

CT62A

COMPUTAÇÃO 1

Aula 06 - Estruturas de Repetição

Profs. Rafael **Mantovani** e **Adalberto** Lazarini

Roteiro



- 1** Introdução
- 2** Comando ENQUANTO ... FAÇA
- 3** Comando FAÇA ... ENQUANTO
- 4** Comando PARA ...
- 5** Referências

Roteiro

- 1** Introdução
- 2** Comando ENQUANTO ... FAÇA
- 3** Comando FAÇA ... ENQUANTO
- 4** Comando PARA ...
- 5** Referências

Introdução



- Nós fazemos muitas coisas mais de uma vez:

Introdução

- Nós fazemos muitas coisas mais de uma vez:



Lavar o cabelo

ensaboa, enxágua, repete ...

Introdução

- Nós fazemos muitas coisas mais de uma vez:



Lavar o cabelo



Folhear um livro

ensaboa, enxágua, repete ...

Introdução

- Nós fazemos muitas coisas mais de uma vez:



Lavar o cabelo

ensaboa, enxágua, repete ...



Folhear um livro



Enviar msgs

Introdução

- Nós fazemos muitas coisas mais de uma vez:



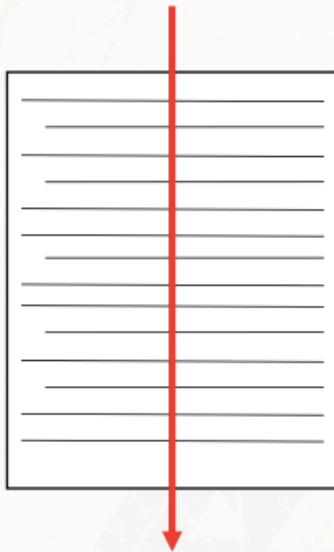
O que mais fazemos repetidamente?

ensaboa, enxágua, repete ...

Introdução

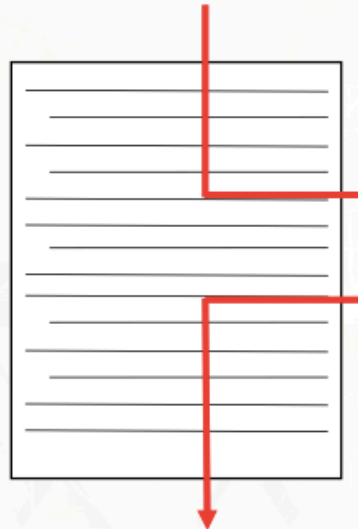
- Estruturas de Repetição:
 - Permite executar um conjunto de instruções um determinado número de vezes
 - Ao final da execução pode ser previamente determinada ou ser decorrente de uma ou mais condições se tornarem verdadeiras durante a execução do programa

Introdução



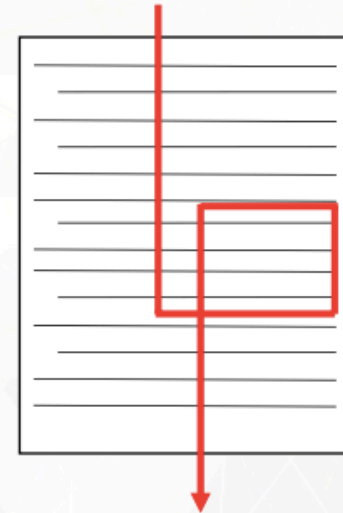
**Fluxo de execução
Sequencial**

*Comandos são executados um após
o outro*



**Fluxo de execução
com desvio**

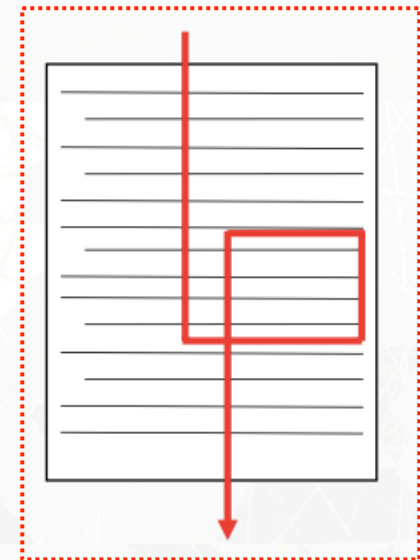
*Comandos são executados
dependendo do valor de uma
condição*



**Fluxo de execução
repetitivo**

*Comandos são executados de forma
repetida*

Introdução



**Fluxo de execução
Repetitivo**

*Comandos são executados de forma
repetida*

Laços de repetição



Vai um sorvetinho? :)

Laços de repetição

```
ENQUANTO (bolas > 0) {  
    IMPRIME(" Outra bola! ");  
    bolas = bolas - 1;  
}
```



Laços de repetição

Um laço de repetição começa com uma palavra-chave

1

```
ENQUANTO (bolas > 0) {  
    IMPRIME(" Outra bola! ");  
    bolas = bolas - 1;  
}
```



Laços de repetição

o laço usa uma expressão lógica como teste condicional (como nos ifs)

2

```
ENQUANTO (bolas > 0) {  
    IMPRIME(" Outra bola! ");  
    bolas = bolas - 1;  
}
```



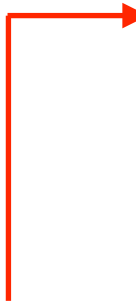
Laços de repetição

```
ENQUANTO (bolas > 0) {  
    IMPRIME(" Outra bola! ");  
    bolas = bolas - 1;  
}
```



se a condição for **VERDADEIRA**,
todo o bloco de código
é executado

Laços de repetição



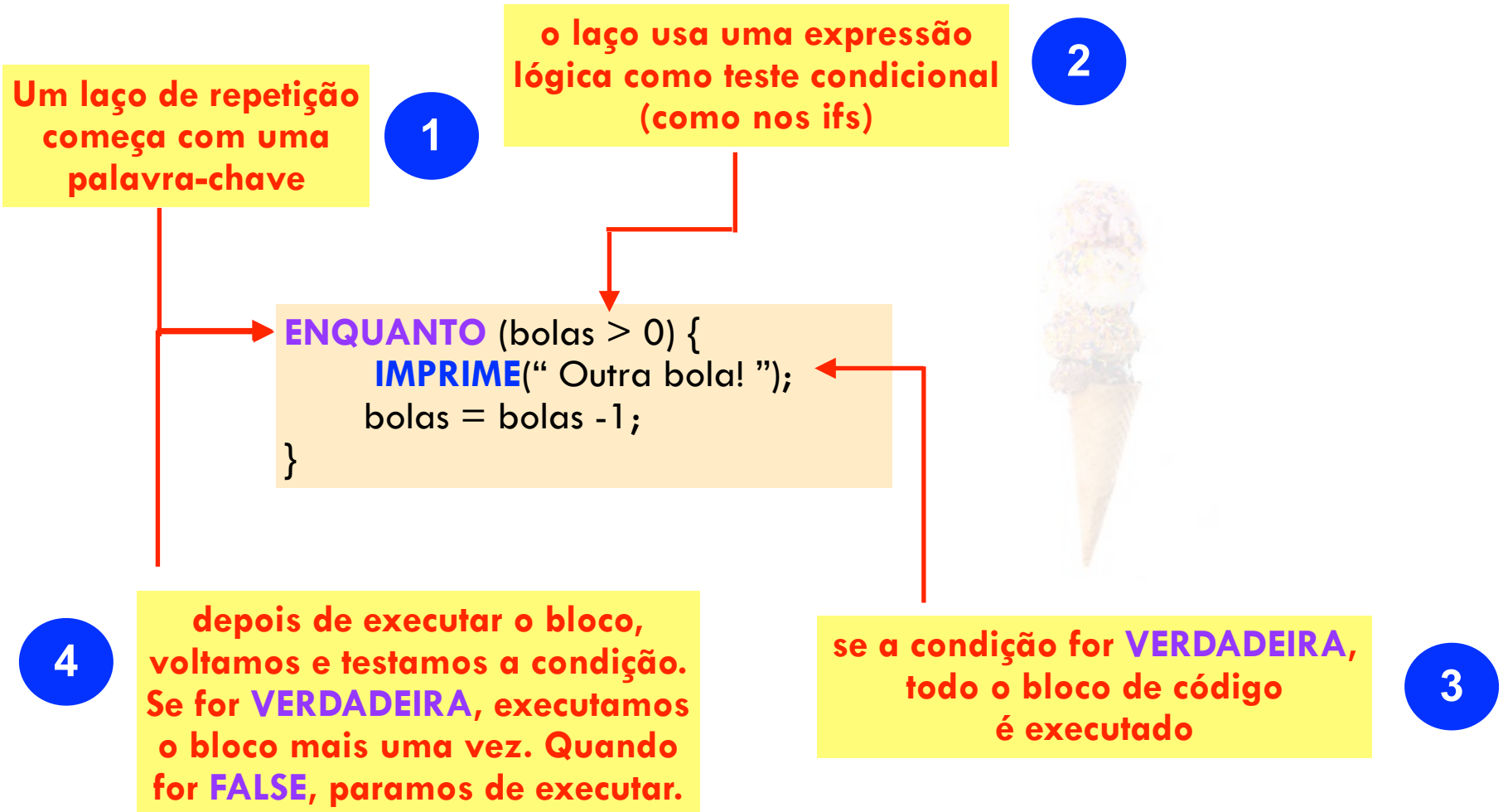
```
ENQUANTO (bolas > 0) {  
    IMPRIME(" Outra bola! ");  
    bolas = bolas - 1;  
}
```

4

depois de executar o bloco, voltamos e testamos a condição. Se for **VERDADEIRA**, executamos o bloco mais uma vez. Quando for **FALSE**, paramos de executar.



