

CONSTRUÇÃO DE ALGORITMOS

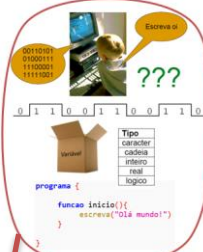
Bacharelado em Sistemas da Informação

Prof. Marco André Abud Kappel

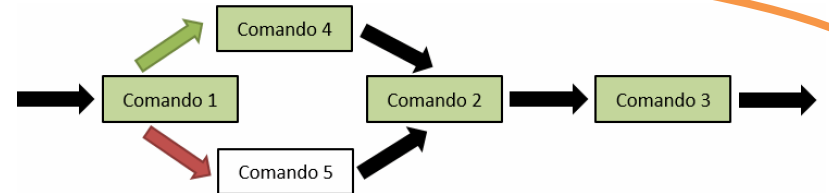
Aula 7 – Estruturas de repetição

Estruturas de repetição

• Anteriormente:



Hoje:
Estruturas de
repetição



```

se (condição){
    comando1
    comando2
    comando3
    ...
} senao {
    comando4
    comando5
    comando6
    ...
}
    
```



| Operador | Operação |
|----------|----------------|
| == | igual |
| > | maior |
| < | menor |
| >= | maior ou igual |
| <= | menor ou igual |

ou

| | | | | | | | | |
|------------|------------|------------|------------|------------|------------|------------|------------|------------|
| x | y | x & y | x y | x ^ y | x < y | x > y | x == y | x != y |
| falso | falso | falso | falso | falso | falso | falso | verdadeiro | falso |
| falso | verdadeiro | falso | verdadeiro | verdadeiro | falso | verdadeiro | falso | verdadeiro |
| verdadeiro | falso | falso | falso | verdadeiro | verdadeiro | falso | falso | verdadeiro |
| verdadeiro | verdadeiro | verdadeiro | verdadeiro | falso | falso | verdadeiro | verdadeiro | falso |

```

funcao tipo_de_retorno identificador(tipo1 par1, tipo2 par2, ...){
    bloco de comandos
    retorne algum_resultado
}
    
```

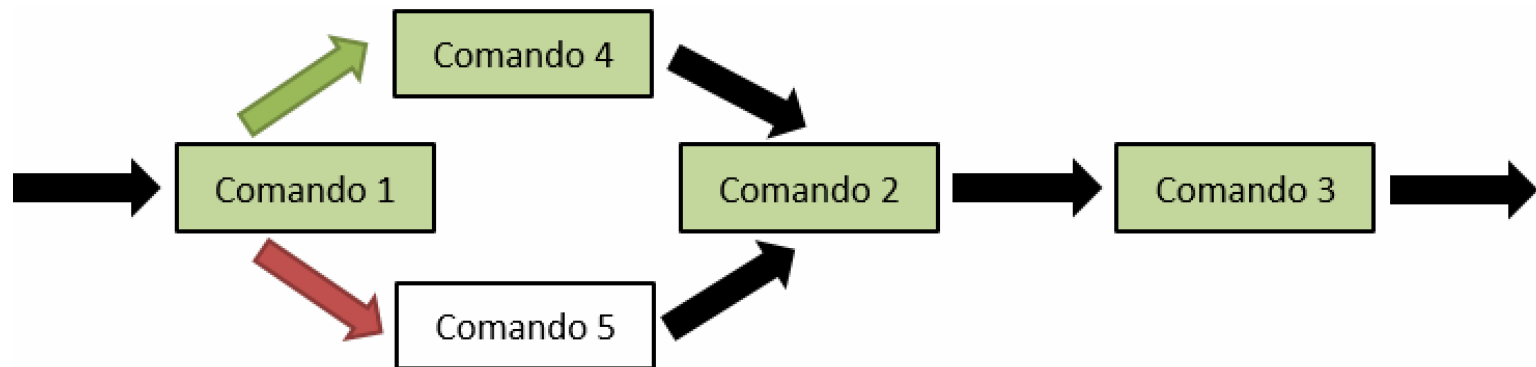
Parâmetros por valor

Parâmetros por referência

Estruturas de repetição

- **Introdução**

- Com tudo o que vimos até agora, já conseguimos implementar algoritmos que envolvem **diferentes caminhos** de execução.



- Porém, todos os blocos de comandos, até agora, são executados apenas uma **única vez**.

Estruturas de repetição

- **Introdução**

- Voltando novamente ao algoritmo que vimos nas primeiras aulas: trocar uma lâmpada.

```
1 - Acionar o interruptor
2 - Se a lâmpada acender, encerre.
3 - Pegar uma escada
4 - Posicionar a escada embaixo da lâmpada
5 - Buscar uma lâmpada nova
6 - Subir na escada
7 - Retirar a lâmpada velha
8 - Colocar a lâmpada nova
9 - Descer da escada
10 - Se a lâmpada nova acender, encerre.
11 - Repita os passos 5 a 10.
```



Estruturas de repetição

- **Introdução**

- Voltando novamente ao algoritmo que troca uma lâmpada.

```
1 - Acionar o interruptor
2 - Se a lâmpada acender, encerre.
3 - Pegar uma escada
4 - Posicionar a escada embaixo da lâmpada
5 - Buscar uma lâmpada nova
6 - Subir na escada
7 - Retirar a lâmpada velha
8 - Colocar a lâmpada nova
9 - Descer da escada
10 - Se a lâmpada nova acender, encerre.
11 - Repita os passos 5 a 10.
```



Repare que as linhas de comando de 5 a 11 serão executadas **repetidamente** até que a nova lâmpada acenda.



Estruturas de repetição

- **Introdução**

- A mesma **lógica** poderá ser aplicada em nossos programas, daqui em diante.
- Isso significa que alguns comandos poderão ser executados **mais de uma vez**, dependendo de uma condição.



Estruturas de repetição

- **Introdução**

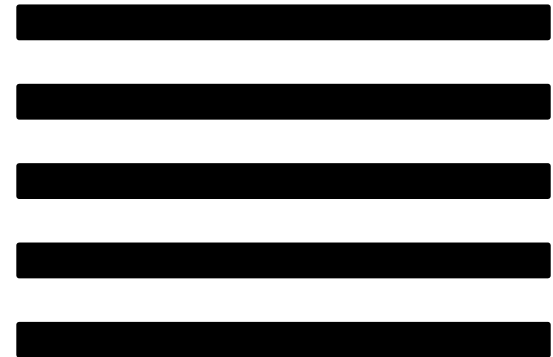
- Existem **3 maneiras** de implementar este **loop**:

- Comando “enquanto”

- Comando “para”



- Comando “faca”



Estruturas de repetição

- **Introdução**

- As três maneiras possibilitam a execução de um **bloco de instruções** sucessivas vezes.
- Todo **laço de repetição** possui três elementos essenciais para o seu funcionamento:

❑ **Contador:** também chamado de variável de controle. Ex: inteiro $i = 0$

❑ **Condição:** Condição para que se continue ou não a executar o trecho do código. Ex: $i < 10$

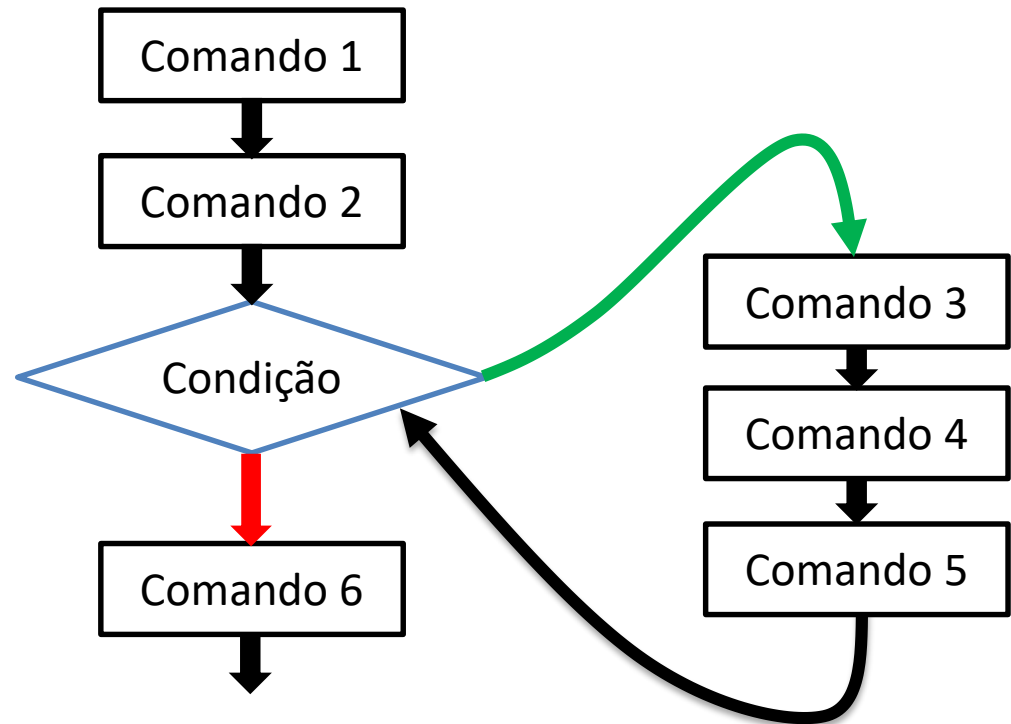
❑ **Incremento/Decremento/Alteração** da variável de controle. Ex: $i = i + 1$

Estruturas de repetição

- **Comando “enquanto”**

- O teste da condição é realizado no início, antes do bloco ser executado pela primeira vez.

- O programa segue sua execução linha por linha, até chegar no bloco “enquanto”.

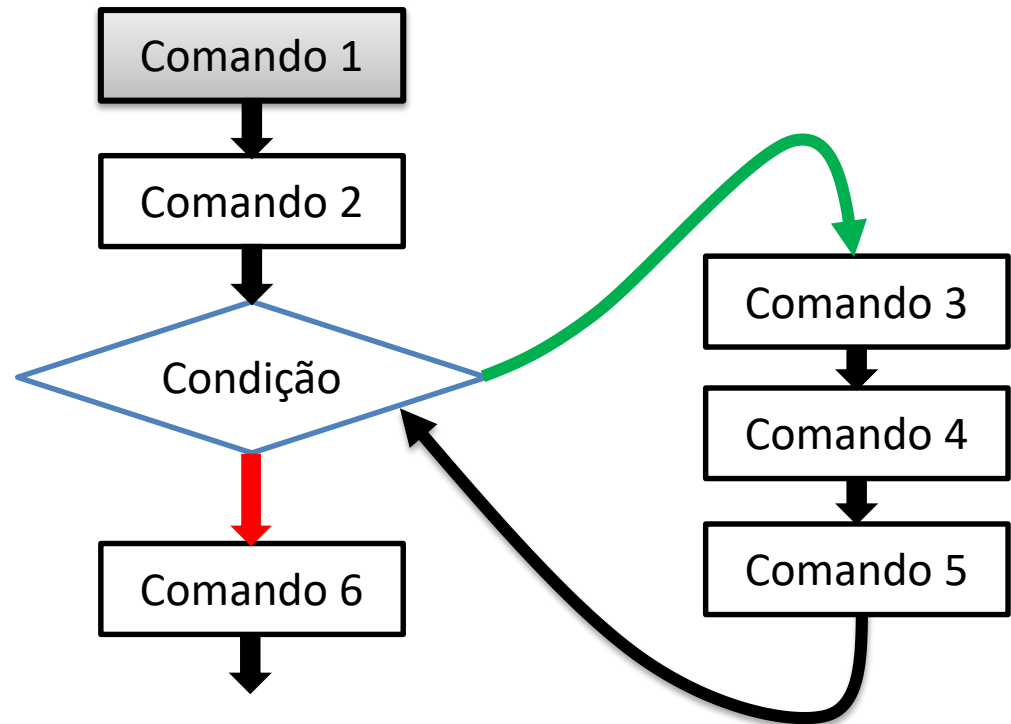


Estruturas de repetição

- **Comando “enquanto”**

- O teste da condição é realizado no início, antes do bloco ser executado pela primeira vez.

- O programa segue sua execução linha por linha, até chegar no bloco “enquanto”.

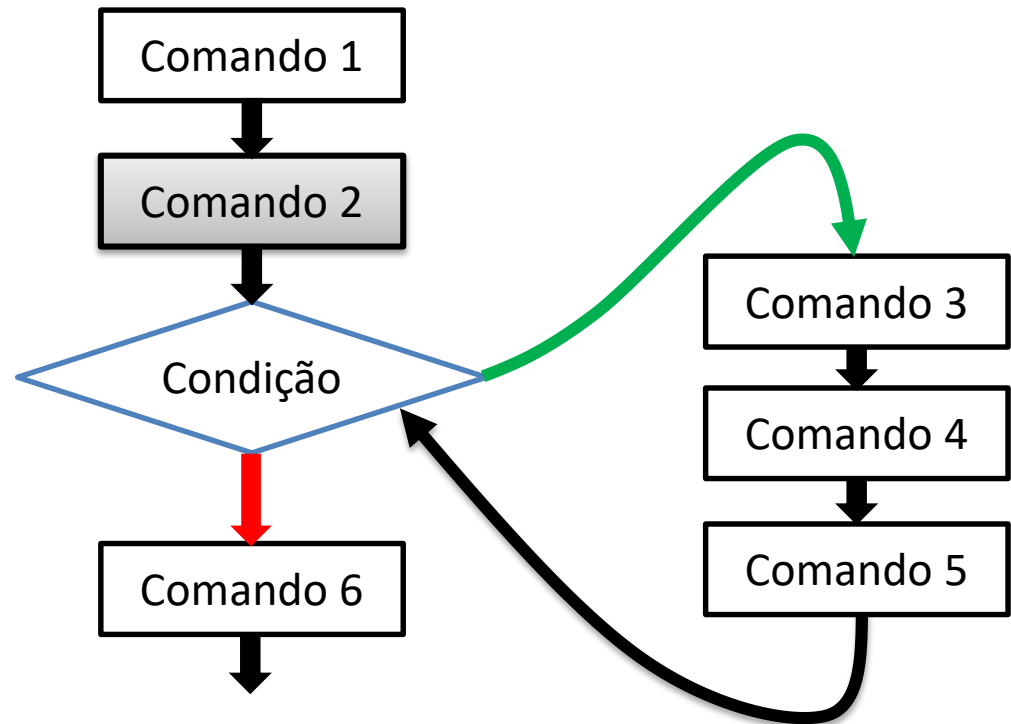


Estruturas de repetição

- **Comando “enquanto”**

- O teste da condição é realizado no início, antes do bloco ser executado pela primeira vez.

