        printf("Entre com um número no intervalo [10,50]: ");
        scanf("%d", &n);
    }
    printf("\nO número lido foi: %d",n);
}
```

# Estruturas de Repetição (*Looping*)

- Validação:

```
int main(){
    int n;

    do {
        printf("Entre com um número no intervalo [10,50]: ");
        scanf("%d", &n);
        if ((n<10) || (n>50)) {
            printf("ERRO: Número inválido.\n");
        }
    } while ((n<10) || (n>50));
    printf("\nO número positivo lido foi: %d",n);
}
```

# Estruturas de Repetição (*Looping*)

- Repetição de seleção de opções:

```
int main(){
    char Opcao;
    do {
        printf("Entre com uma opção do menu abaixo: \n");
        printf("[1] - Inserir dados de aluno.\n");
        printf("[2] - Remover aluno do cadastro.\n");
        printf("[3] - Alterar os dados de um aluno.\n");
        printf("[4] - Sair do sistema de cadastro.\n");
        printf("Opcao: ");
        scanf("%c",&Opcao);
        switch(Opcao){
            case '1' : printf("Aqui insere aluno."); break;
            case '2' : printf("Aqui remove aluno."); break;
            case '3' : printf("Aqui altera Aluno."); break;
            case '4' : printf("Fim da execução do sistema."); break;
            default : printf("Opção inválida."); break;
        }
    } while (Opcao != '4');
}
```



# Estruturas de Repetição (*Looping*)

- Escreva um programa que mostre na tela todos os números de 1 a 100.

- Precisamos de algo equivalente a:

```
printf("%d",1);  
printf("%d",2);  
...  
printf("%d",100);
```

- Podemos fazer este comando repetir 100 vezes:

```
printf("%d",cont);
```

- Mas a variável **cont** precisar começar em 1 e aumentar até 100.

# Estruturas de Repetição (*Looping*)

```
#include<stdio.h>
int main()
{
    int cont;

    cont = 1; ← inicialização
    while (cont<=100) { ← condição
        printf("%d",cont);
        cont++; /* ou cont = cont + 1 */ ← atualização
    }
}
```

# Estruturas de Repetição (*Looping*)

- Simplificação do **While** com um **For**:
  - Evita erros agrupando no cabeçalho: inicialização, condição, e passo.
  - Organização: separação entre controle (lógica) de repetição e código a ser repetido.

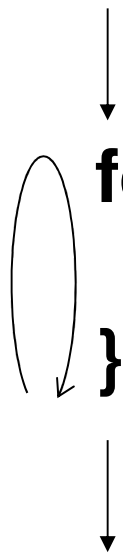
```
#include<stdio.h>
int main()
{
    int cont;

    for(cont=1; cont<=100; cont++){
        printf("%d",cont);
    }
}
```



# Estruturas de Repetição (*Looping*)

## ■ For



**for**(*inic; cond; passo*){  
    *comandos*  
}

```
for(n = 10; n<=20; n++){  
    printf("3 * %d = %d\n", n, 3*n);  
    printf("-----\n");  
}
```

# Estruturas de Repetição (*Looping*)

## ■ Generalização do For

```
#include<stdio.h>
int main() {
    int cont;

    for(cont=1; cont<=100; cont++){
        printf("%d",cont);
    }
}
```

```
#include<stdio.h>
int main() {

    int cont, ini, fim;

    printf("Digite inicio e fim:");
    scanf("%d %d",&ini, &fim);

    for(cont=ini; cont<=fim; cont++){
        printf("\n%d",cont);
    }
}
```

# Estruturas de Repetição (*Looping*)

- De trás para frente:

```
for(cont=100; cont>=1; cont--){  
    printf("%d",cont);  
}
```

- De dois em dois:

```
for(cont=1; cont<=100; cont+=2){  
    printf("%d",cont);  
}
```

# Estruturas de Repetição (*Looping*)

- For dentro de For:

```
int i,j;  
  