Laços de repetição



Como funciona um *loop*



Como funciona um *loop*

```
bolas = 4;  
ENQUANTO (bolas > 0) {  
    IMPRIME(" Outra bola! ");  
    bolas = bolas -1;  
}  
IMPRIME("Uma vida sem sorvete não é uma boa vida");
```



Como funciona um *loop*

- Vamos iniciar o laço. Atribuimos o valor 4 na variável **bolas**

```
bolas = 4;  
ENQUANTO (bolas > 0) {  
    IMPRIME(" Outra bola! ");  
    bolas = bolas -1;  
}  
IMPRIME("Uma vida sem sorvete não é uma boa vida");
```



Como funciona um *loop*

- A variável **bolas** é maior do que zero? É o que parece para nós!

```
bolas = 4;  
ENQUANTO (bolas > 0) {  
    IMPRIME(" Outra bola! ");  
    bolas = bolas -1;  
}  
IMPRIME("Uma vida sem sorvete não é uma boa vida");
```



Como funciona um *loop*

- Como a condição é **VERDADEIRA**, começamos a executar o bloco.

```
bolas = 4;  
ENQUANTO (bolas > 0) {  
    IMPRIME(" Outra bola! ");  
    bolas = bolas - 1;  
}  
IMPRIME("Uma vida sem sorvete não é uma boa vida");
```

```
>> Outra bola!
```



Como funciona um *loop*

- A declaração subtrai 1 do número de **bolas**, e atribui esse novo valor à variável

```
bolas = 4;  
ENQUANTO (bolas > 0) {  
    IMPRIME(" Outra bola! ");  
    bolas = bolas -1;  
}  
IMPRIME("Uma vida sem sorvete não é uma boa vida");
```

```
>> Outra bola!
```

1 bola já foi,
faltam 3!



Como funciona um *loop*

- Essa era a última declaração do bloco, então voltamos à condição. O código não muda, mas os **valores** das variáveis sim.

```
bolas = 4;  
→ ENQUANTO (bolas > 0) {  
    IMPRIME(" Outra bola! ");  
    bolas = bolas - 1;  
}  
IMPRIME("Uma vida sem sorvete não é uma boa vida");
```

```
>> Outra bola!
```



Como funciona um *loop*

- Reavaliando a condição, dessa vez **bolas** é 3.

```
bolas = 4;  
ENQUANTO (bolas > 0) {  
    IMPRIME(" Outra bola! ");  
    bolas = bolas -1;  
}  
IMPRIME("Uma vida sem sorvete não é uma boa vida");
```

```
>> Outra bola!
```



Como funciona um *loop*

- Mais uma vez, escrevemos a mensagem no terminal.

```
bolas = 4;  
ENQUANTO (bolas > 0) {  
    IMPRIME(" Outra bola! ");  
    bolas = bolas - 1;  
}  
IMPRIME("Uma vida sem sorvete não é uma boa vida");
```

```
>> Outra bola!  
>> Outra bola!
```



Como funciona um *loop*

- A declaração subtrai 1 do número de **bolas**, e atribui esse novo valor à variável, que é 2

```
bolas = 4;  
ENQUANTO (bolas > 0) {  
    IMPRIME(" Outra bola! ");  
    bolas = bolas -1;  
}  
IMPRIME("Uma vida sem sorvete não é uma boa vida");
```



```
>> Outra bola!  
>> Outra bola!
```

Como funciona um *loop*

- Reavalia a condição novamente. Bolas é igual a 2, e ainda é maior do que zero.

```
bolas = 4;  
ENQUANTO (bolas > 0) {  
    IMPRIME(" Outra bola! ");  
    bolas = bolas -1;  
}  
IMPRIME("Uma vida sem sorvete não é uma boa vida");
```



```
>> Outra bola!  
>> Outra bola!
```

Como funciona um *loop*

- Mais uma vez, escrevemos a mensagem no terminal.

```
bolas = 4;  
ENQUANTO (bolas > 0) {  
    IMPRIME(" Outra bola! ");  
    bolas = bolas - 1;  
}  
IMPRIME("Uma vida sem sorvete não é uma boa vida");
```

```
>> Outra bola!  
>> Outra bola!  
>> Outra bola!
```



Como funciona um *loop*

- Mais uma vez, escrevemos a mensagem no terminal.

```
bolas = 4;  
ENQUANTO (bolas > 0) {  
    IMPRIME(" Outra bola! ");  
    bolas = bolas -1;  
}  
IMPRIME("Uma vida sem sorvete não é uma boa vida");
```

```
>> Outra bola!  
>> Outra bola!  
>> Outra bola!
```



Como funciona um *loop*

- E continua ...

```
bolas = 4;  
ENQUANTO (bolas > 0) {  
    IMPRIME(" Outra bola! ");  
    bolas = bolas - 1;  
}  
IMPRIME("Uma vida sem sorvete não é uma boa vida");
```

```
>> Outra bola!  
>> Outra bola!  
>> Outra bola!
```



Como funciona um *loop*

- E continua ...

```
bolas = 4;  
ENQUANTO (bolas > 0) {  
    IMPRIME(" Outra bola! ");  
    bolas = bolas - 1;  
}  
IMPRIME("Uma vida sem sorvete não é uma boa vida");
```

```
>> Outra bola!  
>> Outra bola!  
>> Outra bola!  
>> Outra bola!
```



Como funciona um *loop*

- E continua ...

```
bolas = 4;  
ENQUANTO (bolas > 0) {  
    IMPRIME(" Outra bola! ");  
    bolas = bolas -1;  
}  
IMPRIME("Uma vida sem sorvete não é uma boa vida");
```

```
>> Outra bola!  
>> Outra bola!  
>> Outra bola!  
>> Outra bola!
```

Como funciona um loop

- ... até que algo diferente aconteça. No caso, a variável **bolas** é 0, então nossa condição resulta em **FALSO**. Dessa vez, pulamos o bloco

```
bolas = 4;  
ENQUANTO (bolas > 0) {  
    IMPRIME(" Outra bola! ");  
    bolas = bolas - 1;  
}  
IMPRIME("Uma vida sem sorvete não é uma boa vida");
```

```
>> Outra bola!  
>> Outra bola!  
>> Outra bola!  
>> Outra bola!
```

Como funciona um *loop*

- Escrevemos a mensagem após o bloco. Acabou!

```
bolas = 4;  
ENQUANTO (bolas > 0) {  
    IMPRIME(" Outra bola! ");  
    bolas = bolas - 1;  
}  
IMPRIME("Uma vida sem sorvete não é uma boa vida");
```

```
>> Outra bola!  
>> Outra bola!  
>> Outra bola!  
>> Outra bola!  
>> Uma vida sem sorvete não é uma boa vida
```

Exercício



- “Adivinhe o que estou pensando?”

Exercício

- “Adivinhe o que estou pensando?”

```
1. cor = "azul"
2. palpite = " "
3. chutes = 0

4. ENQUANTO _____ {
5.     IMPRIME("Adivinha qual cor estou pensando?")
6.     LER(palpite)
7.     _____
8. }

9. IMPRIME("Você Acertou! Isto levou ", chutes, "palpites.")
```

Exercício

- “Adivinhe o que estou pensando?”

```
1. cor = "azul"
2. palpite = " "
3. chutes = 0

4. ENQUANTO _____ {
5.     IMPRIME("Adivinha qual cor estou pensando?")
6.     LER(palpite)
7.     _____
8. }

9. IMPRIME("Você Acertou! Isto levou ", chutes, "palpites.")
```

Como tornar o programa correto?

Roteiro

- 1 Introdução
- 2 Comando ENQUANTO ... FAÇA
- 3 Comando FAÇA ... ENQUANTO
- 4 Comando PARA ...
- 5 Referências