- O programa segue sua execução linha por linha, até chegar no bloco “enquanto”.

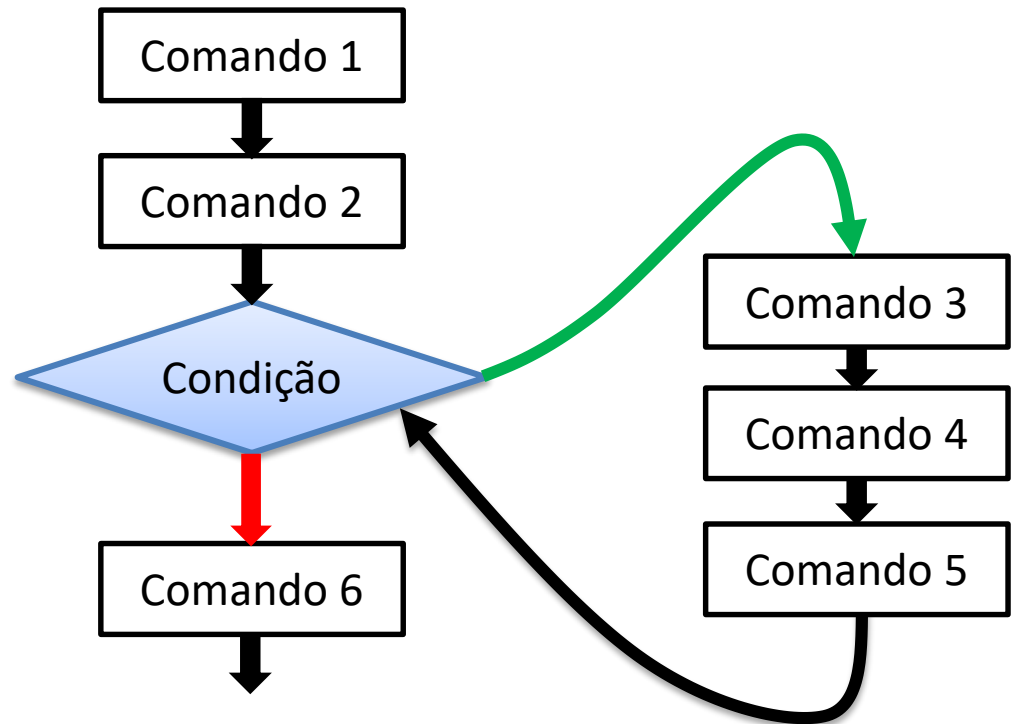


Estruturas de repetição

- **Comando “enquanto”**

- O teste da condição é realizado no início, antes do bloco ser executado pela primeira vez.

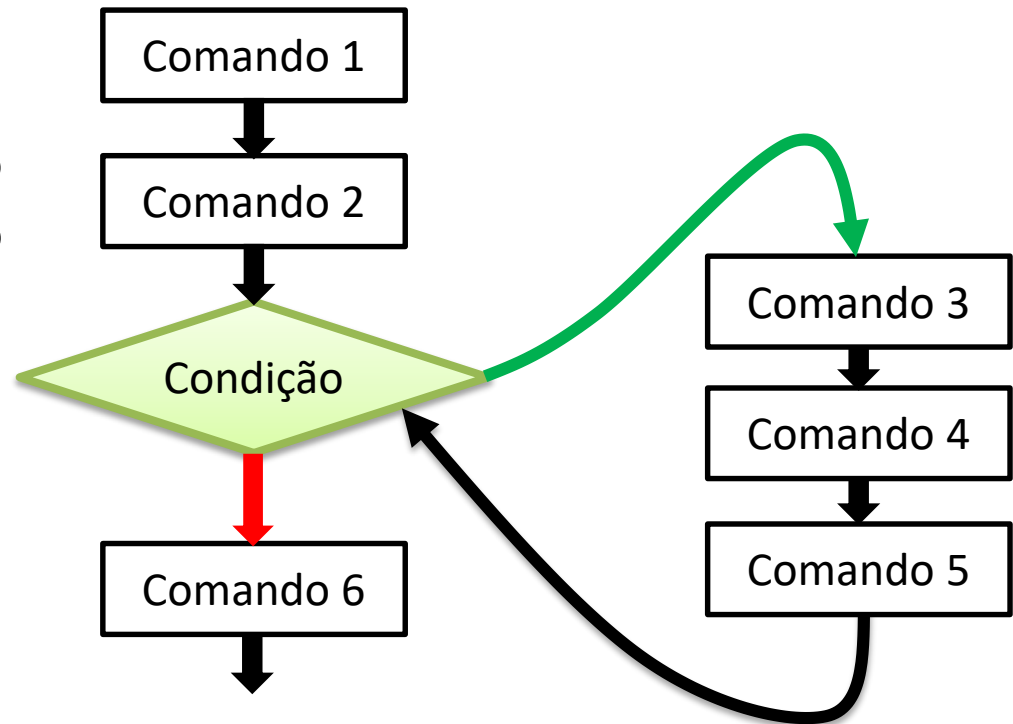
- O programa segue sua execução linha por linha, até chegar no bloco “enquanto”.



Estruturas de repetição

- **Comando “enquanto”**

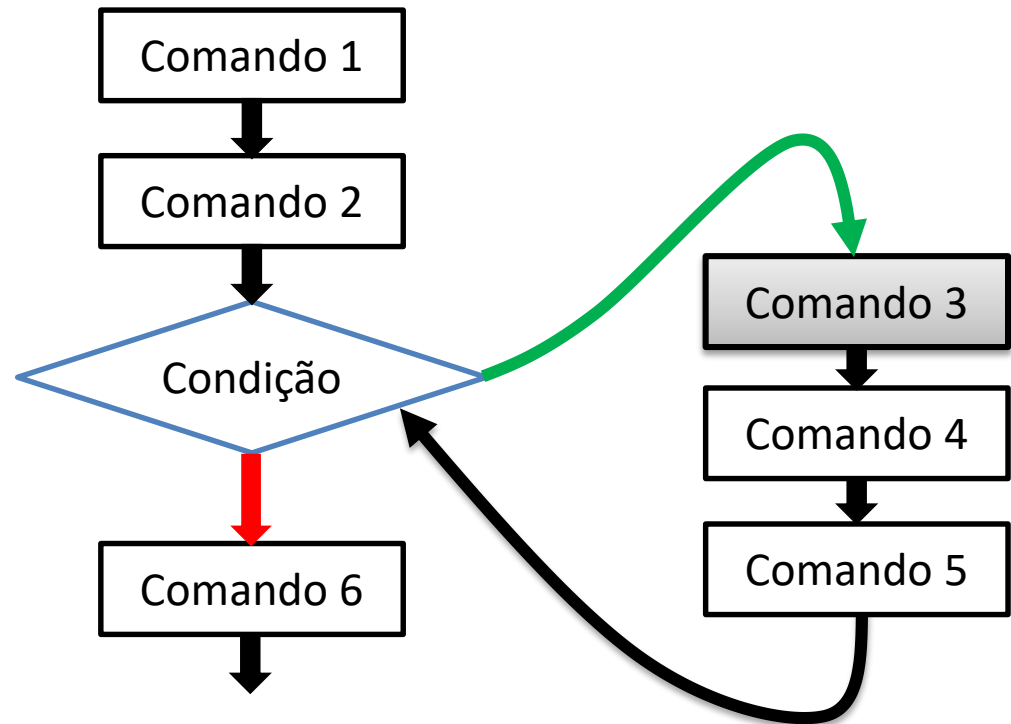
- O teste da condição é realizado no início, antes do bloco ser executado pela primeira vez.
- O programa segue sua execução linha por linha, até chegar no bloco “enquanto”.
- Se a condição resultar em verdadeiro, a execução entrará no bloco de comandos.



Estruturas de repetição

- **Comando “enquanto”**

- O teste da condição é realizado no início, antes do bloco ser executado pela primeira vez.
- O programa segue sua execução linha por linha, até chegar no bloco “enquanto”.
- Se a condição resultar em verdadeiro, a execução entrará no bloco de comandos.



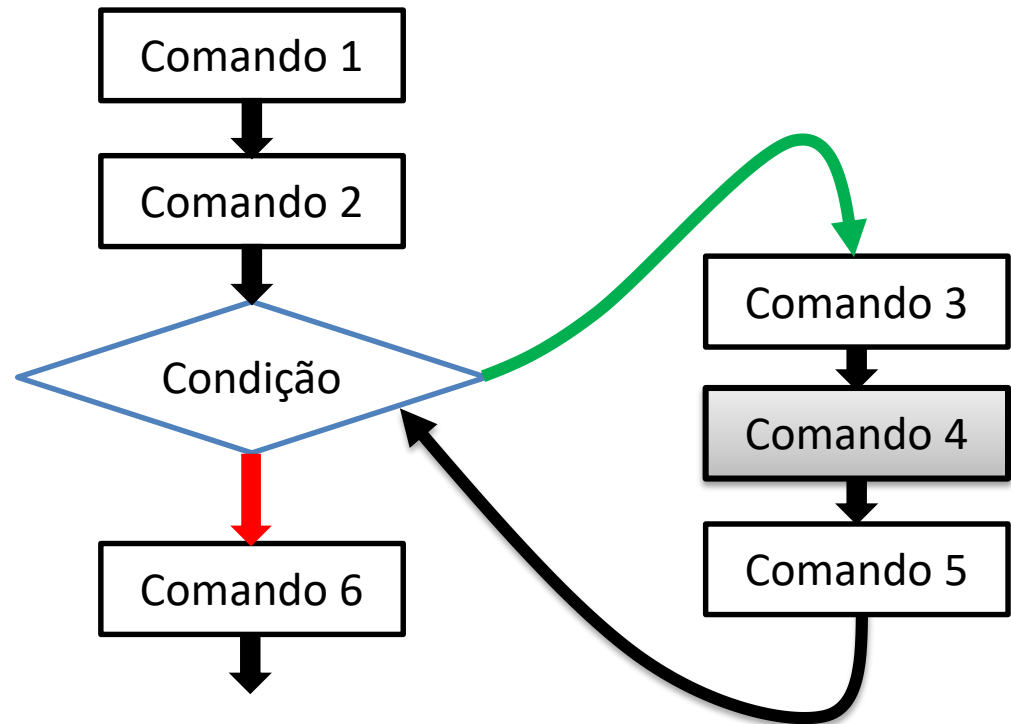
Estruturas de repetição

- **Comando “enquanto”**

- O teste da condição é realizado no início, antes do bloco ser executado pela primeira vez.

- O programa segue sua execução linha por linha, até chegar no bloco “enquanto”.

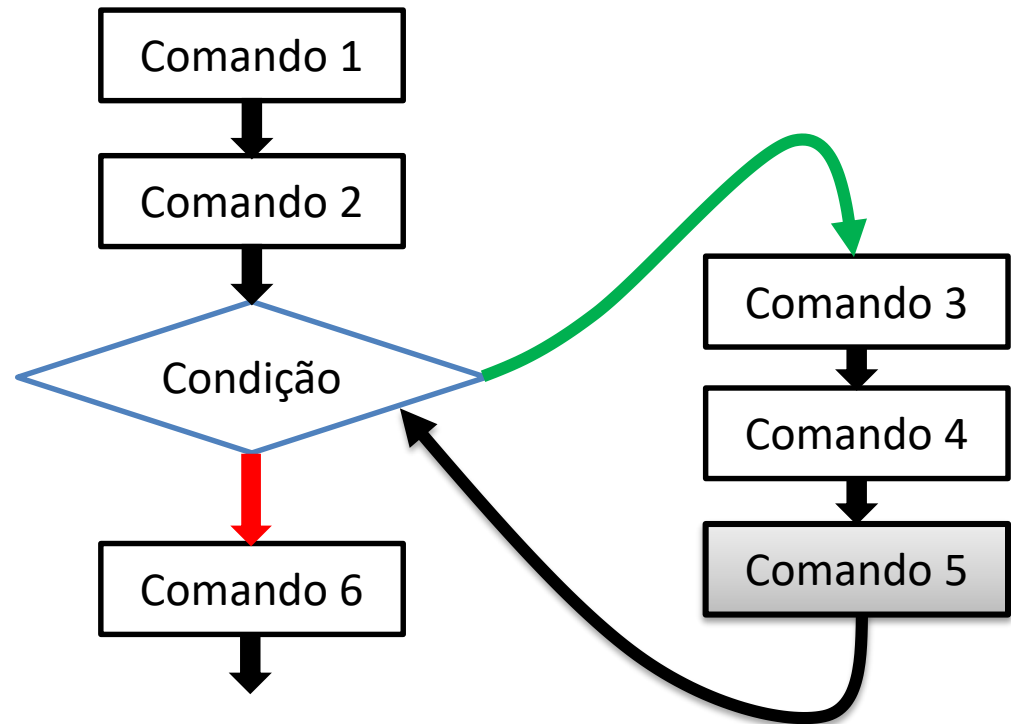
- Se a condição resultar em verdadeiro, a execução entrará no bloco de comandos.



