Computação 1

Estruturas de Repetição

Prof. Luiz Fernando Carvalho

luizfcarvalho@utfpr.edu.br

Estruturas de Repetição

- Um estrutura de repetição permite executar um conjunto de instruções um determinado número de vezes

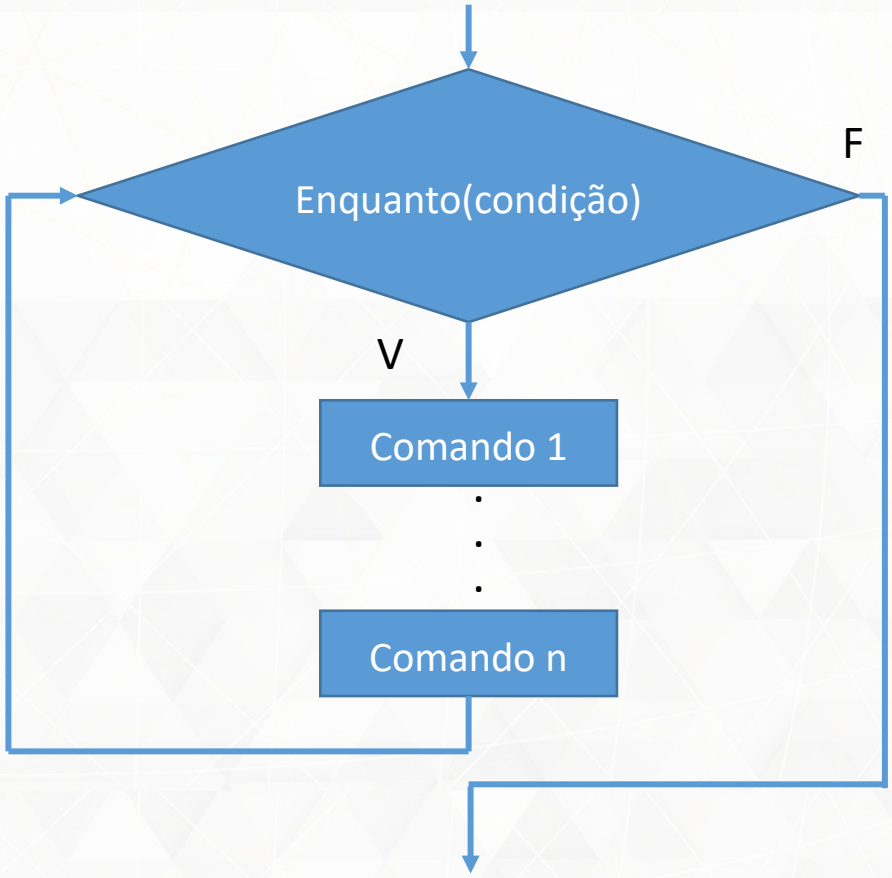
- A finalização da execução pode ser previamente determinada ou ser decorrente de uma ou mais condições se tornarem verdadeiras durante a execução do programa

- Estruturas de repetição da linguagem C:
  - `while` → enquanto ... faça
  - `do` ... `while` → faça ... Enquanto
  - `for` → para

Repetição – Teste no início

- Calculando a tabuada de algum número
- Enquanto... faça

```
int multiplicador = 0, resultado, num;

printf("Tabuada de qual numero: ");
scanf("%d", &num);

while(multiplicador <= 10)
{
     resultado = num * multiplicador;
     printf("%d", resultado);
     multiplicador = multiplicador + 1;
}
```



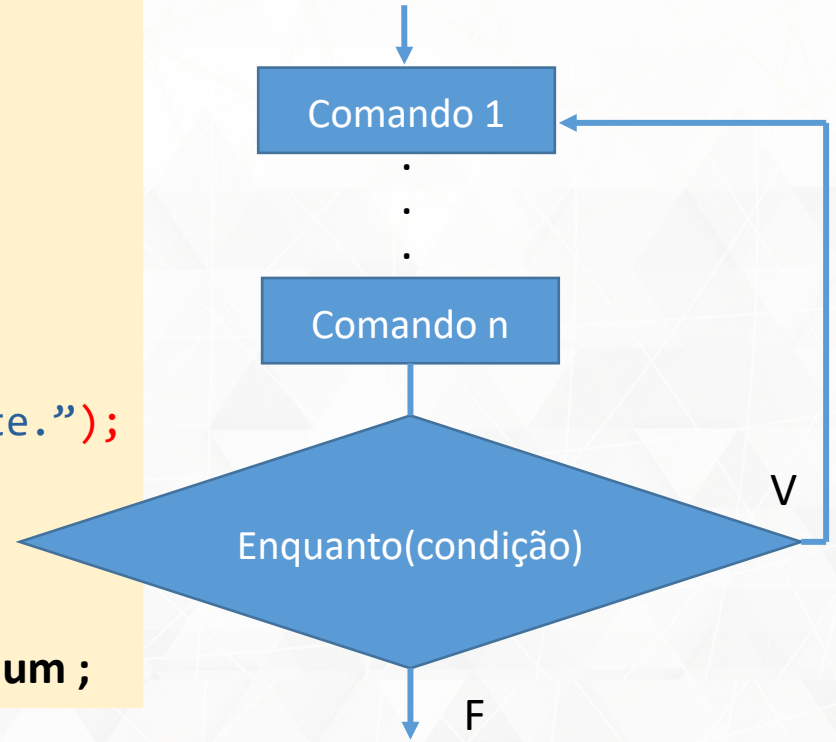
Em alguns programas os comandos dentro da instrução *while* podem não ser executados, uma vez que a condição é verificada antes que eles possam ser executados.