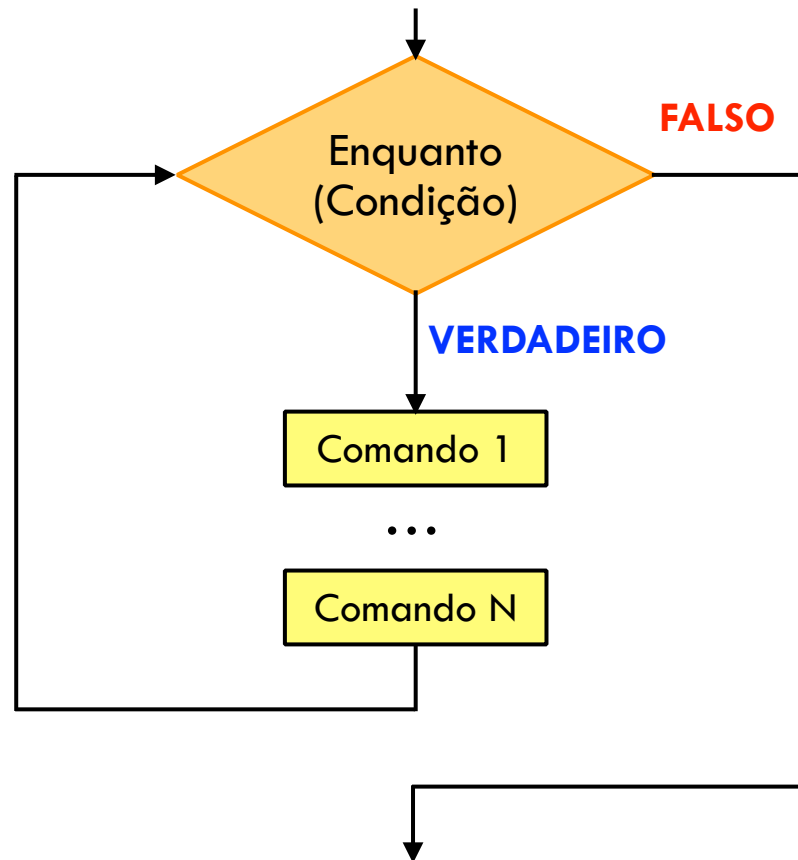
Comando Enquanto ... Faça



- Teste lógico é no início do laço

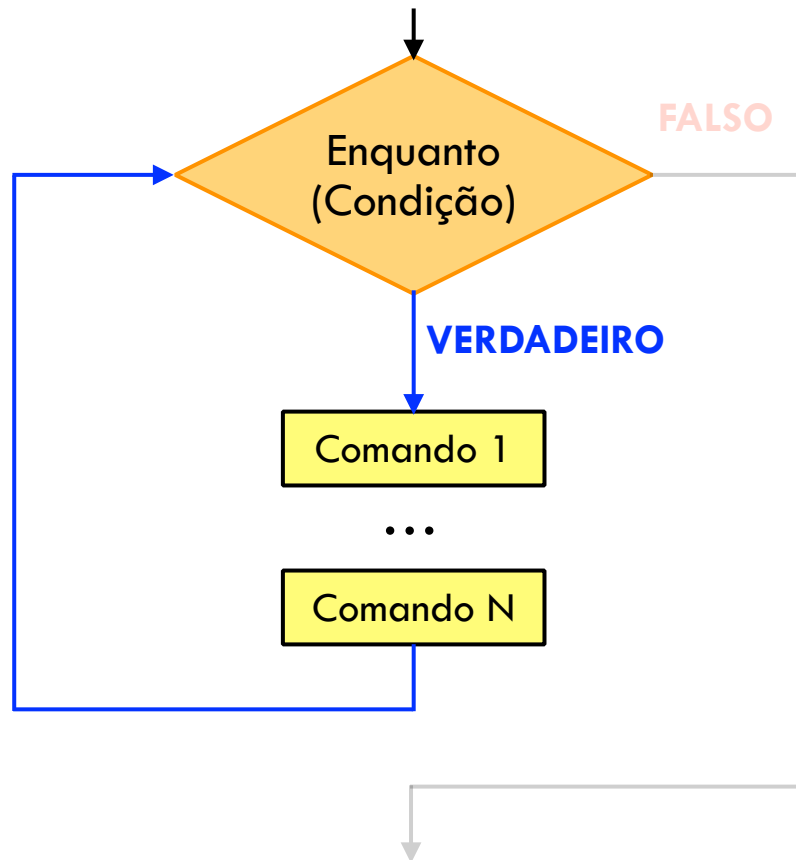
Comando Enquanto ... Faça

- Teste lógico é no início do laço



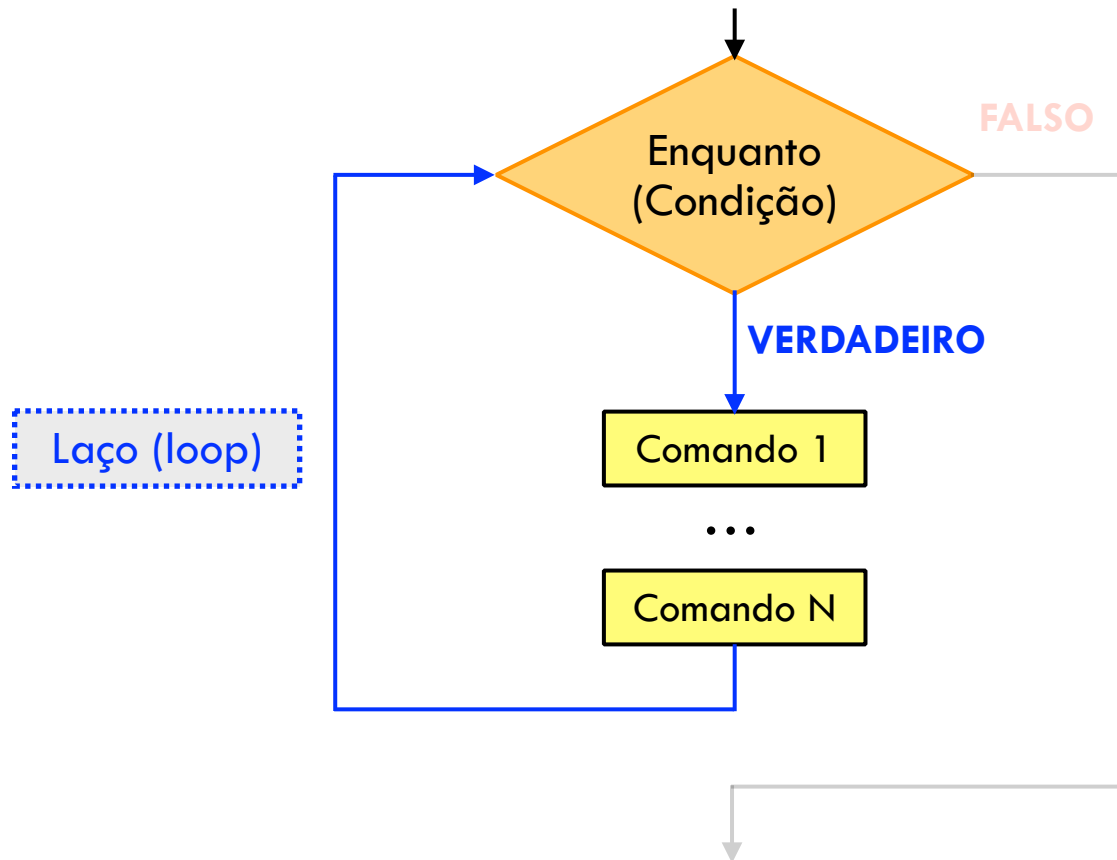
Comando Enquanto ... Faça

- Teste lógico é no início do laço



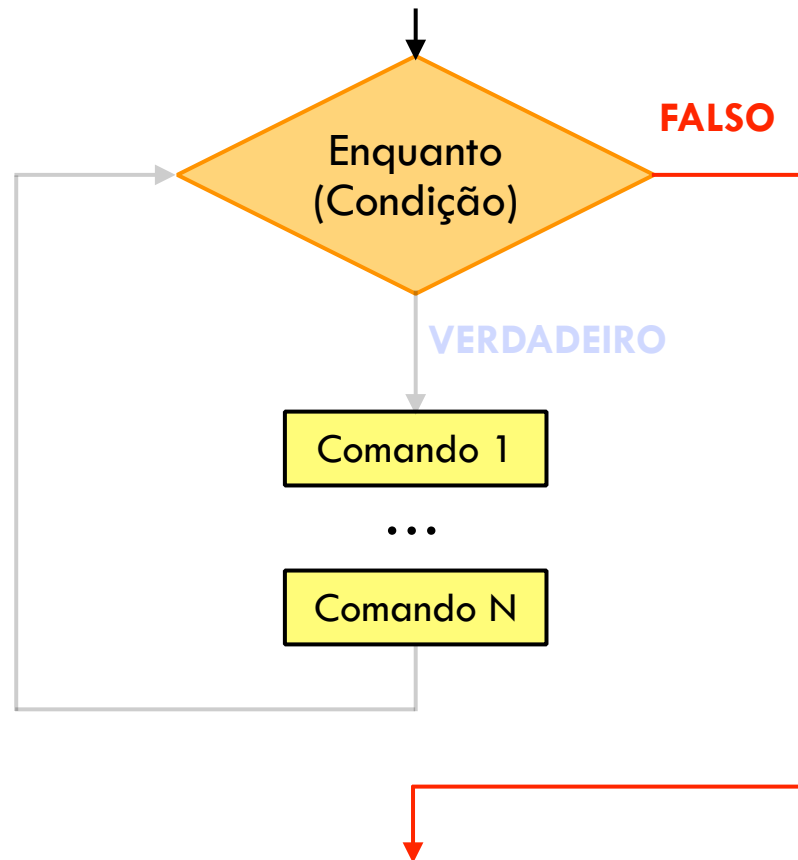
Comando Enquanto ... Faça

- Teste lógico é no início do laço



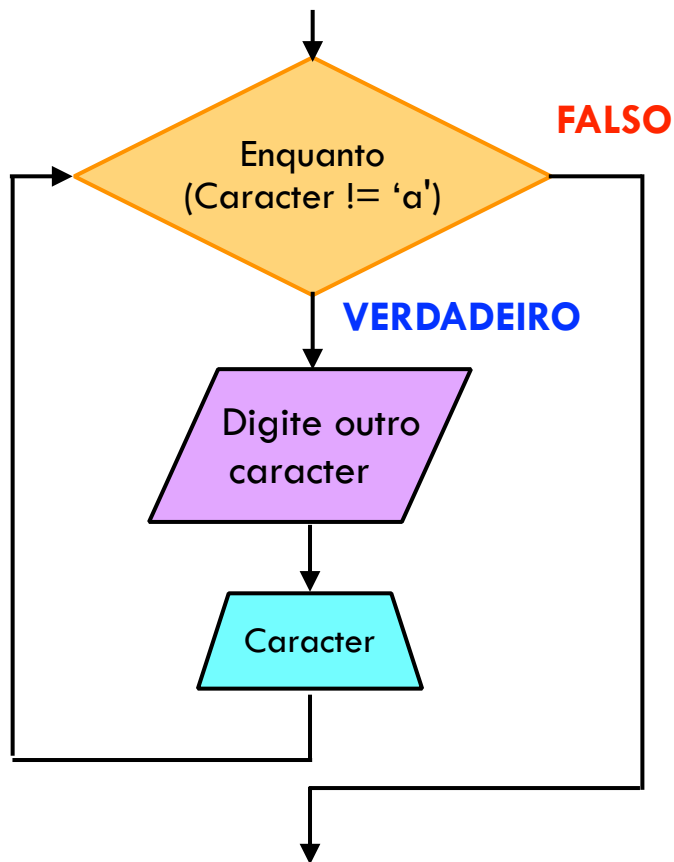
Comando Enquanto ... Faça

- Teste lógico é no início do laço



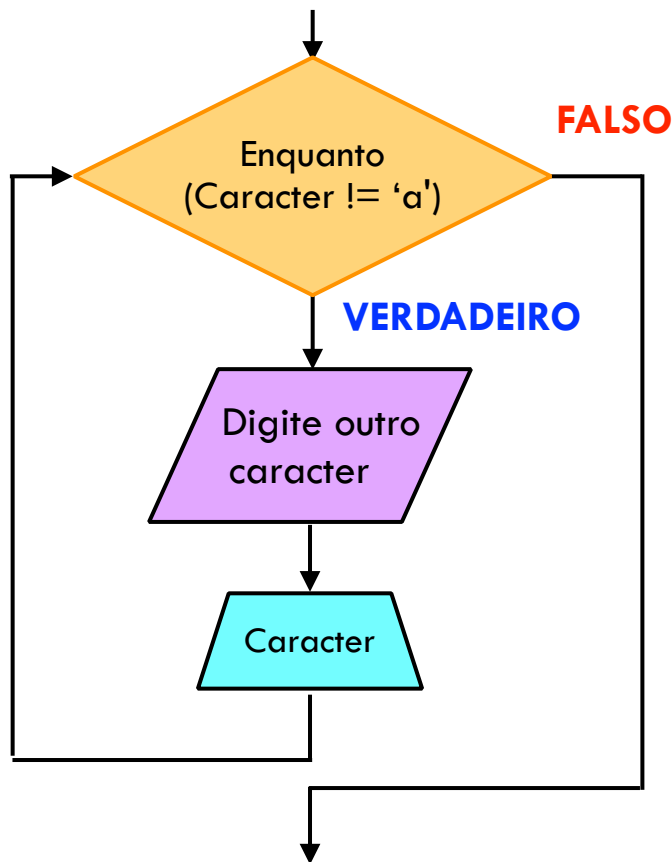
Comando Enquanto ... Faça

□ Exemplo:



Comando Enquanto ... Faça

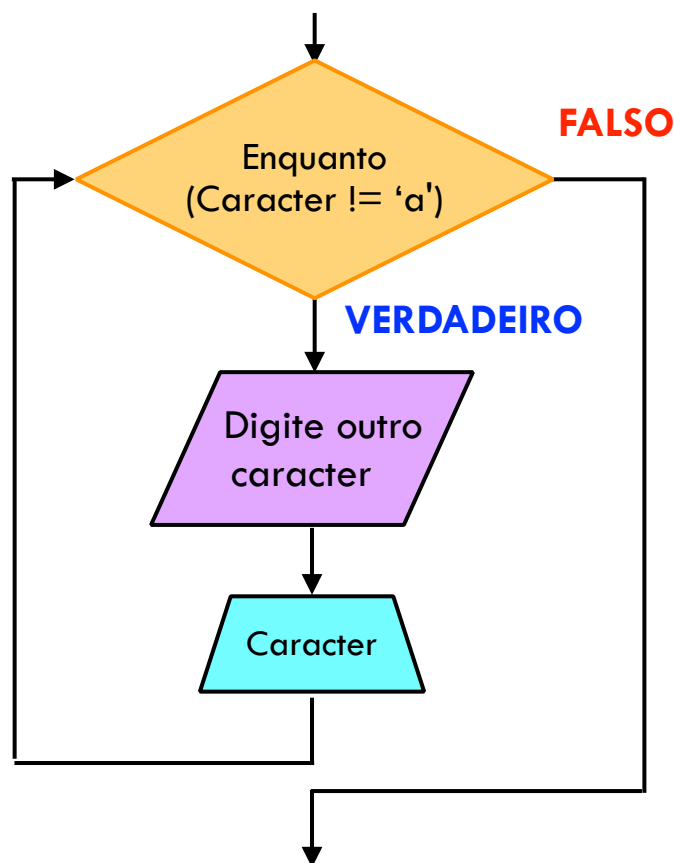
□ Exemplo:



```
ENQUANTO (condição) {  
    // comandos  
}
```

Comando Enquanto ... Faça

□ Exemplo:



```
ENQUANTO (condição) {  
    // comandos  
}
```

```
while (caracter != 'a') {  
    printf("Digite outro caracter.");  