Estruturas de repetição

- **Comando “enquanto”**

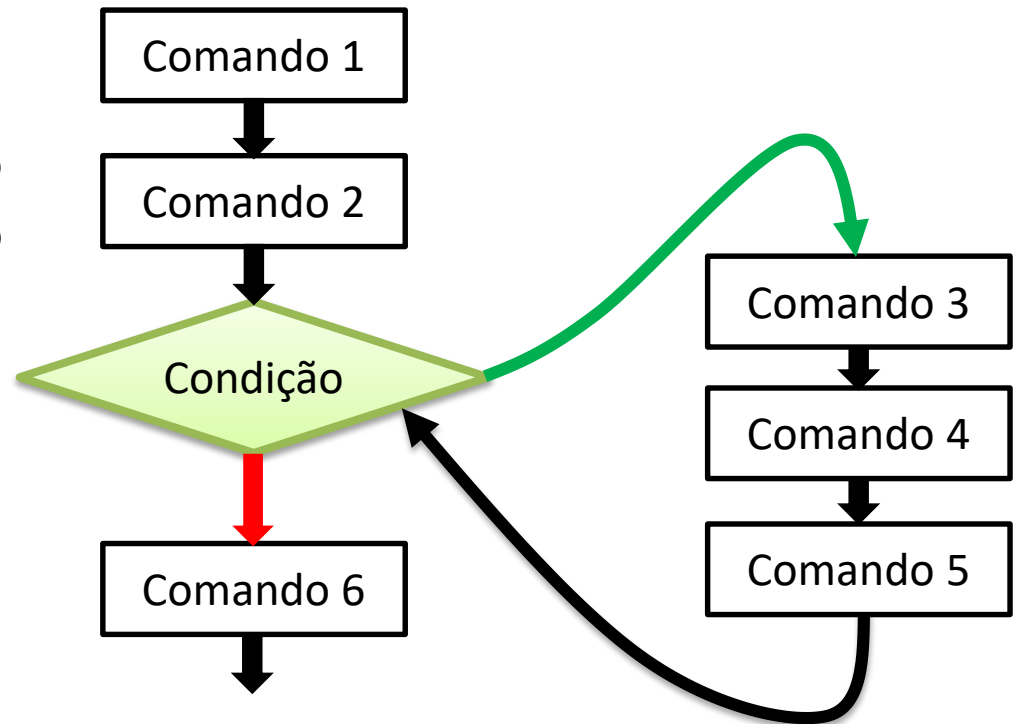
- O teste da condição é realizado no início, antes do bloco ser executado pela primeira vez.
- O programa segue sua execução linha por linha, até chegar no bloco “enquanto”.
- Se a condição resultar em verdadeiro, a execução entrará no bloco de comandos.



Estruturas de repetição

- **Comando “enquanto”**

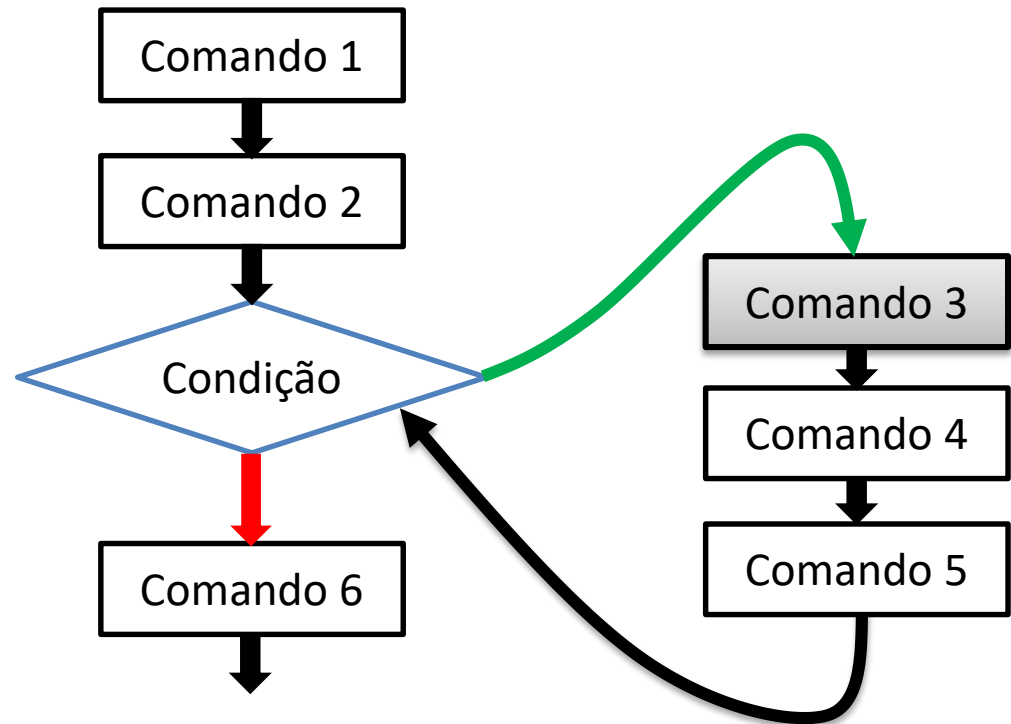
- O teste da condição é realizado no início, antes do bloco ser executado pela primeira vez.
- O programa segue sua execução linha por linha, até chegar no bloco “enquanto”.
- Se a condição resultar em verdadeiro, a execução entrará no bloco de comandos.



Estruturas de repetição

- **Comando “enquanto”**

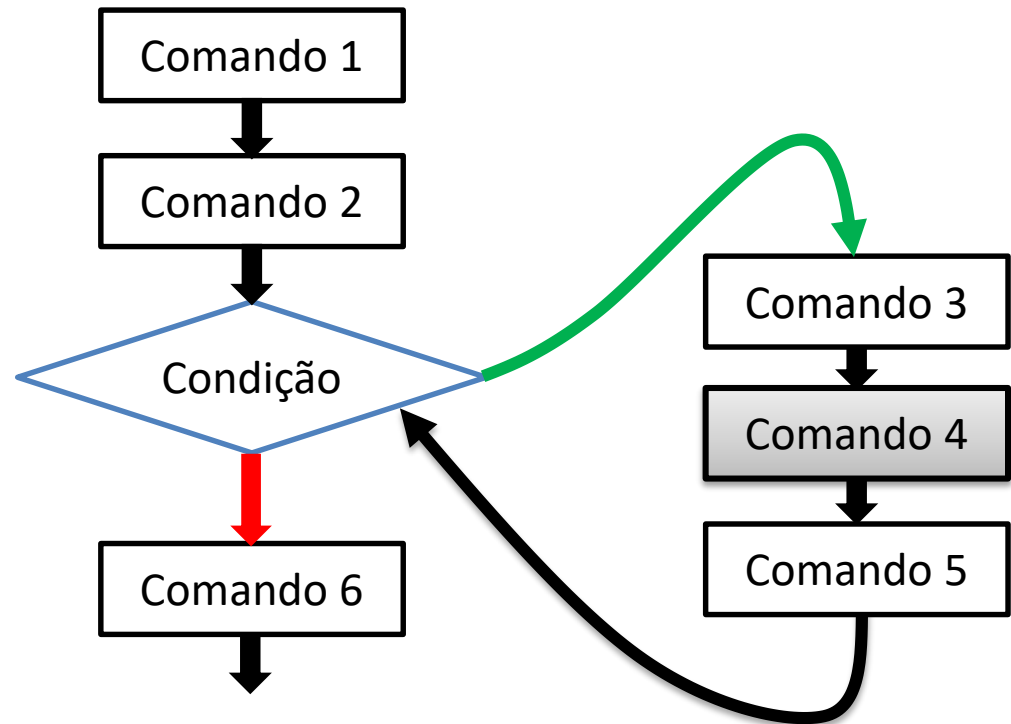
- O teste da condição é realizado no início, antes do bloco ser executado pela primeira vez.
- O programa segue sua execução linha por linha, até chegar no bloco “enquanto”.
- Se a condição resultar em verdadeiro, a execução entrará no bloco de comandos.



Estruturas de repetição

- **Comando “enquanto”**

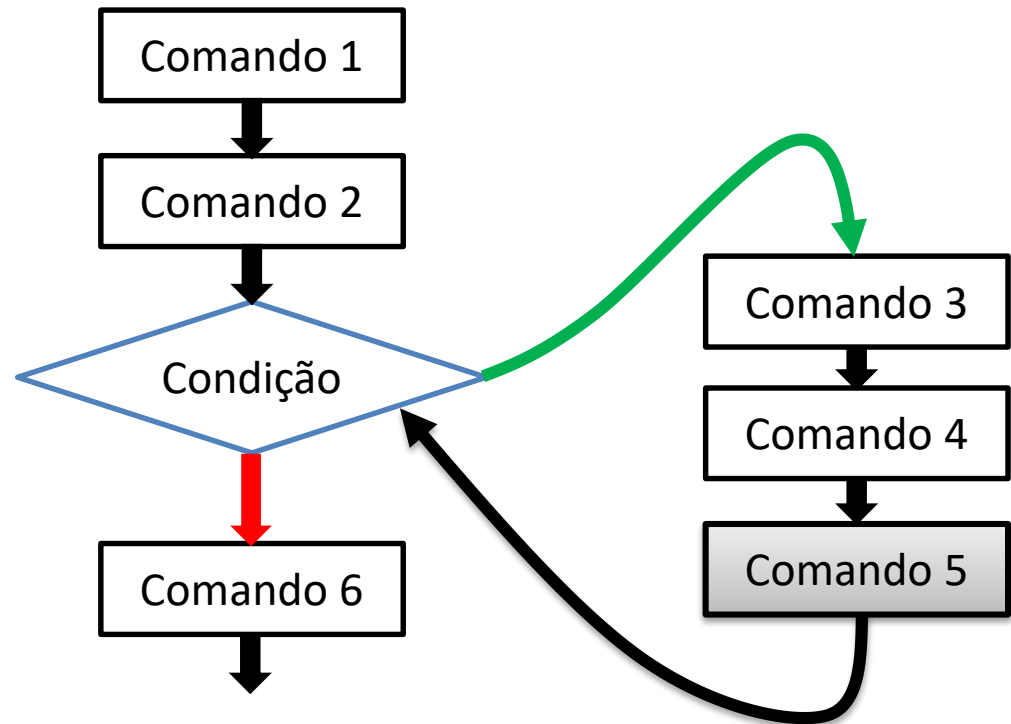
- O teste da condição é realizado no início, antes do bloco ser executado pela primeira vez.
- O programa segue sua execução linha por linha, até chegar no bloco “enquanto”.
- Se a condição resultar em verdadeiro, a execução entrará no bloco de comandos.



Estruturas de repetição

- **Comando “enquanto”**

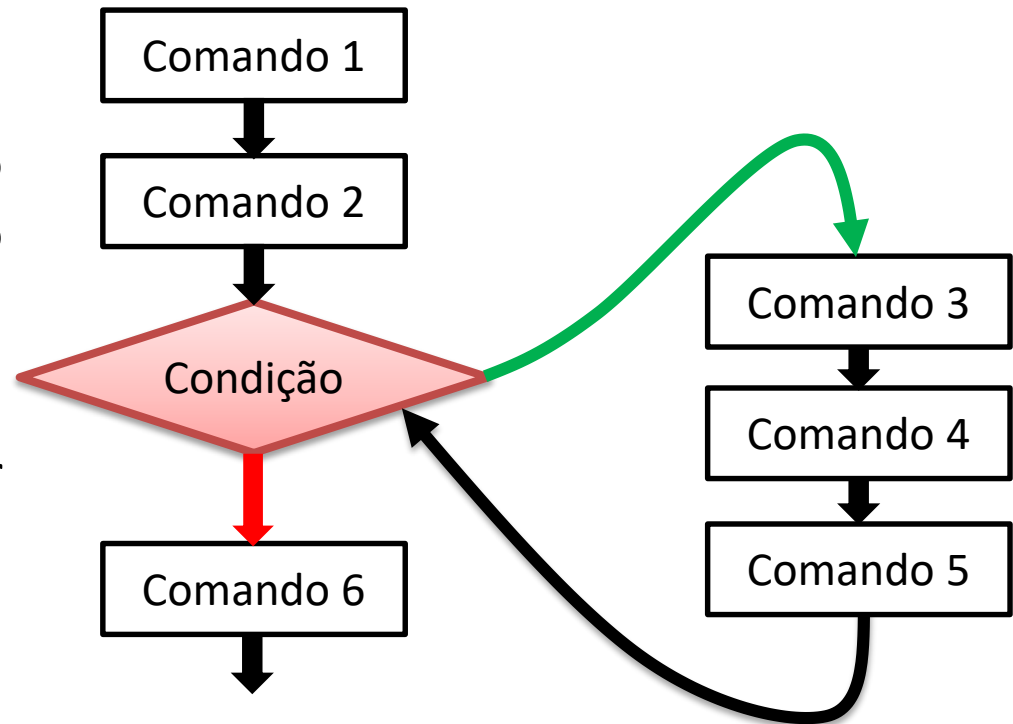
- O teste da condição é realizado no início, antes do bloco ser executado pela primeira vez.
- O programa segue sua execução linha por linha, até chegar no bloco “enquanto”.
- Se a condição resultar em verdadeiro, a execução entrará no bloco de comandos.



Estruturas de repetição

- **Comando “enquanto”**

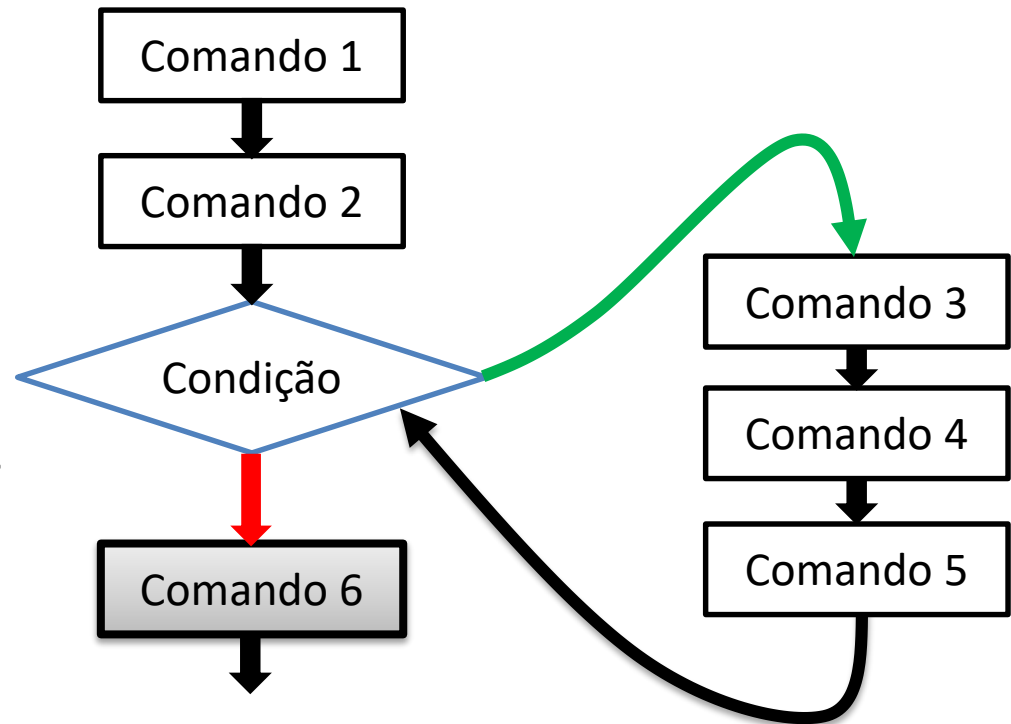
- O teste da condição é realizado no início, antes do bloco ser executado pela primeira vez.
- O programa segue sua execução linha por linha, até chegar no bloco “enquanto”.
- Quando a condição se tornar falsa, o algoritmo continuará com os comandos seguintes.