Repetição – Teste no final

- Verificar se o usuário digitou um valor positivo
- Faça... Enquanto

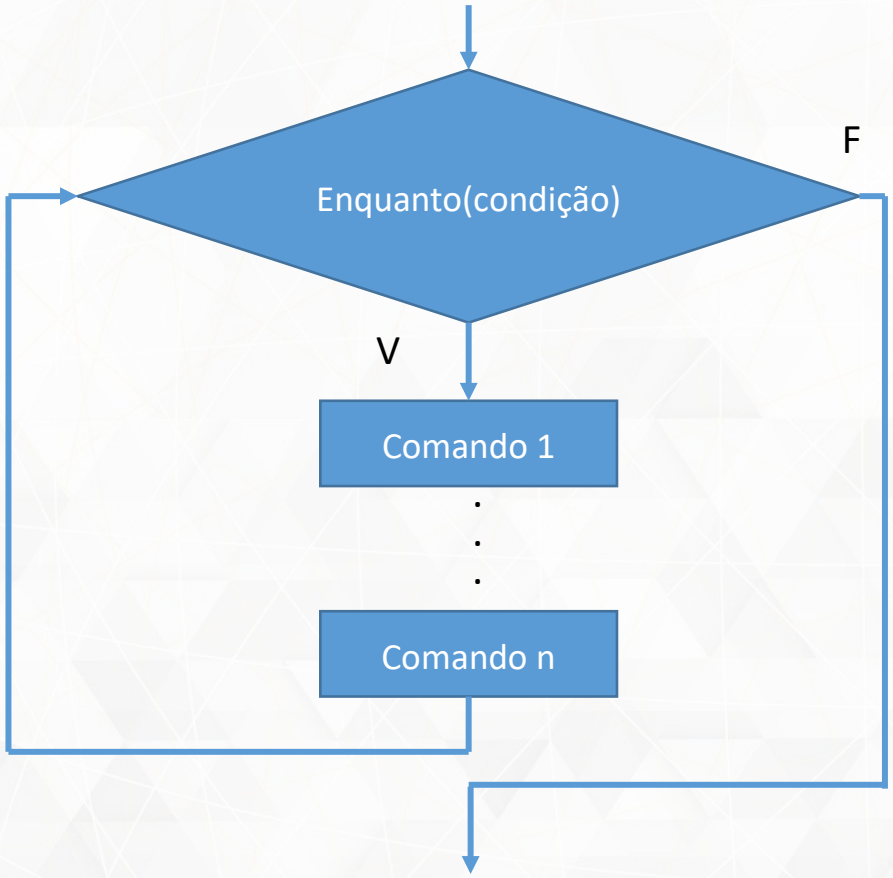
```
int num;

do
{
   printf("Forneca um inteiro positivo: ");
   scanf("%d", &num);

   if(num < 0)
     printf("Valor invalido. Tente novamente.");
}
while(num < 0);
```

Esse while tem um ;