    scanf("%c", &caracter);  
}
```


Exemplo



- Como calcular a tabuada de um número?

Exemplo

- Como calcular a tabuada de um número?



```
div {  
  div {  
    var atpos=inputs[1].indexOf("x");  
    var dotpos=inputs[1].lastIndexOf(".");  
    if (atpos<1 || dotpos<atpos+2 || dotpos>inputs[1].length-1) {  
      document.getElementById("errsmall").innerHTML += "Error: Invalid input format. Please use a number followed by 'x' and a decimal point followed by a number." + "  
    } else {  
      document.getElementById(div).innerHTML += "  
    }  
  }  
} else if (i==5) {  
  document.getElementById("errsmall").innerHTML += "Error: Invalid input format. Please use a number followed by 'x' and a decimal point followed by a number." + "  
}
```

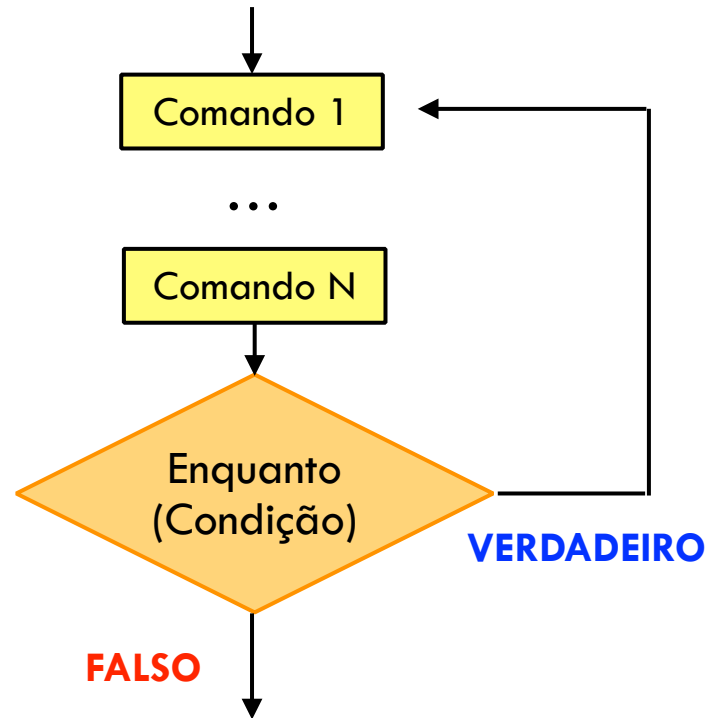
Hands on!

Roteiro

- 1 Introdução
- 2 Comando ENQUANTO ... FAÇA
- 3 Comando FAÇA ... ENQUANTO
- 4 Comando PARA ...
- 5 Referências