Estruturas de repetição

- **Comando “enquanto”**

- O teste da condição é realizado no início, antes do bloco ser executado pela primeira vez.
- O programa segue sua execução linha por linha, até chegar no bloco “enquanto”.
- Quando a condição se tornar falsa, o algoritmo continuará com os comandos seguintes.



Estruturas de repetição

- Comando “enquanto”

- Em português, a sintaxe para o comando “enquanto” é:


```
enquanto(condição){  
    comando1  
    comando2  
    comando3  
    ...  
}
```

Estruturas de repetição

- Comando “enquanto”

- Em português, a sintaxe para o comando “enquanto” é:

Palavra reservada que indica um laço de repetição “enquanto”.



```
enquanto(condição){  
    comando1  
    comando2  
    comando3  
    ...  
}
```


Estruturas de repetição

- Comando “enquanto”

- Em português, a sintaxe para o comando “enquanto” é:

Palavra reservada que indica um laço de repetição “enquanto”.

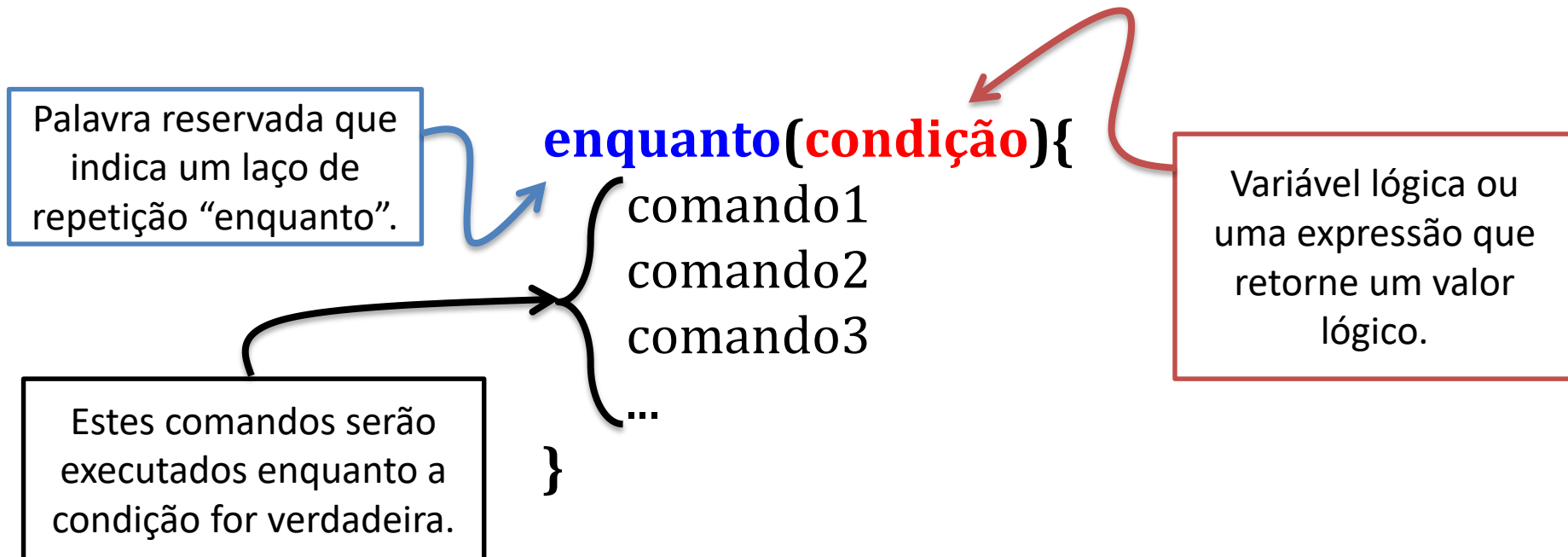
```
enquanto(condição){  
    comando1  
    comando2  
    comando3  
    ...  
}
```

Variável lógica ou uma expressão que retorne um valor lógico.

Estruturas de repetição

- Comando “enquanto”

- Em português, a sintaxe para o comando “enquanto” é:



Estrutu



Importante!

Toda estrutura de repetição precisa ter os três elementos:

- **Comando**
 - Em por
- ☐ Variável de controle
 - ☐ Condição
 - ☐ Alteração na variável de controle
- nto” é:

```
enquanto(condição){  
    comando1  
    comando2  
    comando3  
    ...  
}
```

Estrutu



Importante!

Toda estrutura de repetição precisa ter os três elementos:

- **Comando**
 - Em por
- ☐ **Variável de controle**
- ☐ **Condição**
- ☐ **Alteração na variável de controle**

ento” é:

Atribuição de um valor inicial para a variável de controle.

```
enquanto(condição){  
    comando1  
    comando2  
    comando3  
    ...  
}
```

Estrutu



Importante!

Toda estrutura de repetição precisa ter os três elementos:

- **Comando**
 - Em português, o “comando” é:

- ☐ Variável de controle

- ☐ Condição

- ☐ Alteração na variável de controle

Atribuição de um valor inicial para a variável de controle.

```
enquanto( Condição ){  
    comando1  
    comando2  
    comando3  
    ...  
}
```

Estrutu



Importante!

Toda estrutura de repetição precisa ter os três elementos:

- **Comando**
 - Em português, a estrutura de repetição “**enquanto**” é:

☐ **Variável de controle**

☐ **Condição**

☐ **Alteração na variável de controle**

Atribuição de um valor inicial para a variável de controle.

enquanto (**Condição**) {

comando1

comando2

comando3

Alteração na variável de controle.

}

Estruturas de repetição

- Comando “enquanto”

- Por exemplo, veja o seguinte trecho de código:

```
inteiro a = 1

enquanto (a < 10) {
    escreva(a, "\n")
    a = a + 1
}
```

- O que será impresso na tela se executarmos esse código?

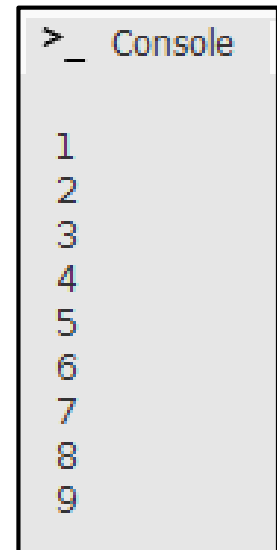
Estruturas de repetição

- Comando “enquanto”

- Por exemplo, veja o seguinte trecho de código:

```
inteiro a = 1

enquanto (a < 10) {
    escreva(a, "\n")
    a = a + 1
}
```



```
>_ Console

1
2
3
4
5
6
7
8
9
```

- O que será impresso na tela se executarmos esse código?

Estruturas de repetição

- Comando “enquanto”

- Por exemplo, veja o seguinte trecho de código:

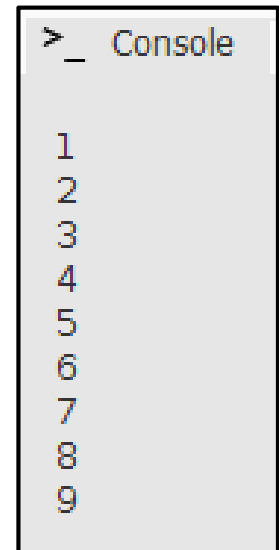
Variável de controle?

Condição?

Alteração da variável de controle?

```
inteiro a = 1

enquanto (a < 10) {
    escreva(a, "\n")
    a = a + 1
}
```



```
>_ Console

1
2
3
4
5
6
7
8
9
```

- O que será impresso na tela se executarmos esse código?

Estruturas de repetição

- Comando “enquanto”

- Por exemplo, veja o seguinte trecho de código:

Variável de controle?

`inteiro a = 1`

Condição?

```
enquanto (a < 10) {  
    escreva(a, "\n")  
    a = a + 1  
}
```

Alteração da variável de controle?

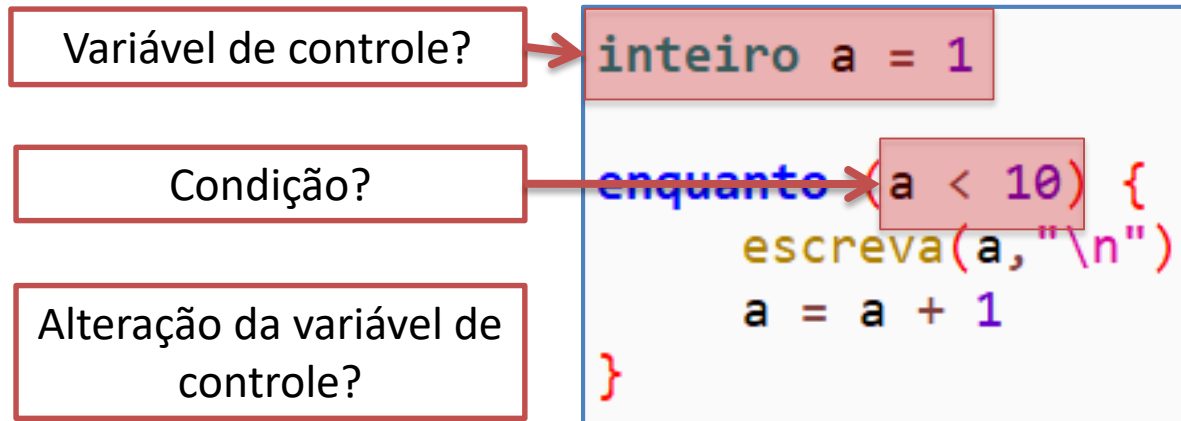
```
>_ Console  
  
1  
2  
3  
4  
5  
6  
7  
8  
9
```

- O que será impresso na tela se executarmos esse código?

Estruturas de repetição

- Comando “enquanto”

- Por exemplo, veja o seguinte trecho de código:



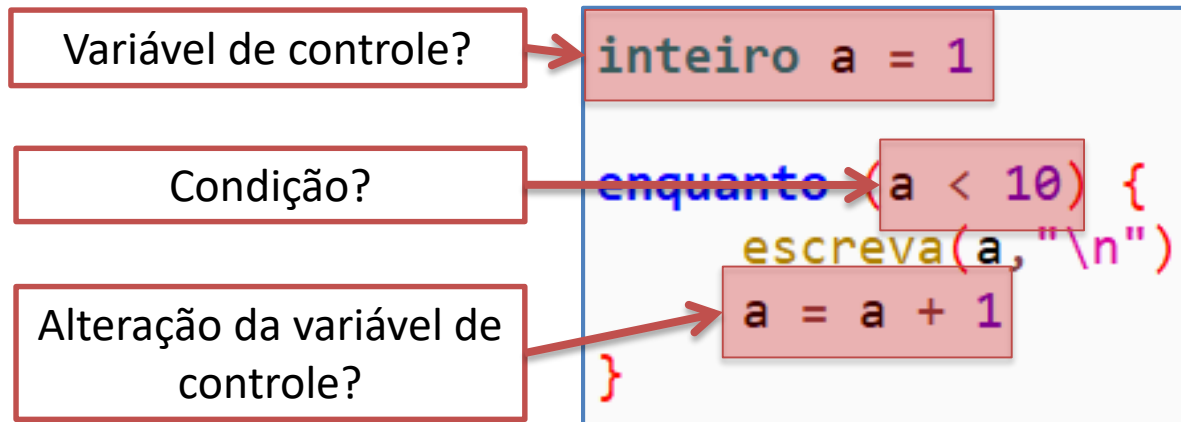
```
>_ Console
1
2
3
4
5
6
7
8
9
```

- O que será impresso na tela se executarmos esse código?

Estruturas de repetição

- Comando “enquanto”

- Por exemplo, veja o seguinte trecho de código:



```
>_ Console  
  
1  
2  
3  
4  
5  
6  
7  
8  
9
```

- O que será impresso na tela se executarmos esse código?



Estruturas de repetição

- **Exemplo**

1. Faça uma função “somaIntervalo” que recebe os limites de um intervalo de números inteiros e retorna a soma de todos eles (inclusive os limites).

Crie também uma função “inicio” que recebe dois números inteiros do usuário e escreve na tela o resultado da função “somaIntervalo”.

Exemplo:

`somaIntervalo(2, 5)` deve retornar: $2 + 3 + 4 + 5 = 14$

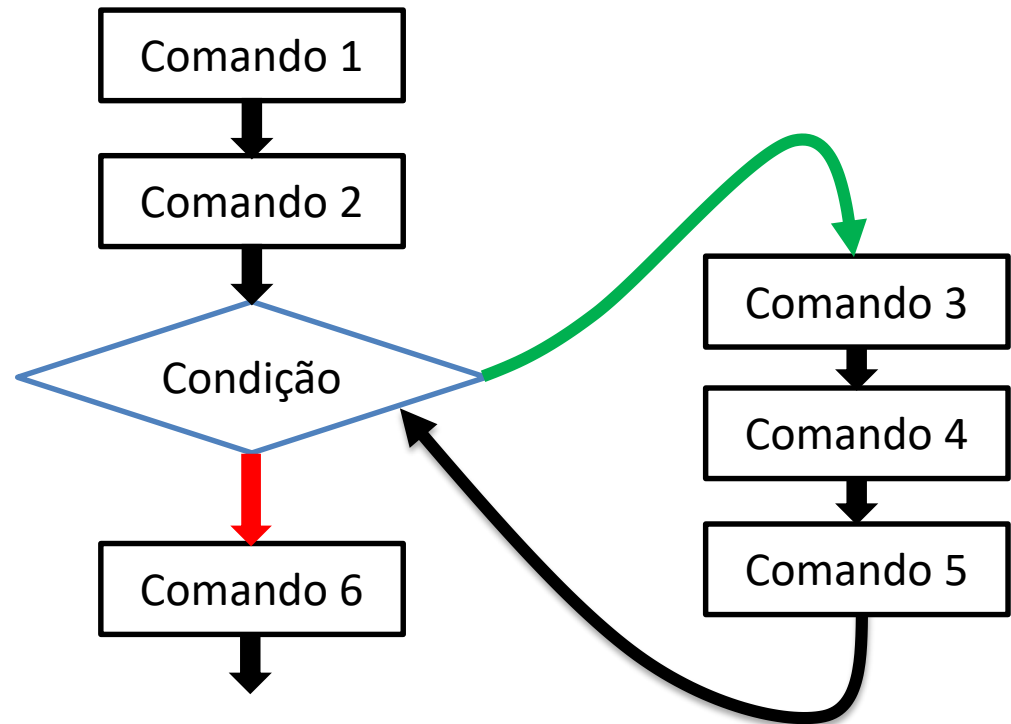
Obs: Utilize o comando “enquanto” para criar a função.