for(i=1; i<=5; i++){  
    for(j=1; j<=4; j++){  
        printf("%3d",i+j);  
    }  
    printf("\n");  
}
```

# Exercícios

1) Faça um programa que calcule o fatorial de um número qualquer digitado pelo usuário.

2) Faça um programa que imprima na tela a tabuada do 1 ao 5, na seguinte formatação:

$$1 \times 1 = 1$$

$$1 \times 2 = 2$$

...

3) Fazer um programa que leia  $n$  números reais positivos e informe qual é o maior deles. O valor de  $n$  deve ser solicitado no início.

# Simulação de Programas

- Como verificar o funcionamento do programa sem utilizar o computador?
  - Reproduzindo o comportamento do computador
  - Também conhecido como teste de mesa
  - Execução "mental" e memória no papel
- Podemos fazer no computador, com o auxílio de ferramentas de depuração (debug).
  - Execução passo a passo das linhas do programa
  - Observação dos valores das variáveis
  - Escolha de pontos de parada

# Simulação de Programas

- Como simular?
  - Monte uma tabela com uma coluna para cada variável relevante
  - Coloque os valores iniciais das variáveis
  - Execute cada linha do programa e anote as mudanças nas variáveis

Variavel1		Variável2		...		VariavelN
-----	+	-----	+	...		-----
3.14		lixo		...		'A'
				...		
				...		
				...		

# Simulação de Programas

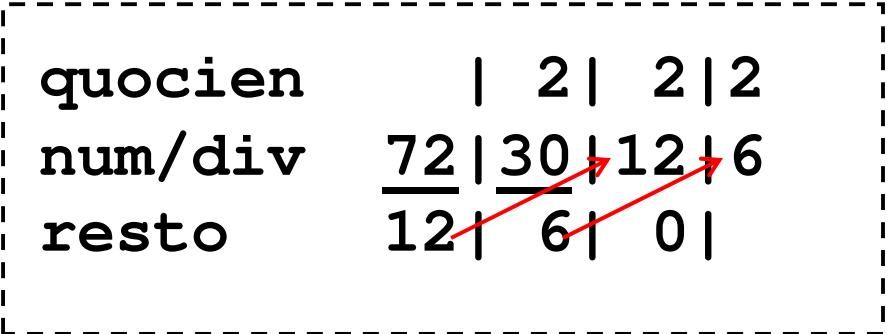
```
int main(){
    int num,den,r,n1,n2;

    scanf("%d %d",&n1,&n2);
    num = n1;
    den = n2;

    do {
        r = num % den;
        num = den;
        den = r;
    } while(r !=0);

    printf("O MDC entre %d e %d eh %d.",n1,n2,num);
}
```

quocien		2	2	2
num/div	<u>72 </u>	<u>30 </u>	12	6
resto	12	6	0	





# Simulação de Programas

```
num = n1;  
den = n2; ←
```

```
do {  
  r = num % den;  
  num = den;  
  den = r;  
} while(r != 0);
```

num		den		r
---	+	---	+	---
72		30		?

# Simulação de Programas

```
num = n1;  
den = n2;  
  
do {  
  r = num % den; ←  
  num = den;  
  den = r;  
} while(r != 0);
```

num		den		r
---	+	---	+	---
72		30		?
				12

# Simulação de Programas

```
num = n1;  
den = n2;  
  
do {  
    r = num % den;  
    num = den; ←  
    den = r;  
} while(r != 0);
```

num		den		r
---	+	---	+	---
72		30		?
30				12

# Simulação de Programas

```
num = n1;  
den = n2;  
  
do {  
    r = num % den;  
    num = den;  
    den = r; ←  
} while(r != 0);
```

num		den		r
---	+	---	+	---
72		30		?
30		12		12

# Simulação de Programas

```
num = n1;  
den = n2;  
  
do {  
  r = num % den; ←  
  num = den;  
  den = r;  
} while(r != 0);
```

num		den		r
---	+	---	+	---
72		30		?
30		12		12
				6

# Simulação de Programas

```
num = n1;  
den = n2;  
  
do {  
    r = num % den;  
    num = den; ←  
    den = r;  
} while(r != 0);
```

num		den		r
---	+	---	+	---
72		30		?
30		12		12
12				6

# Simulação de Programas

```
num = n1;  
den = n2;  
  
do {  
    r = num % den;  
    num = den;  
    den = r; ←  
} while(r != 0);
```

num		den		r
---	+	---	+	---
72		30		?
30		12		12
12		6		6

# Simulação de Programas

```
num = n1;  
den = n2;  
  
do {  
  r = num % den; ←  
  num = den;  
  den = r;  
} while(r != 0);
```

num		den		r
---	+	---	+	---
72		30		?
30		12		12
12		6		6
				0



# Simulação de Programas

```
num = n1;  
den = n2;  
  
do {  
    r = num % den;  
    num = den; ←  
    den = r;  
} while(r != 0);
```

num		den		r
---	+	---	+	---
72		30		?
30		12		12
12		6		6
6				0

# Simulação de Programas

```
num = n1;  
den = n2;  
  
do {  
    r = num % den;  
    num = den;  
    den = r; ←  
} while(r != 0);
```

num		den		r
---	+	---	+	---
72		30		?
30		12		12
12		6		6
6		0		0

O MDC entre 72 e 30 eh 6.