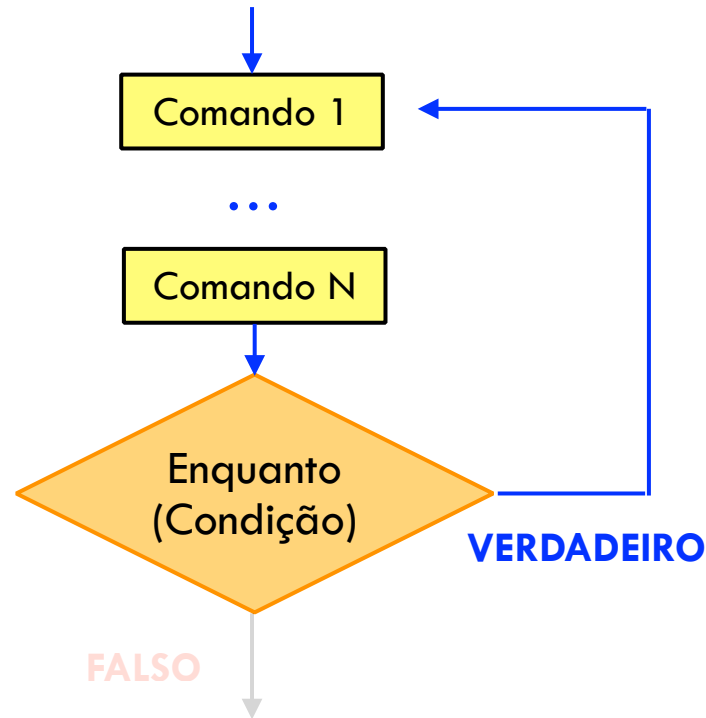
Comando Faça ... Enquanto

- Teste lógico é no fim do laço



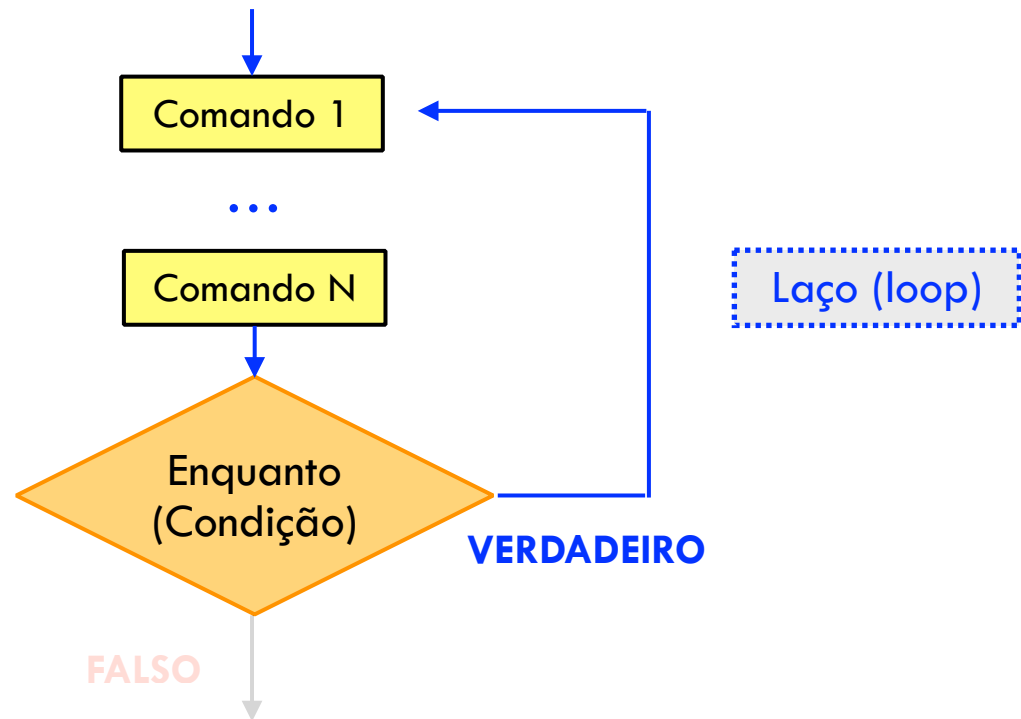
Comando Faça ... Enquanto

- Teste lógico é no fim do laço



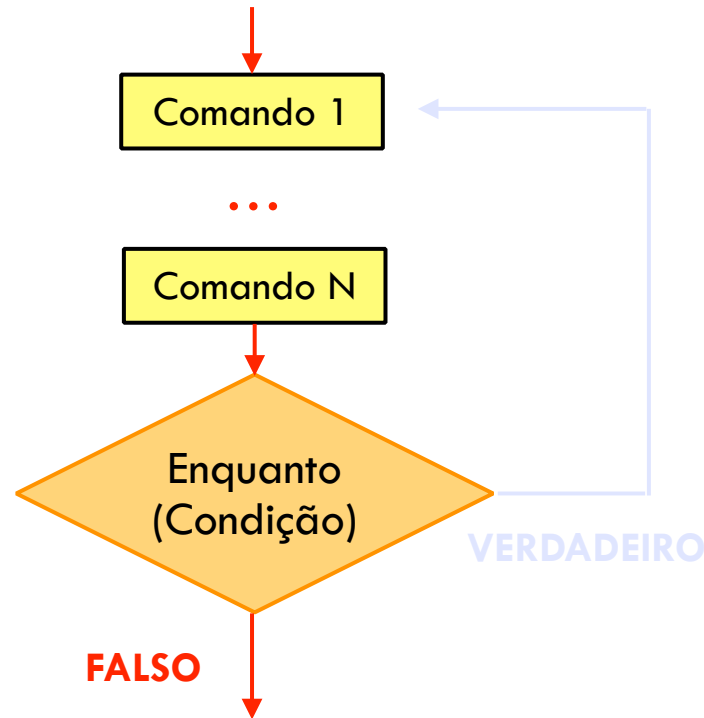
Comando Faça ... Enquanto

- Teste lógico é no fim do laço



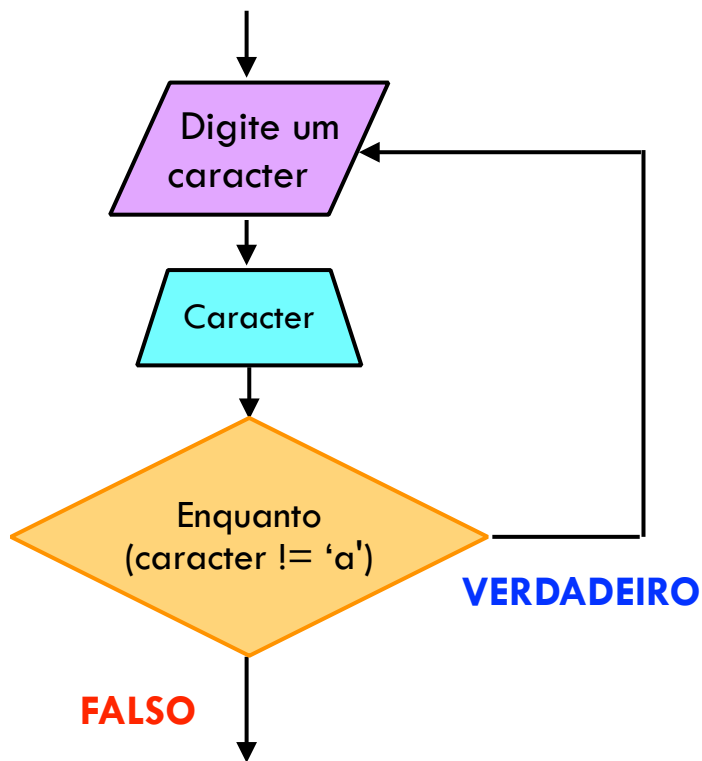
Comando Faça ... Enquanto

- Teste lógico é no fim do laço



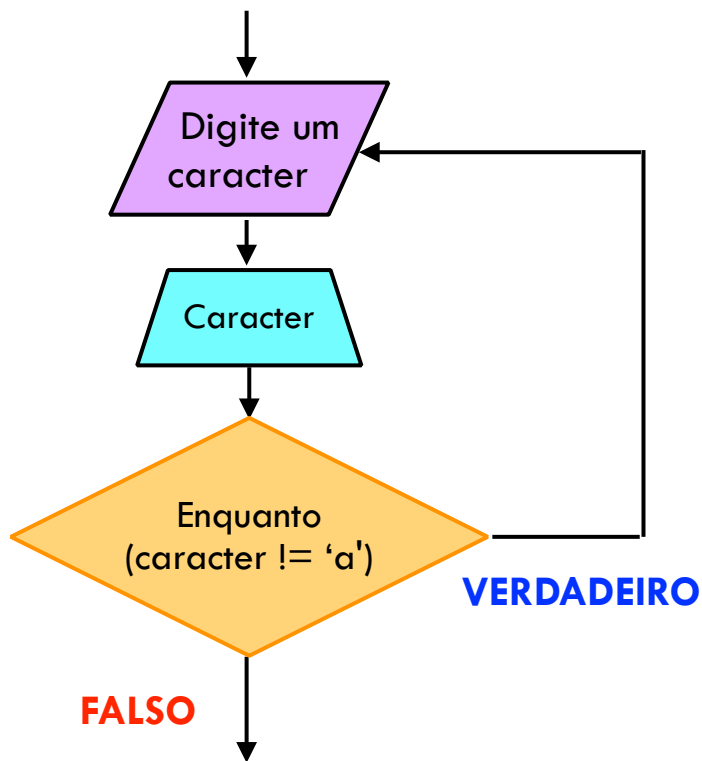
Comando Faça ... Enquanto

- Teste lógico é no fim do laço



Comando Faça ... Enquanto

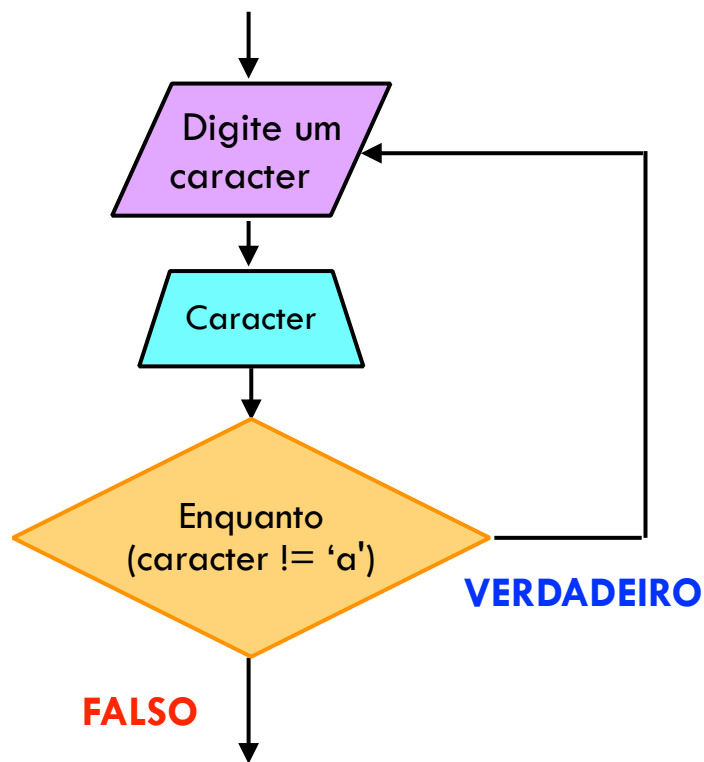
- Teste lógico é no fim do laço



```
FAÇA {  
  // comandos  
} ENQUANTO (condição);
```

Comando Faça ... Enquanto

- Teste lógico é no fim do laço



```
FAÇA {  
    // comandos  
} ENQUANTO (condição);
```

```
do {  
    printf("Digite um caracter.");  
    scanf("%c", &caracter);  
} while (caracter != 'a');
```

Exemplo



- Verificar se o usuário digitou um número positivo.