Estruturas de repetição

- **Comando “para”**

- É uma estrutura de repetição muito parecida com o enquanto.

- A lógica permanece a mesma: o teste de condição é feito antes de executar o bloco de comandos pela primeira vez.
 - A diferença é que a estrutura do “para” exige que se **escreva menos**.

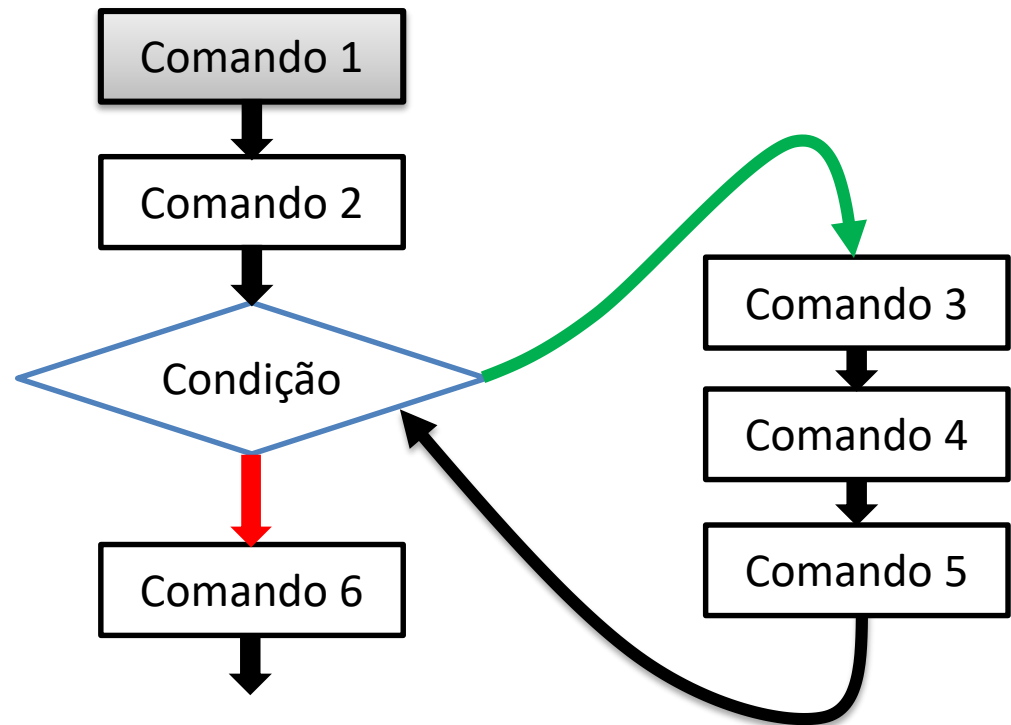


Estruturas de repetição

- Comando “para”

- É uma estrutura de repetição muito parecida com o enquanto.

- A lógica permanece a mesma: o teste de condição é feito antes de executar o bloco de comandos pela primeira vez.
 - A diferença é que a estrutura do “para” exige que se **escreva menos**.

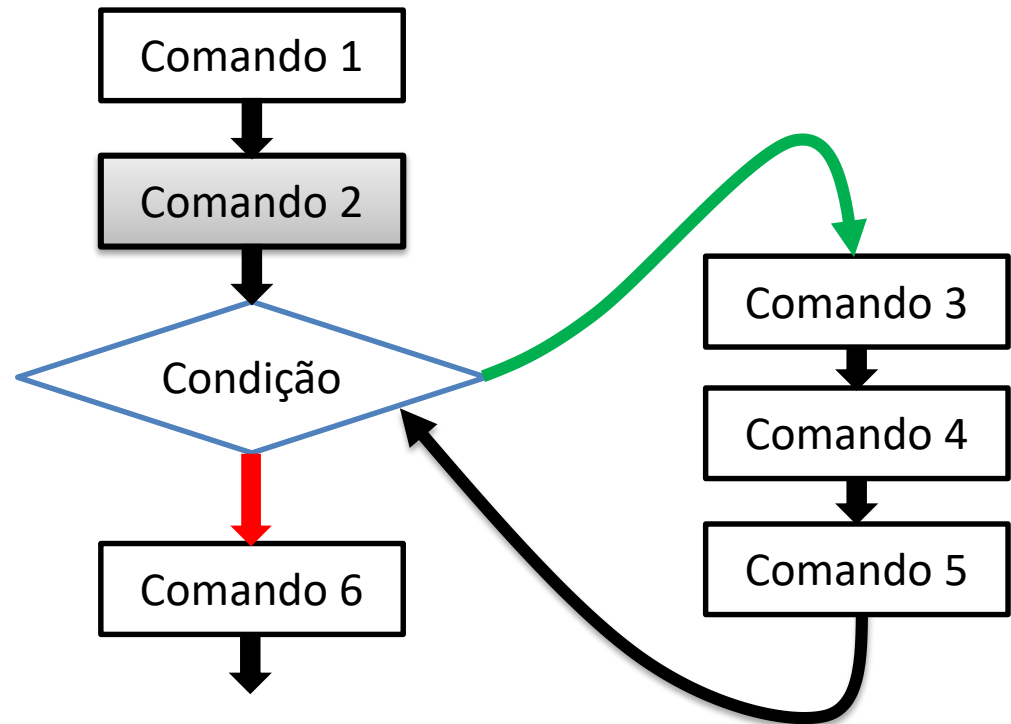


Estruturas de repetição

- **Comando “para”**

- É uma estrutura de repetição muito parecida com o enquanto.

- A lógica permanece a mesma: o teste de condição é feito antes de executar o bloco de comandos pela primeira vez.
 - A diferença é que a estrutura do “para” exige que se **escreva menos**.

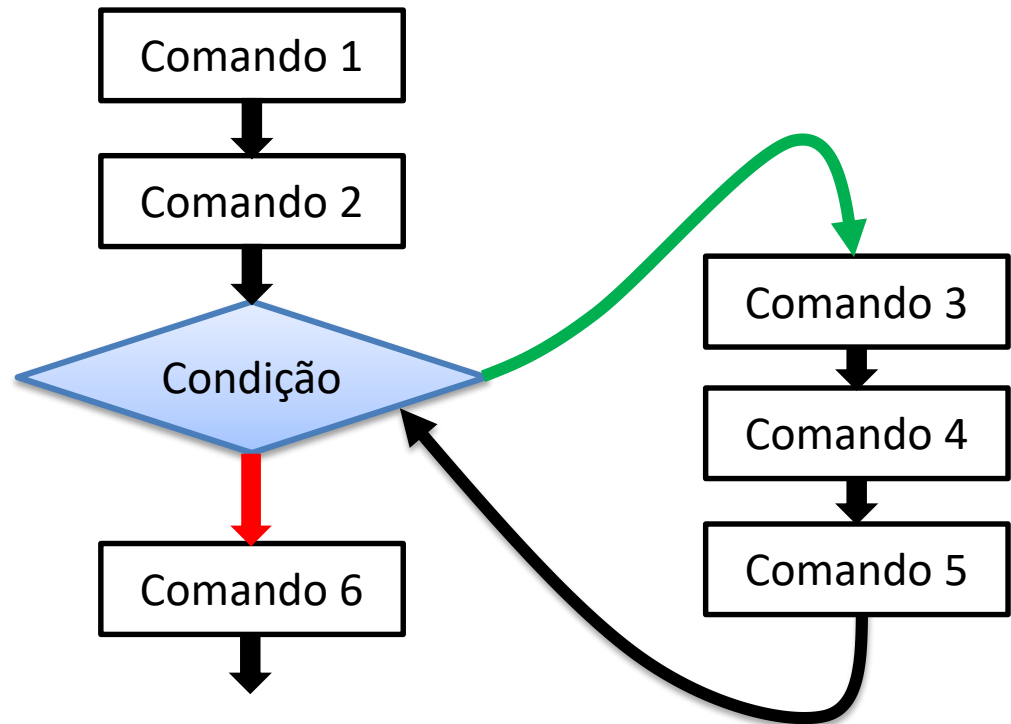


Estruturas de repetição

- Comando “para”

- É uma estrutura de repetição muito parecida com o enquanto.

- A lógica permanece a mesma: o teste de condição é feito antes de executar o bloco de comandos pela primeira vez.
 - A diferença é que a estrutura do “para” exige que se **escreva menos**.

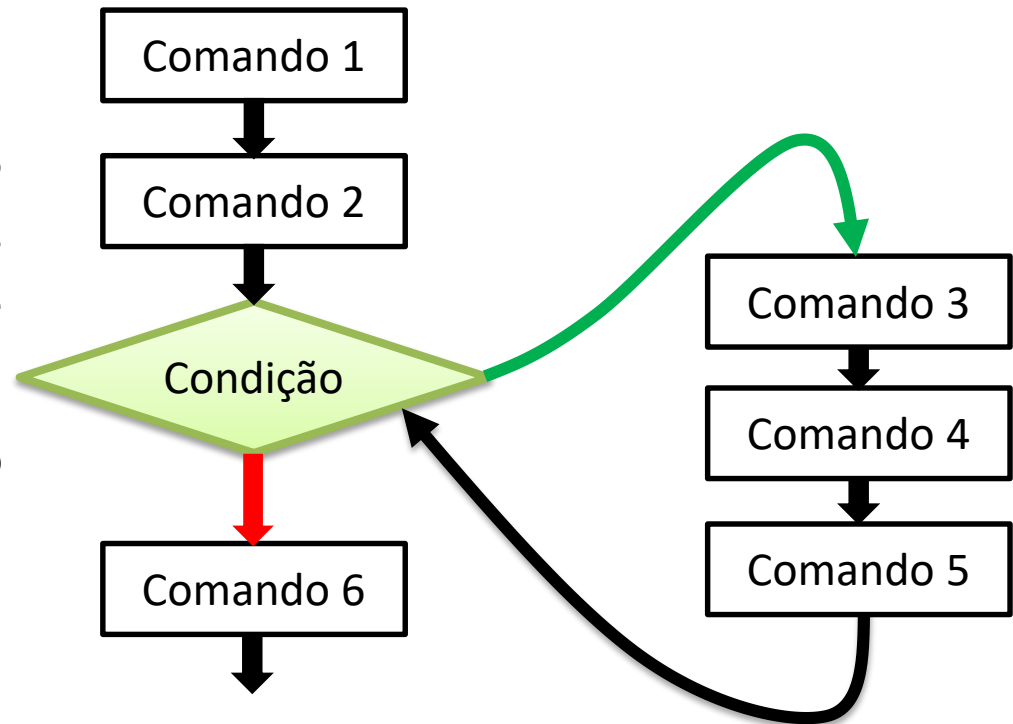


Estruturas de repetição

- **Comando “para”**

- É uma estrutura de repetição muito parecida com o enquanto.

- A lógica permanece a mesma: o teste de condição é feito antes de executar o bloco de comandos pela primeira vez.
 - A diferença é que a estrutura do “para” exige que se **escreva menos**.

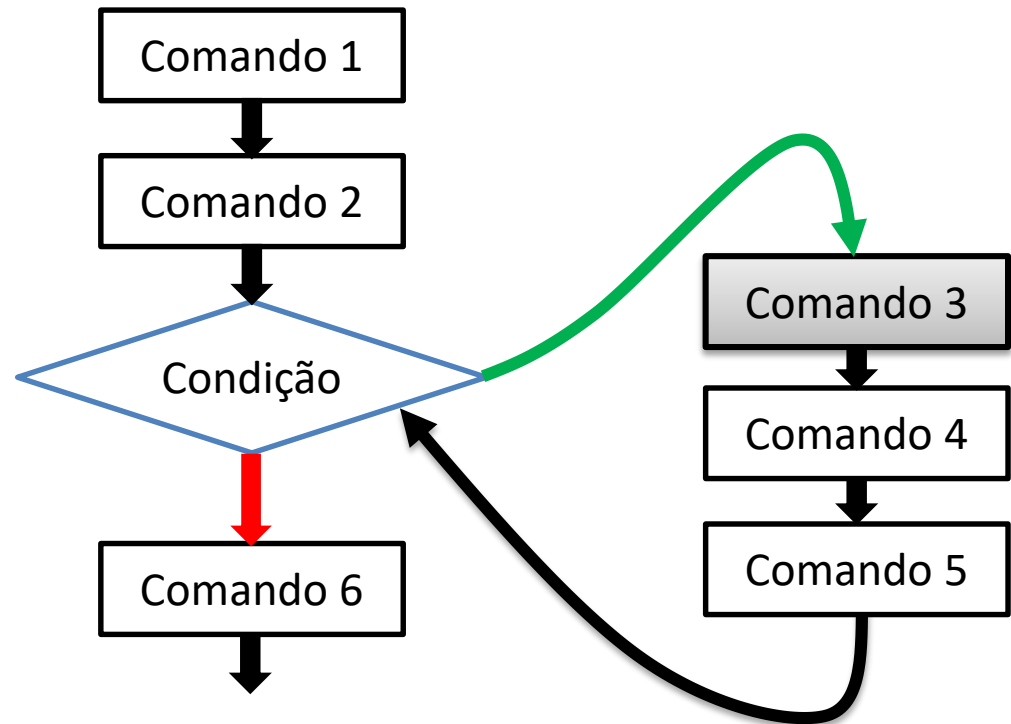


Estruturas de repetição

- **Comando “para”**

- É uma estrutura de repetição muito parecida com o enquanto.

- A lógica permanece a mesma: o teste de condição é feito antes de executar o bloco de comandos pela primeira vez.
 - A diferença é que a estrutura do “para” exige que se **escreva menos**.