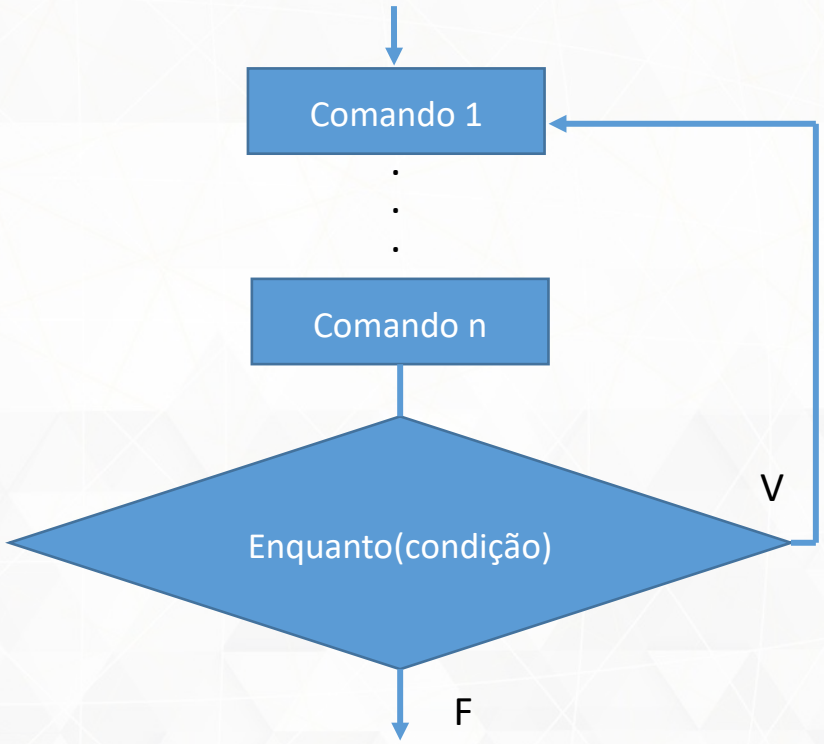


Os comandos dentro da instrução **do** são executadas pelo menos uma vez, já que o teste só é realizado no final.

Repetição - Comparação




```
while(condição){
      comandos;
}
```




```
do{
      comandos;
}
while(condição);
```

Repetição - Comparação

```
Enquanto(condição) faça
      comandos;

Fim-enquanto
```

```
Faça
    comandos;

Enquanto(condição);
```

```
while(condição){
      comandos;
}
```

```
do{
      comandos;
}
while(condição);
```

Declaração FOR

- Calculando a Tabuada com o FOR:

```
int multiplicador, resultado, num;

printf("Tabuada de qual numero: ");
scanf("%d", &num);

for(multiplicador = 0; multiplicador <= 10; multiplicador++){
      resultado = multiplicador * num;
      printf("%d", resultado);
}
```

Inicialização

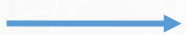
Condição

Incremento/
Decremento

```
for(inicialização; condição; incremento)
    comandos;
```

Atalhos para incremento e decremento

```
i++
++i
```



```
i = i + 1
```

```
i = 5;
a = i++;
```

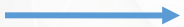
```
a = 5, i = 6
```

```
i = 5;
a = ++i;
```

```
a = 6, i = 6
```

Tomar cuidado com atalhos de incremento em operações de atribuição

```
i--
--i
```



```
i = i - 1
```

```
i = 5;
a = i--;
```

```
a = 5, i = 4
```

```
i = 5;
a = --i;
```

```
a = 4, i = 4
```

Atalhos para operações aritméticas

Equivale a ...

```
i += 5;
```

```
i = i + 5;
```

```
i -= 5;
```

```
i = i - 5;
```

```
i *= 5;
```

```
i = i * 5;
```

```
i /= 5;
```

```
i = i / 5;
```

Variáveis: acumulador e contador

- **ACUMULADOR (SOMADOR):** É uma variável que atua acumulando os valores a cada vez que o código é executado. Por exemplo, poderíamos implementar um somador num caixa de supermercado, acumulando na variável total todas as compras. Essa implementação é feita fazendo com que a variável total receba o seu próprio valor + o valor parcial de cada execução.
  - Ex.: `total = total + valor`
- **CONTADOR:** Os contadores acumulam seu próprio valor, acrescentando 1 a cada execução do programa. No mesmo exemplo anterior, o total_de_itens receberia seu próprio valor + 1 a cada item que passasse pelo caixa.
  - Ex.: `total_de_itens = total_de_itens + 1`

Acumulador e contador não são variáveis especiais. Esses nomes apenas indicam suas funções

Declaração FOR

- Assim como os comandos de seleção (**if-else**), as estruturas das controle de repetição **while** e **for** também podem ser aninhadas

```
for(x=0; x<10; x++){
   for(y=0; y<10; y++)
      printf("Valor de x=%i e y=%i", x, y);
}
```

Comparação entre WHILE e FOR

- Mesmo algoritmo, escrito de forma diferente

```
soma = 0;
c=0;
while(c<5){
    soma = soma + c;
    c++;
}
printf("Soma=%i", soma);
```

```
soma = 0;
      1     2     3
for(c=0; c<5; c++){
    soma = soma + c;
}
printf("Soma=%i", soma);
```