Exemplo

- Verificar se o usuário digitou um número positivo.



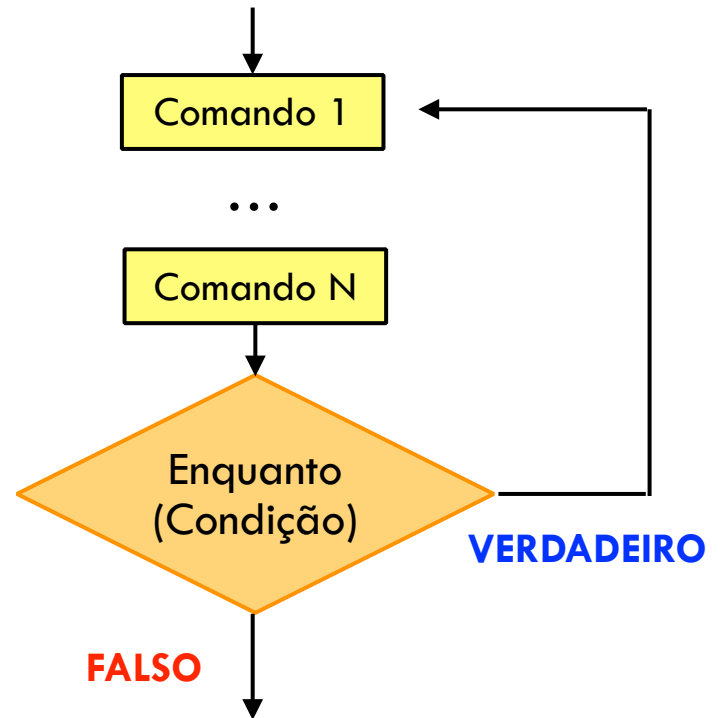
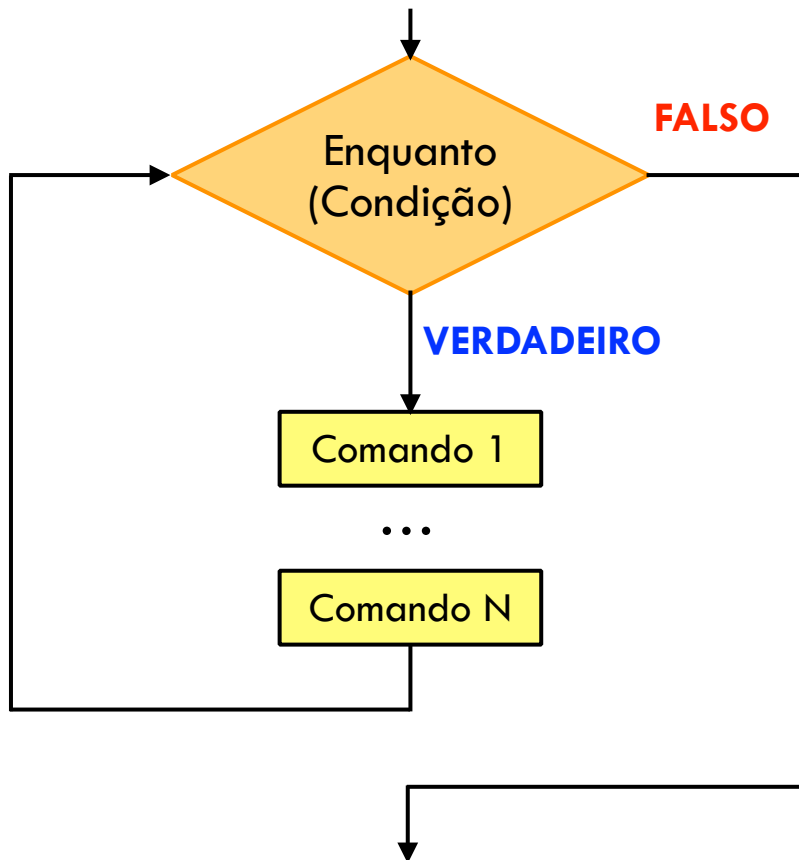
```
div {  
  div {  
    var atpos=inputs[1].indexOf("@");  
    var dotpos=inputs[1].lastIndexOf(".");  
    if (atpos<1 || dotpos<atpos+2 || dotpos>inputs[1].length-1) {  
      document.getElementById("errEmail").innerHTML += "Email inválido  
    } else {  
      document.getElementById(div).innerHTML += "OK  
    }  
  }  
}  
if (i==5)
```

Hands on!

Exercícios

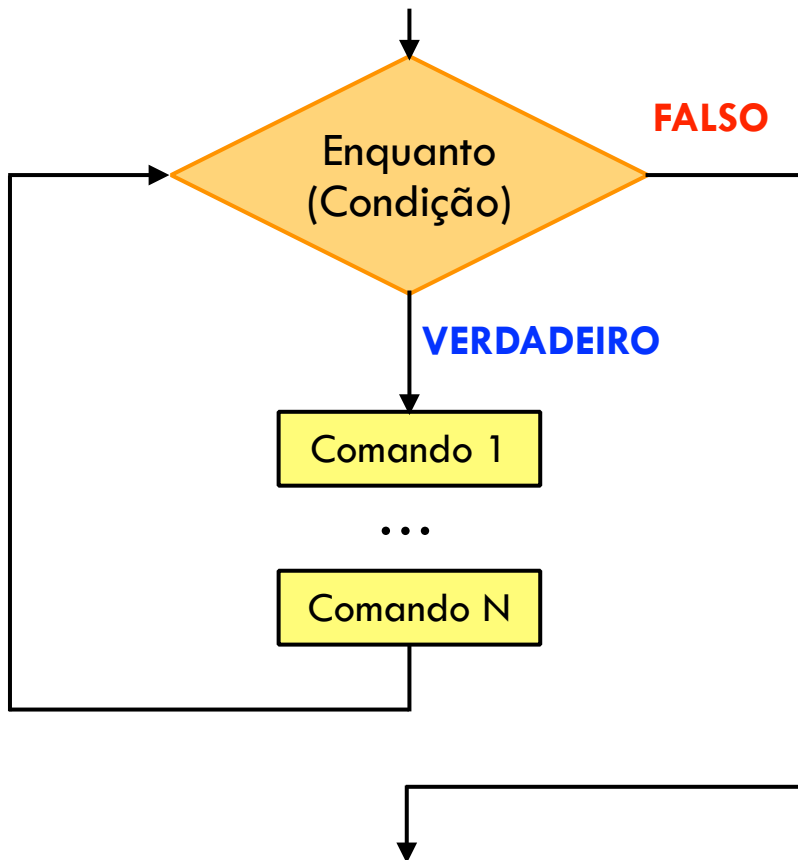
- 1 Faça um programa que leia 5 valores inteiros e escreva no final a soma dos valores lidos
- 2 Faça um programa que leia 5 valores inteiros e escreva no final a média dos valores lidos
- 3 Faça um programa que receba a idade de 10 pessoas, calcule e mostre a quantidade de pessoas com idade maior ou igual a 18 anos.
- 4 Faça um programa que leia 7 números e mostre a quantidade entre 30 e 90.