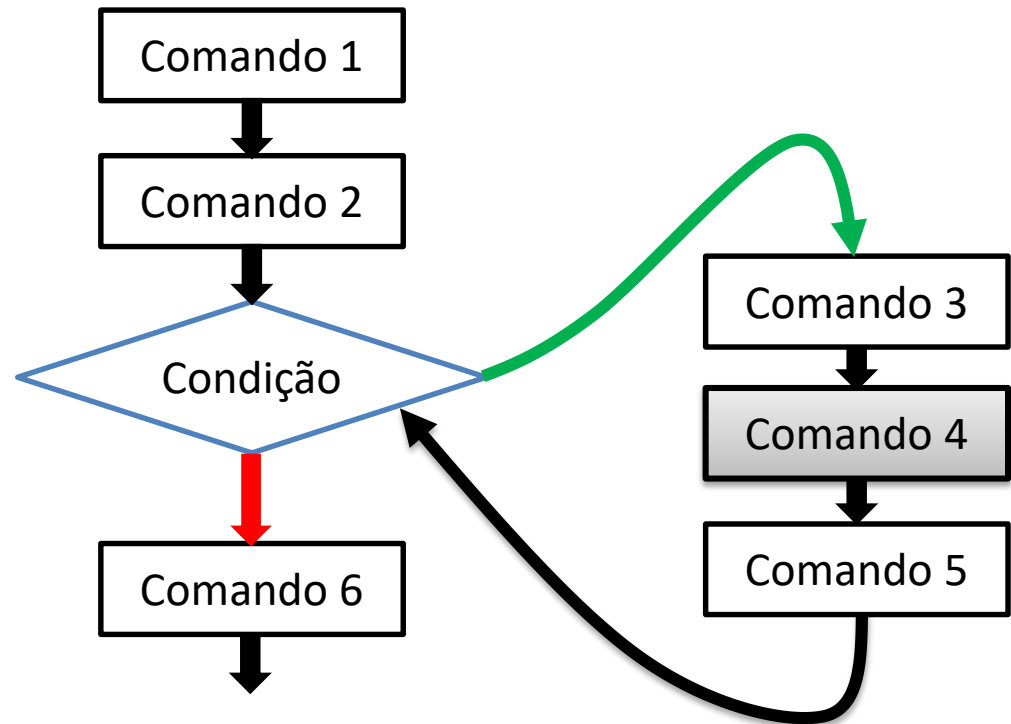


Estruturas de repetição

- **Comando “para”**

- É uma estrutura de repetição muito parecida com o enquanto.

- A lógica permanece a mesma: o teste de condição é feito antes de executar o bloco de comandos pela primeira vez.
 - A diferença é que a estrutura do “para” exige que se **escreva menos**.

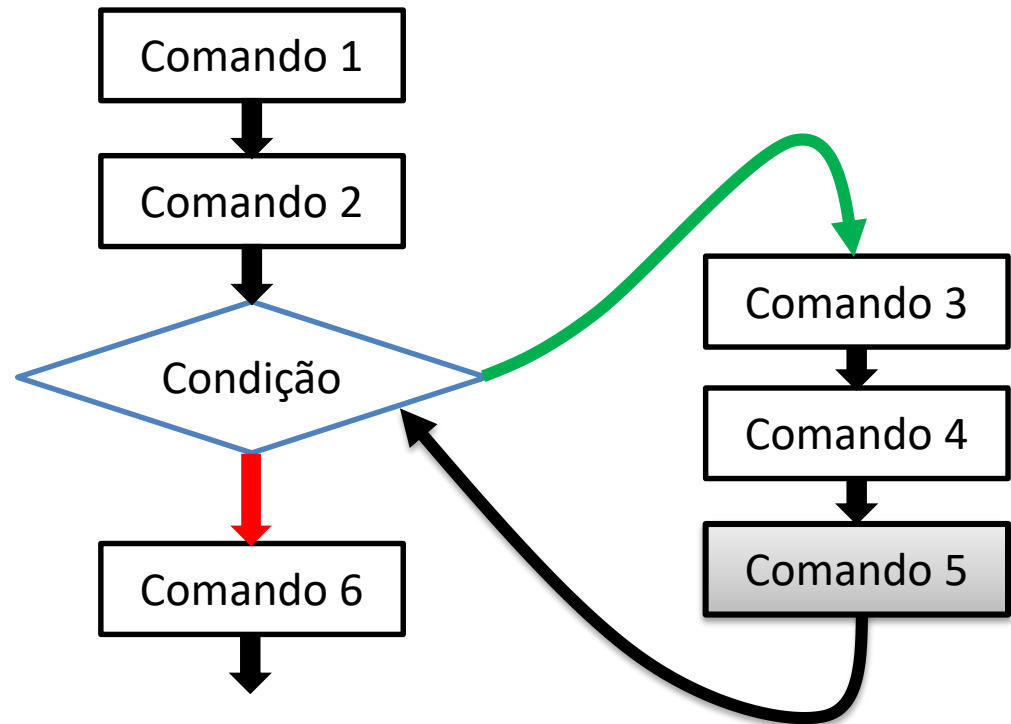


Estruturas de repetição

- **Comando “para”**

- É uma estrutura de repetição muito parecida com o enquanto.

- A lógica permanece a mesma: o teste de condição é feito antes de executar o bloco de comandos pela primeira vez.
 - A diferença é que a estrutura do “para” exige que se **escreva menos**.

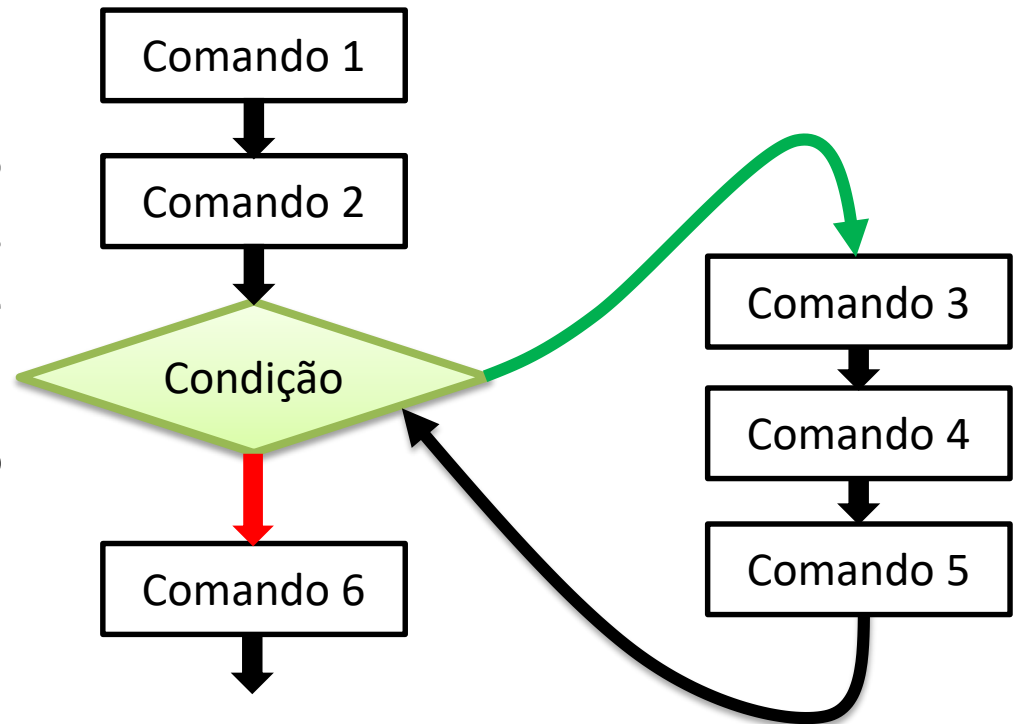


Estruturas de repetição

- **Comando “para”**

- É uma estrutura de repetição muito parecida com o enquanto.

- A lógica permanece a mesma: o teste de condição é feito antes de executar o bloco de comandos pela primeira vez.
 - A diferença é que a estrutura do “para” exige que se **escreva menos**.

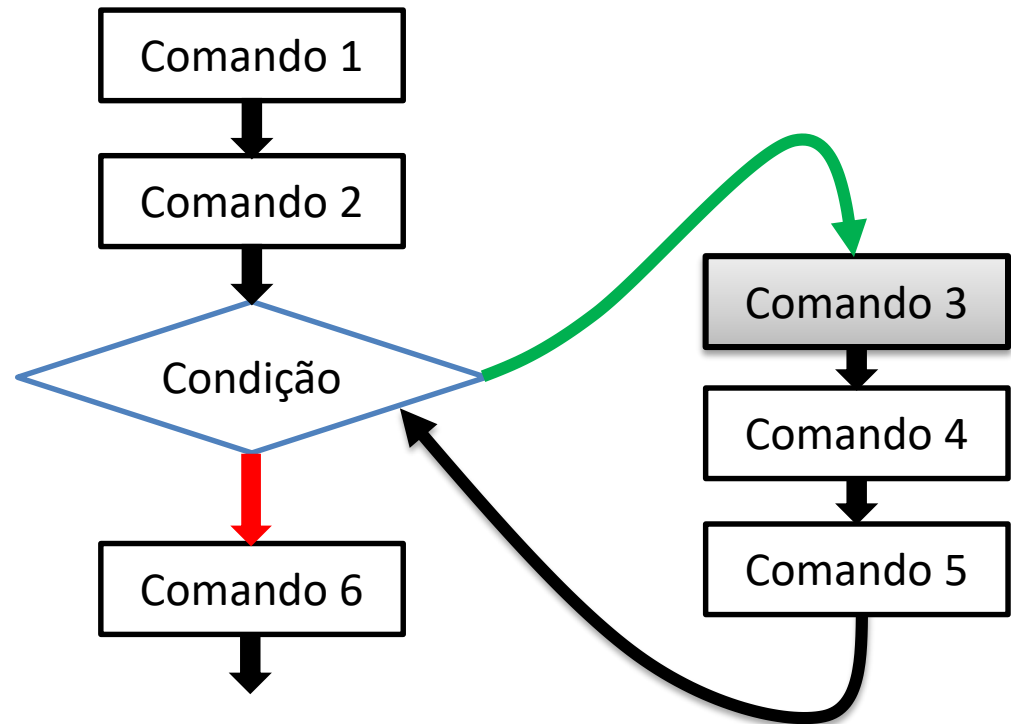


Estruturas de repetição

- **Comando “para”**

- É uma estrutura de repetição muito parecida com o enquanto.

- A lógica permanece a mesma: o teste de condição é feito antes de executar o bloco de comandos pela primeira vez.
 - A diferença é que a estrutura do “para” exige que se **escreva menos**.

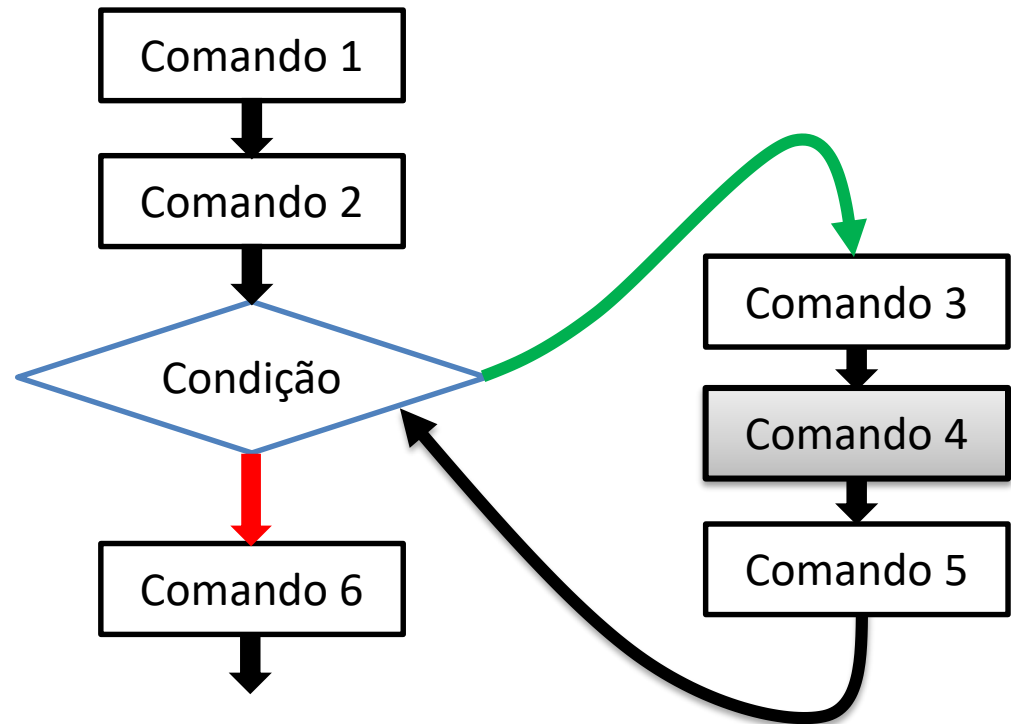


Estruturas de repetição

- **Comando “para”**

- É uma estrutura de repetição muito parecida com o enquanto.

- A lógica permanece a mesma: o teste de condição é feito antes de executar o bloco de comandos pela primeira vez.
 - A diferença é que a estrutura do “para” exige que se **escreva menos**.