Quebrando o laço de repetição

- Break
  - Utilizado para sair abruptamente da estrutura de controle;
- Continue
  - Ignora o resto do bloco de dados de uma iteração, mas continua executando a estrutura de controle;

```
for(i=1; i <= 100; i++)
{
   if(i % 10 == 0)
      continue;
   else
      printf("%d", i);
}
```

1 2 3 4 5 6 7 8 9 11 12 ... 19 21 ... 29 31 99

```
for(i=1; i <= 100; i++)
{
   if(i % 10 == 0)
      break;
   else
      printf("%d", i);
}
```

1 2 3 4 5 6 7 8 9

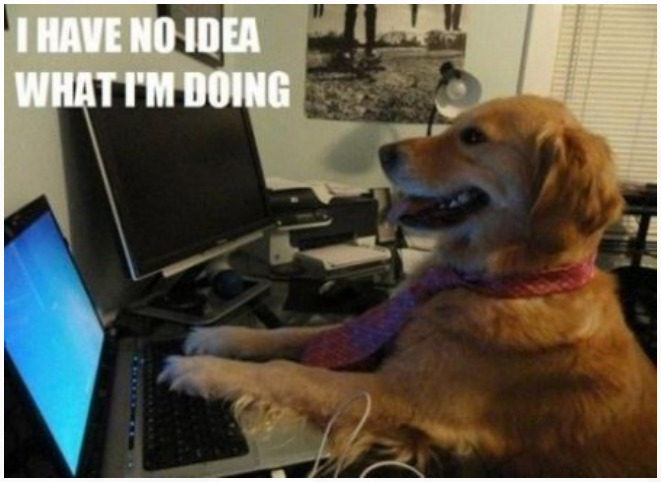
Quebrando o laço de repetição

- Break
  - Quando está dentro de um laço de repetição, o laço é imediatamente terminado e o programa passa a executar comandos logo após o laço;
- Continue
  - Em vez de terminar a execução do laço, continue força que ocorra a próxima iteração do laço, pulando qualquer código intermediário.
  - Para os laços **while** e **do-while**, o controle do programa passa para o teste condicional, assim que encontra o comando **continue**.
  - Para um **for**, o comando continue faz com que o teste condicional e a porção de incremento do laço sejam executados;

Loop Infinito I

- Podem ocorrer por erros durante a programação:

```
int multiplicador = 0, resultado, num;

printf(“Tabuada de qual numero: ”);
scanf(“%d”, &num);

while(multiplicador <= 10)
{
      resultado = num * multiplicador;
      printf(“%d”, resultado);
}
```



O que está errado?

Qual o valor de multiplicador ao longo das iterações?

Quando o loop irá acabar?

Loop Infinito II

- Podem ocorrer de propósito:

```
for(;;)
{
   printf(“Digite um numero inteiro: ”);
   scanf(“%d”, &n);
   if (n == 7)
   {
      printf(“Saindo do loop...\n”);
      break; //força a saída do loop
   }
   printf(“Numero: %d\n”, n);
}
printf(“Fim de programa”);
```

```
while(1)
{
   printf(“Digite um numero inteiro: ”);
   scanf(“%d”, &n);
   if (n == 7)
   {
      printf(“Saindo do loop...\n”);
      break; //força a saída do loop
   }
   printf(“Numero: %d\n”, n);
}
printf(“Fim de programa”);
```

Exercícios V

7) Faça um programa que leia 10 valores inteiros e escreva no final a soma dos valores lidos;

8) Faça um programa que receba a idade de dez pessoas, calcule e mostre a quantidade de pessoas com idade maior ou igual a 18 anos;

9) Faça um programa que receba dez números e mostre a quantidade entre 30 e 90;

10) Faça um programa para calcular o fatorial de um número inteiro *n*;

11) Calcule o resultado da série:

$$S = 1 + \frac{3}{2} + \frac{5}{3} + \frac{7}{4} + \cdots + \frac{99}{50}$$

Exercícios VI

12) A prefeitura de uma cidade fez uma pesquisa entre seus habitantes, coletando dados sobre salários e número de filhos. A prefeitura deseja saber:

- a média do salário da população;
- a média do número de filhos;
- o maior salário;
- a percentagem de pessoas com salários até R$ 400,00.

Finalizar a entrada de dados ao ser digitado um valor negativo em salário.

13) Elaborar um algoritmo para receber os seguintes dados dos funcionários de uma empresa: salário e sexo (M ou F). Computar e escrever:

- quantidade de funcionários do sexo masculino;
- quantidade de funcionários do sexo feminino;
- quantidade total de funcionários;
- média dos salários.
- Encerrar o processo ao ler o valor 0 como salário.