Comparativo

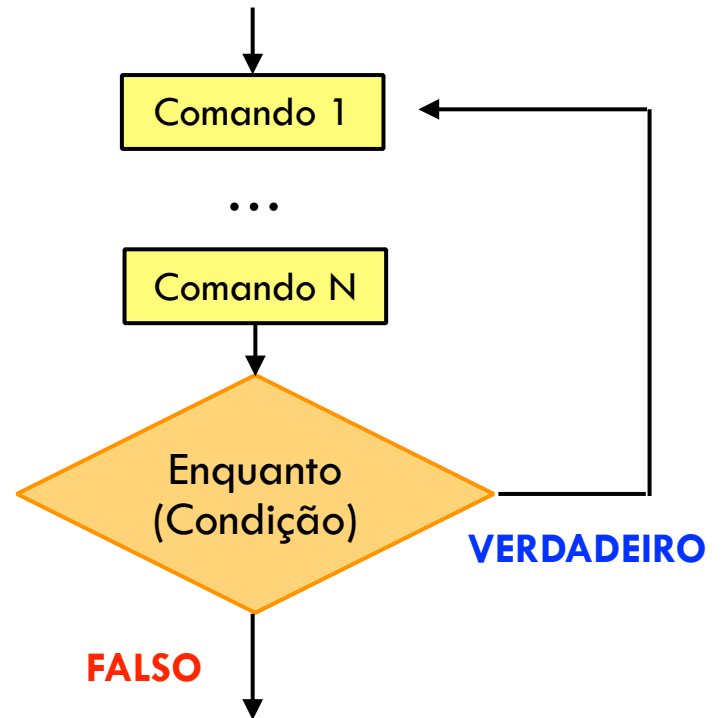


Comparativo

ENQUANTO .. FAÇA



FAÇA ... ENQUANTO



Comparativo

ENQUANTO .. FAÇA

```
ENQUANTO (condição) {  
    // comandos  
}
```

FAÇA ... ENQUANTO

```
FAÇA {  
    // comandos  
} ENQUANTO (condição);
```


Comparativo

ENQUANTO .. FAÇA

```
ENQUANTO (condição) {  
    // comandos  
}
```

```
while (caracter != 'A') {  
    printf("Digite outro caracter.");  
    scanf("%c", &caracter);  
}
```

FAÇA ... ENQUANTO

```
FAÇA {  
    // comandos  
} ENQUANTO (condição);
```

```
do {  
    printf("Digite um caracter.");  
    scanf("%c", &caracter);  
} while (caracter != 'A');
```

Roteiro



- 1 Introdução
- 2 Comando ENQUANTO ... FAÇA
- 3 Comando FAÇA ... ENQUANTO
- 4 Comando PARA ...
- 5 Referências

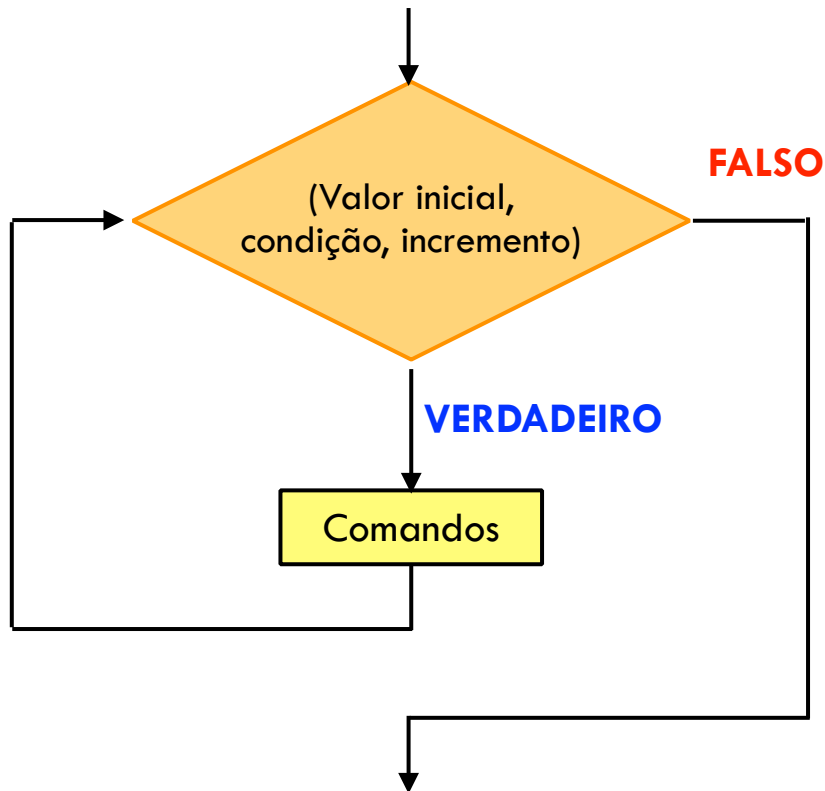
Comando Para



- Laço executa um número fixo de vezes (**N**)

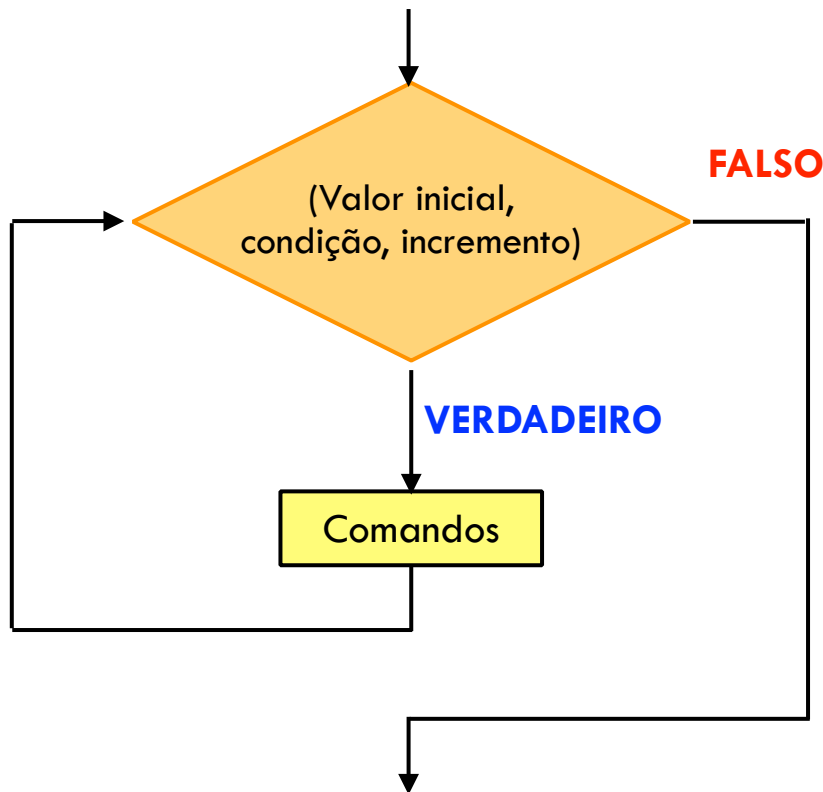
Comando Para

- Laço executa um número fixo de vezes (**N**)



Comando Para

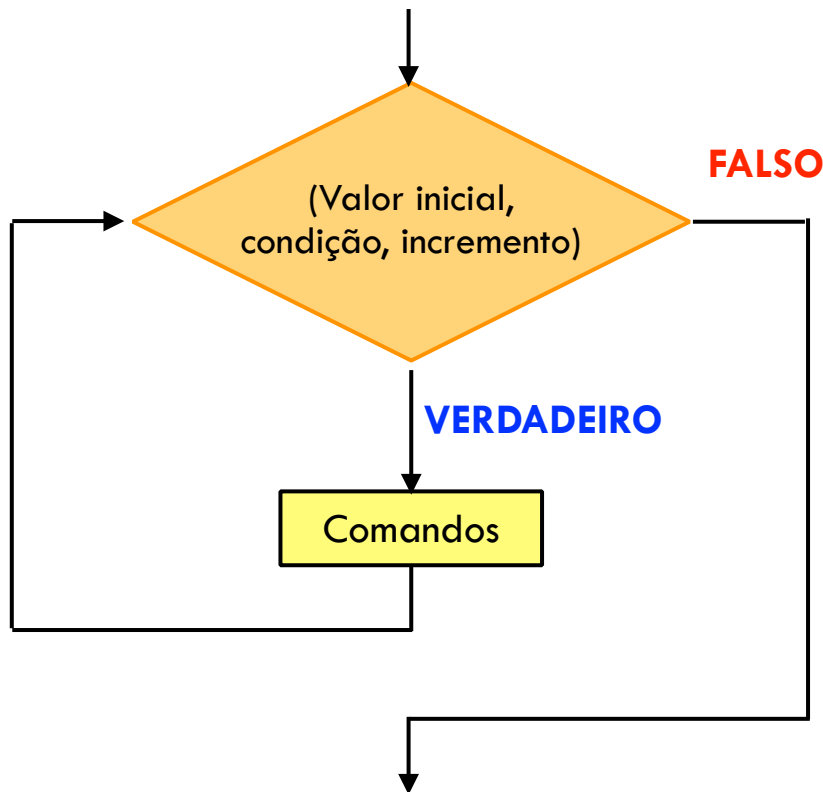
- Laço executa um número fixo de vezes (**N**)



```
PARA (valor inicial, condição, incremento) {  
    // bloco de comandos  
}
```

Comando Para

- Laço executa um número fixo de vezes (**N**)



```
PARA (valor inicial, condição, incremento) {  
    // bloco de comandos  
}
```

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

Comando Para



- Calculando a tabuada com FOR:

Comando Para

- Calculando a tabuada com FOR:

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

Inicialização	Condição	Incremento/ Decremento
<pre>for(multiplicador = 0; multiplicador <= 10; multiplicador++){ resultado = multiplicador * num; printf("%d", resultado); }</pre>		

Comando Para

- Atalhos para incremento/decremento:

`i++`
`++i` \longrightarrow `i = i + 1`

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

`a = 5, i = 6`

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

`a = 6, i = 6`

`i--`
`--i` \longrightarrow `i = i - 1`

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

`a = 5, i = 4`

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

`a = 4, i = 4`

Tomar cuidado com
atalhos de incremento
em operações de
atribuição

Comando Para

- Atalhos operações aritméticas

Equivale a ...	
<code>i += 5;</code>	<code>i = i + 5;</code>
<code>i -= 5;</code>	<code>i = i - 5;</code>
<code>i *= 5;</code>	<code>i = i * 5;</code>
<code>i /= 5;</code>	<code>i = i / 5;</code>

Laços Aninhados

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

Laços Aninhados

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

```
1. for (x=0; x<10; x++){  
2.     for (y=0; y<10; y++){  
3.         printf("Valores de x=%d e y=%d\n", x, y);  
4.     }  
5. }
```