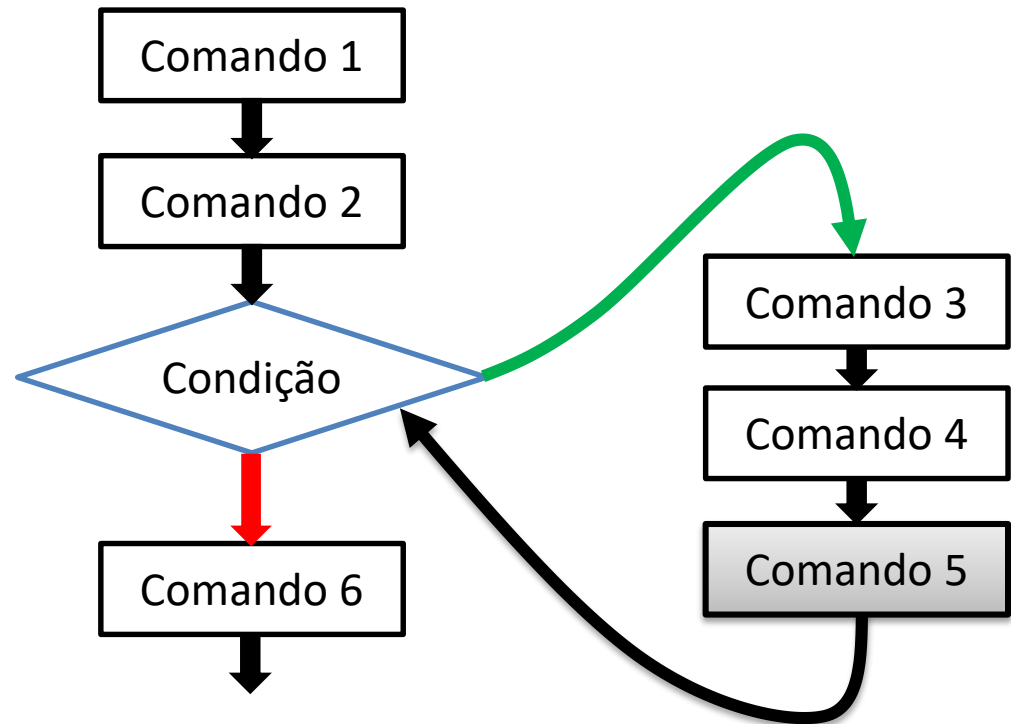


Estruturas de repetição

- **Comando “para”**

- É uma estrutura de repetição muito parecida com o enquanto.

- A lógica permanece a mesma: o teste de condição é feito antes de executar o bloco de comandos pela primeira vez.
 - A diferença é que a estrutura do “para” exige que se **escreva menos**.

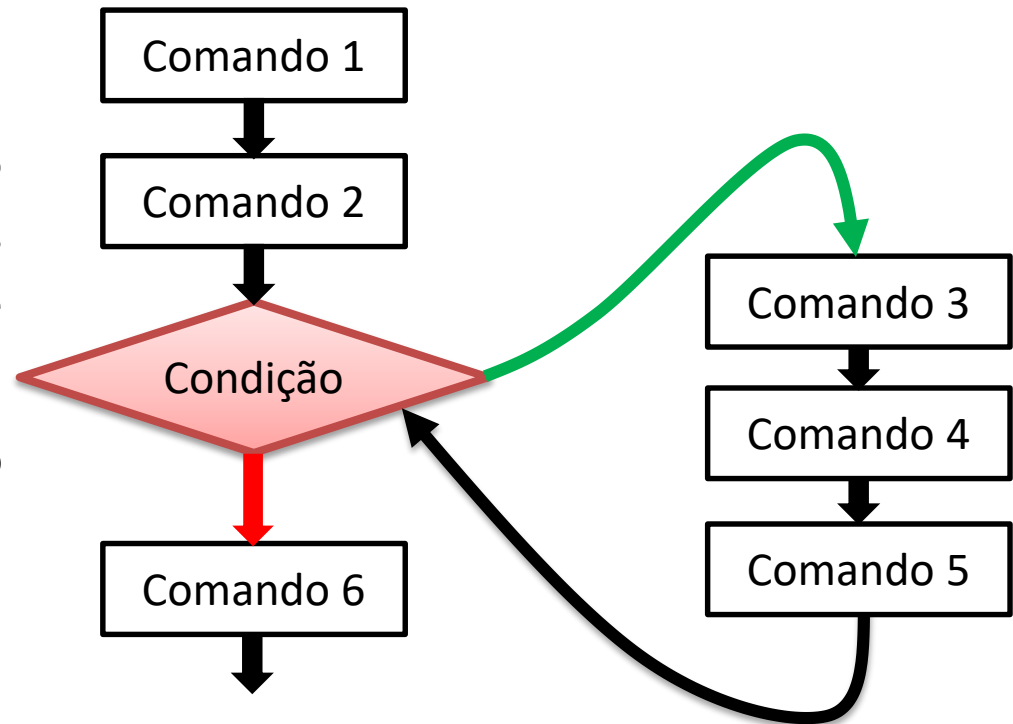


Estruturas de repetição

- **Comando “para”**

- É uma estrutura de repetição muito parecida com o enquanto.

- A lógica permanece a mesma: o teste de condição é feito antes de executar o bloco de comandos pela primeira vez.
 - A diferença é que a estrutura do “para” exige que se **escreva menos**.

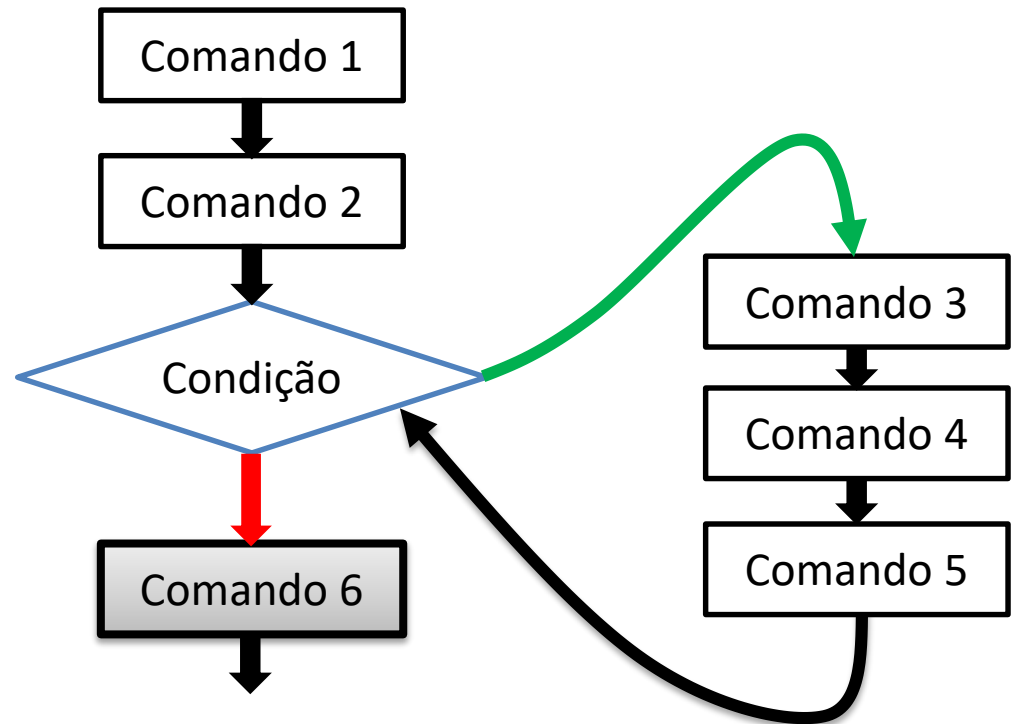


Estruturas de repetição

- **Comando “para”**

- É uma estrutura de repetição muito parecida com o enquanto.

- A lógica permanece a mesma: o teste de condição é feito antes de executar o bloco de comandos pela primeira vez.
 - A diferença é que a estrutura do “para” exige que se **escreva menos**.



Estruturas de repetição

- Comando “para”

- Em português, a sintaxe para o comando “para” é:


```
para(variávelDeControle = valorInicial; condição; variávelDeControle = novoValor){  
    comando1  
    comando2  
    comando3  
    ...  
}
```

Estruturas de repetição

- **Comando “para”**

Palavra reservada que indica um laço de repetição “enquanto”.

taxe para o comando “para” é:



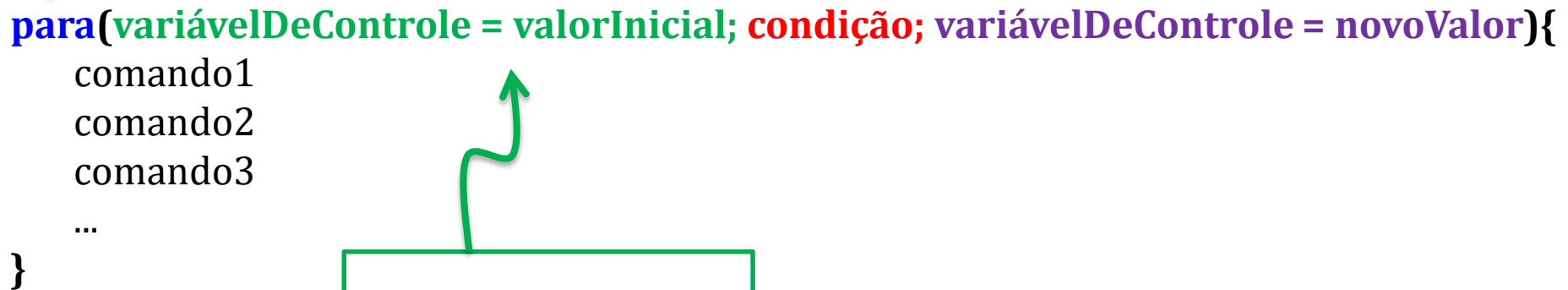
```
para(variávelDeControle = valorInicial; condição; variávelDeControle = novoValor){  
    comando1  
    comando2  
    comando3  
    ...  
}
```

Estruturas de repetição

- **Comando “para”**

Palavra reservada que indica um laço de repetição “enquanto”.

taxe para o comando “para” é:



```
para(variávelDeControle = valorInicial; condição; variávelDeControle = novoValor){  
    comando1  
    comando2  
    comando3  
    ...  
}
```

The diagram illustrates the syntax of a 'para' loop. A blue box highlights the word 'para' as a reserved word. A blue arrow points from this box to the 'para' keyword in the code snippet. A green box highlights the initialization part of the loop: '**variávelDeControle = valorInicial**'. A green arrow points from this box to the corresponding part of the code snippet. The code snippet itself shows the 'para' keyword followed by the initialization, condition, and update in parentheses, then a block of commands (comando1, comando2, comando3, ...) enclosed in curly braces.

Inicialização da
variável de controle.

Estruturas de repetição

- **Comando “para”**

Palavra reservada que indica um laço de repetição “enquanto”.

Sintaxe para o comando “para” é:

```
para(variávelDeControle = valorInicial; condição; variávelDeControle = novoValor){  
    comando1  
    comando2  
    comando3  
    ...  
}
```

Inicialização da variável de controle.

Variável lógica ou uma expressão que retorne um valor lógico.

Estruturas de repetição

- **Comando “para”**

Palavra reservada que indica um laço de repetição “enquanto”.

Sintaxe para o comando “para” é:

```
para(variávelDeControle = valorInicial; condição; variávelDeControle = novoValor){  
    comando1  
    comando2  
    comando3  
    ...  
}
```

Inicialização da variável de controle.

Variável lógica ou uma expressão que retorne um valor lógico.

Alteração da variável de controle.

Estruturas de repetição

- **Comando “para”**

Palavra reservada que indica um laço de repetição “enquanto”.

Sintaxe para o comando “para” é:

```
para(variávelDeControle = valorInicial; condição; variávelDeControle = novoValor){
```

```
    comando1  
    comando2  
    comando3  
    ...  
}
```

Estes comandos serão executados enquanto a condição for verdadeira.

Inicialização da variável de controle.

Variável lógica ou uma expressão que retorne um valor lógico.

Alteração da variável de controle.

Estruturas de repetição

- **Comando “para”**
 - Por exemplo, veja o seguinte trecho de código:

```
para (inteiro a=1; a < 10; a=a+1) {  
    escreva(a, "\n")  
}
```

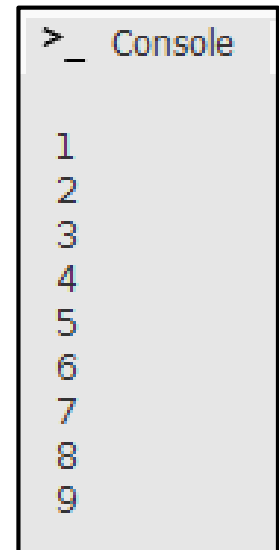
- O que será impresso na tela se executarmos esse código?

Estruturas de repetição

- Comando “para”

- Por exemplo, veja o seguinte trecho de código:

```
para (inteiro a=1; a < 10; a=a+1) {  
    escreva(a, "\n")  
}
```



```
>_ Console  
  
1  
2  
3  
4  
5  
6  
7  
8  
9
```

- O que será impresso na tela se executarmos esse código?

Estruturas de repetição

- **Comando “para”**
 - Por exemplo, veja o seguinte trecho de código:

Variável de controle?

Condição?

Alteração da variável de controle?

```
para (inteiro a=1; a < 10; a=a+1) {  
    escreva(a, "\n")  
}
```

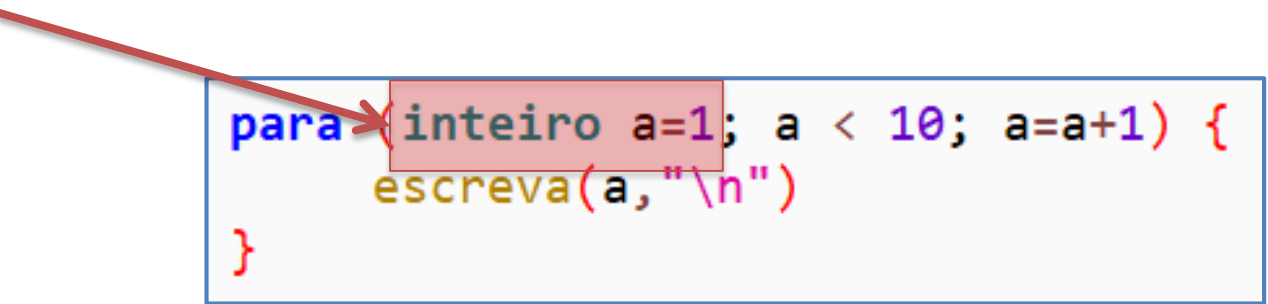
Estruturas de repetição

- Comando “para”
 - Por exemplo, veja o seguinte trecho de código:

Variável de controle?

Condição?

Alteração da variável de controle?



```
para (inteiro a=1; a < 10; a=a+1) {  
    escreva(a, "\n")  
}
```

Estruturas de repetição

- Comando “para”
 - Por exemplo, veja o seguinte trecho de código:

Variável de controle?