Comparativo



- Mesmo algoritmo, escrito de forma diferente

Comparativo

- Mesmo algoritmo, escrito de forma diferente

```
1. soma = 0;
2. c = 0;
3. while (c<5) {
4.     soma = soma + c;
5.     c++;
6. }
7. printf("Soma=%i", soma);
```

Comparativo

- Mesmo algoritmo, escrito de forma diferente

```
1. soma = 0;
2. c = 0;
3. while (c<5) {
4.     soma = soma + c;
5.     c++;
6. }
7. printf("Soma=%i", soma);
```

```
1. soma = 0;
2. for (c=0; c<5; c++) {
3.     soma = soma + c;
4. }
5. printf("Soma=%i", soma);
```

Comparativo

- Mesmo algoritmo, escrito de forma diferente

```
1. soma = 0;
2. c = 0;
3. while (c<5) {
4.     soma = soma + c;
5.     c++;
6. }
7. printf("Soma=%i", soma);
```

```
1. soma = 0;
2. for (c=0; c<5; c++) {
3.     soma = soma + c;
4. }
5. printf("Soma=%i", soma);
```


Laços

- Conseguimos quebrar ou manter um laço executando:
 - **break**: utilizado para sair abruptamente de uma estrutura
 - **continue**: ignora o resto do bloco de comandos, mas continua executando a estrutura

Laços

- Conseguimos quebrar ou manter um laço executando:
 - **break**: utilizado para sair abruptamente de uma estrutura
 - **continue**: ignora o resto do bloco de comandos, mas continua executando a estrutura

```
1. for (x=1; x<=10; x++){  
2.     if (x%10 == 0){  
3.         continue;  
4.     } else {  
5.         printf("%d", i)  
6.     }  
7. }
```

```
1. for (x=1; x<=10; x++){  
2.     if (x%10 == 0){  
3.         break;  
4.     } else {  
5.         printf("%d", i)  
6.     }  
7. }
```

Loops Infinitos



- Podem ocorrer erros durante a programação:

Loops Infinitos

- Podem ocorrer erros durante a programação:

```
1.  int multiplicador = 0, resultado, num;
2.  printf("Tabuada de qual numero: ");
3.  scanf("%d", &num);

4.  while(multiplicador <= 10) {
5.      resultado = num * multiplicador;
6.      printf("%d", resultado);
7.  }
```

Loops Infinitos

- Podem ocorrer erros durante a programação:

```
1. int multiplicador = 0, resultado, num;  
2. printf("Tabuada de qual numero: ");  
3. scanf("%d", &num);  
  
4. while(multiplicador <= 10) {  
5.     resultado = num * multiplicador;  
6.     printf("%d", resultado);  
7. }
```

O que está errado?

Loops Infinitos



- As vezes, podem ser **intencionais!**

Loops Infinitos

- As vezes, podem ser **intencionais**!

```
1.  for (;;) {
2.      printf("Digite um numero inteiro: ");
3.      scanf("%d", &n);
4.      if(n == 7) {
5.          printf("Saindo do loop ... \n");
6.          break;
7.          // força a saída do loop
8.      }
9.      printf("Numero: %d\n", n);
10. }
11. printf("Fim de programa");
```

```
1.  while (1) {
2.      printf("Digite um numero inteiro: ");
3.      scanf("%d", &n);
4.      if(n == 7) {
5.          printf("Saindo do loop ... \n");
6.          break;
7.          // força a saída do loop
8.      }
9.      printf("Numero: %d\n", n);
10. }
11. printf("Fim de programa");
```

Loops Infinitos

- As vezes, podem ser **intencionais**!

```
1.  for (;;) {  
2.      printf("Digite um numero inteiro: ");  
3.      scanf("%d", &n);  
4.      if(n == 7) {  
5.          printf("Saindo do loop ... \n");  
6.          break;  
7.          // força a saída do loop  
8.      }  
9.      printf("Numero: %d\n", n);  
10. }  
11. printf("Fim de programa");
```

```
1.  while (1) {  
2.      printf("Digite um numero inteiro: ");  
3.      scanf("%d", &n);  
4.      if(n == 7) {  
5.          printf("Saindo do loop ... \n");  
6.          break;  
7.          // força a saída do loop  
8.      }  
9.      printf("Numero: %d\n", n);  
10. }  
11. printf("Fim de programa");
```


Exercícios



```
{
  var atpos=inputs[i].indexOf("@");
  var dotpos=inputs[i].lastIndexOf(".");
  if (atpos<1 || dotpos<atpos+2 || dotpos>inputs[i].length-1)
    document.getElementById('errmsg1').innerHTML += "Error: Invalid email address. ";
  else
    document.getElementById(div).innerHTML += "Valid email address. ";
  var atpos=inputs[i].indexOf("@");
  var dotpos=inputs[i].lastIndexOf(".");
  if (i==5)
    document.getElementById('errmsg1').innerHTML += "Error: Invalid email address. ";
}
```

Hands on!

Exercícios

- 5 Faça um programa que calcule o fatorial de um número inteiro N.
- 6 Faça um programa que leia 10 valores inteiros e escreva o menor valor lido e o maior valor lido.
- 7 Faça um programa que determine e mostre os cinco primeiros múltiplos de 3, considerando números maiores que 0.
- 8 Calcule o resultado da série:

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

Exercícios

9 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 porcentagem de pessoas com salários até R\$400,00.

Finalizar a entrada dos dados ao ser digitado um valor negativo.

Exercícios

- 10 **Final Boss:** melhore o jogo Pedra-Papel-Tesoura usando laços de repetição.



Exercícios

- 10** **Final Boss:** melhore o jogo Pedra-Papel-Tesoura usando laços de repetição.

- * **Solicitar uma jogada ao usuário até que ele forneça um valor válido para a jogada**
- * **Executar o jogo até que o usuário decida por encerrar o programa.**

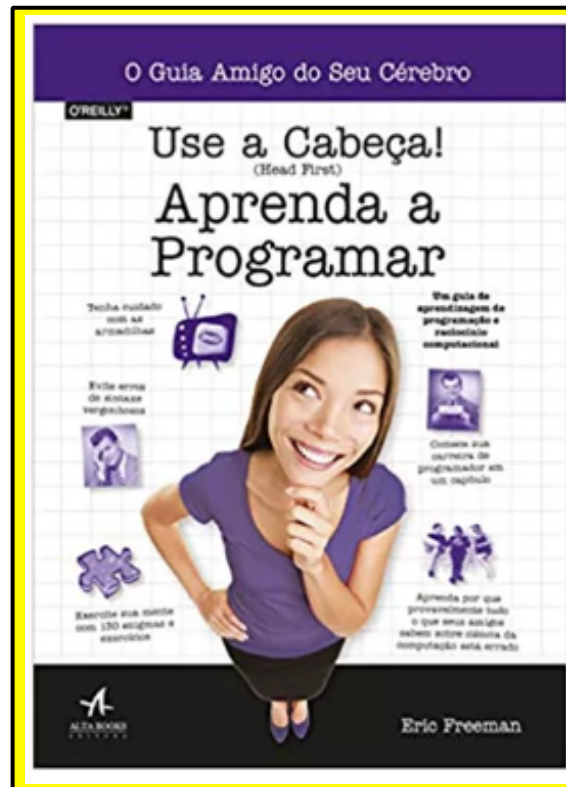
Roteiro

- 1 Introdução
- 2 Comando ENQUANTO ... FAÇA
- 3 Comando FAÇA ... ENQUANTO
- 4 Comando PARA ...
- 5 Referências

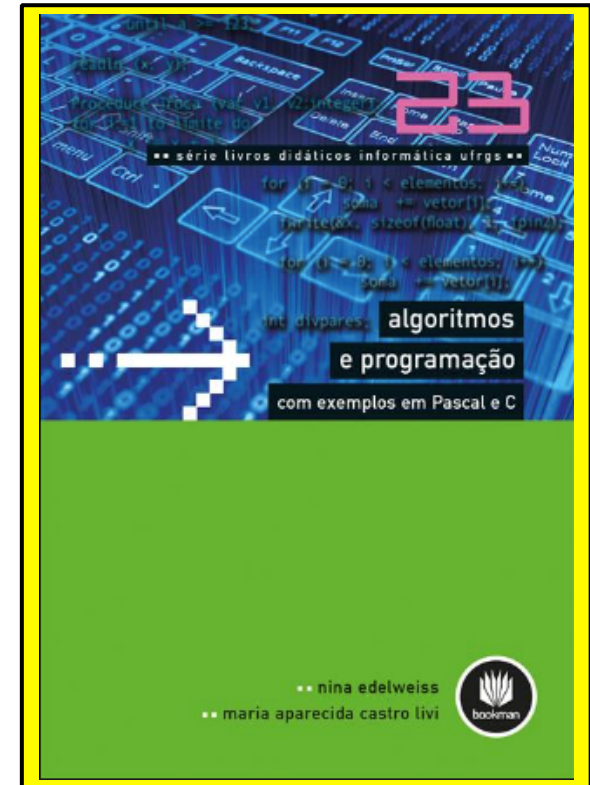
Referências sugeridas



[Souza et al, 2019]



[Freeman, 2019]



[Edelweiss & Livi, 2014]

Dúvidas?

Prof. Rafael G. **Mantovani**

rafaelmantovani@utfpr.edu.br

Prof. **Adalberto** Lazarini

adalbertoz@utfpr.edu.br