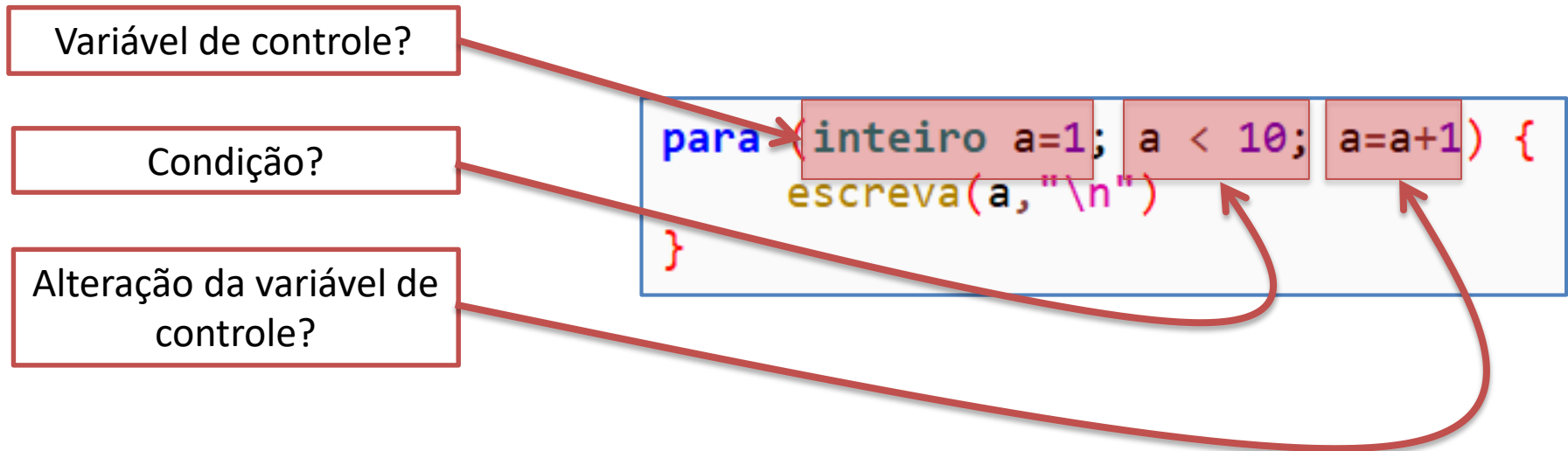
Condição?

Alteração da variável de controle?

```
para (inteiro a=1; a < 10; a=a+1) {  
    escreva(a, "\n")  
}
```

Estruturas de repetição

- Comando “para”
 - Por exemplo, veja o seguinte trecho de código:



Estruturas de repetição

- **Exemplo**

2. Faça uma função “somaIntervalo” que recebe os limites de um intervalo de números inteiros e retorna a soma de todos eles (inclusive os limites).

Crie também uma função “inicio” que recebe dois números inteiros do usuário e escreve na tela o resultado da função “somaIntervalo”.

Exemplo:

`somaIntervalo(2, 5)` deve retornar: $2 + 3 + 4 + 5 = 14$

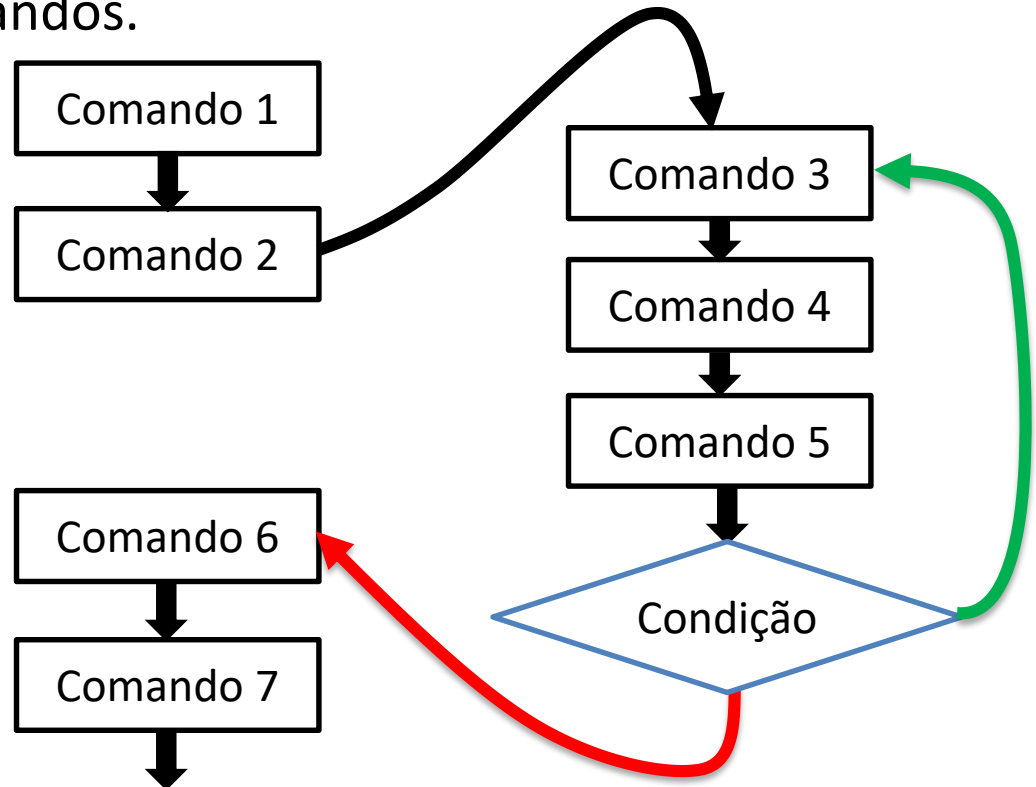
Obs: Utilize o comando “para” para criar a função.

Estruturas de repetição

- Comando “faca-enquanto”

- O teste da condição é realizado no final do bloco, após a primeira execução dos seus comandos.

- O programa irá executar normalmente até encontrar o bloco “faca-enquanto”.

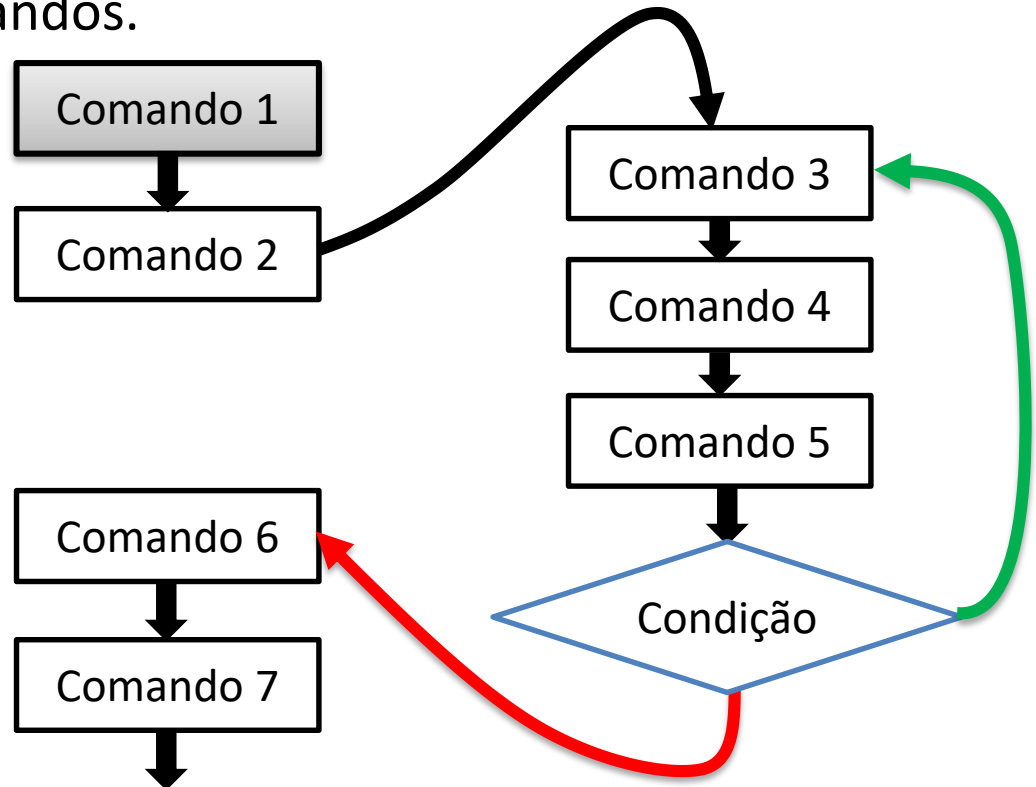


Estruturas de repetição

- Comando “faca-enquanto”

- O teste da condição é realizado no final do bloco, após a primeira execução dos seus comandos.

- O programa irá executar normalmente até encontrar o bloco “faca-enquanto”.

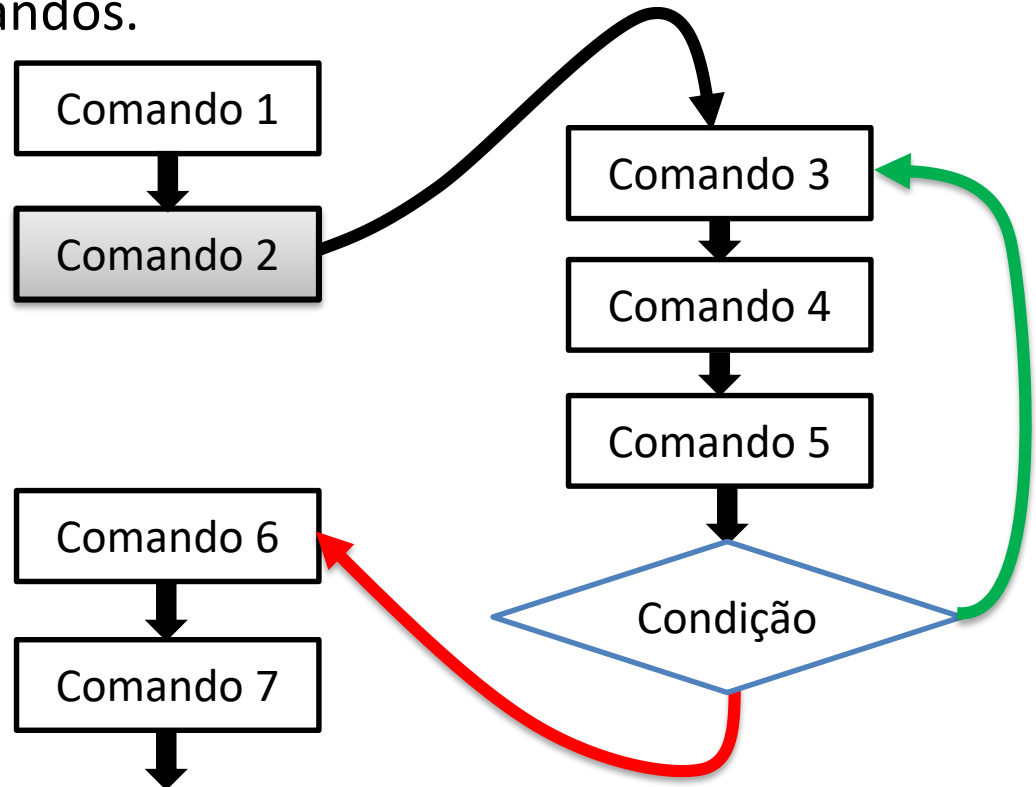


Estruturas de repetição

- Comando “faca-enquanto”

- O teste da condição é realizado no final do bloco, após a primeira execução dos seus comandos.

- O programa irá executar normalmente até encontrar o bloco “faca-enquanto”.

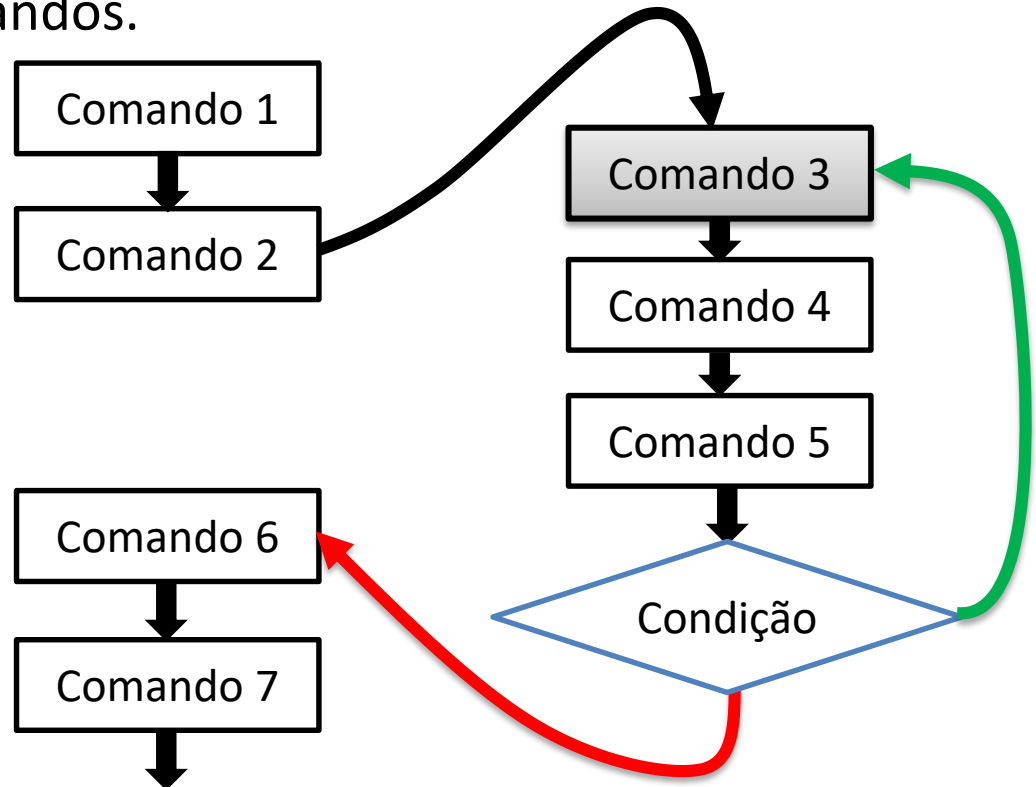


Estruturas de repetição

- **Comando “faca-enquanto”**

- O teste da condição é realizado no final do bloco, após a primeira execução dos seus comandos.

- O programa irá executar normalmente até encontrar o bloco “**faca-enquanto**”.
- O bloco será executado ao menos uma vez, até encontrar a condição.

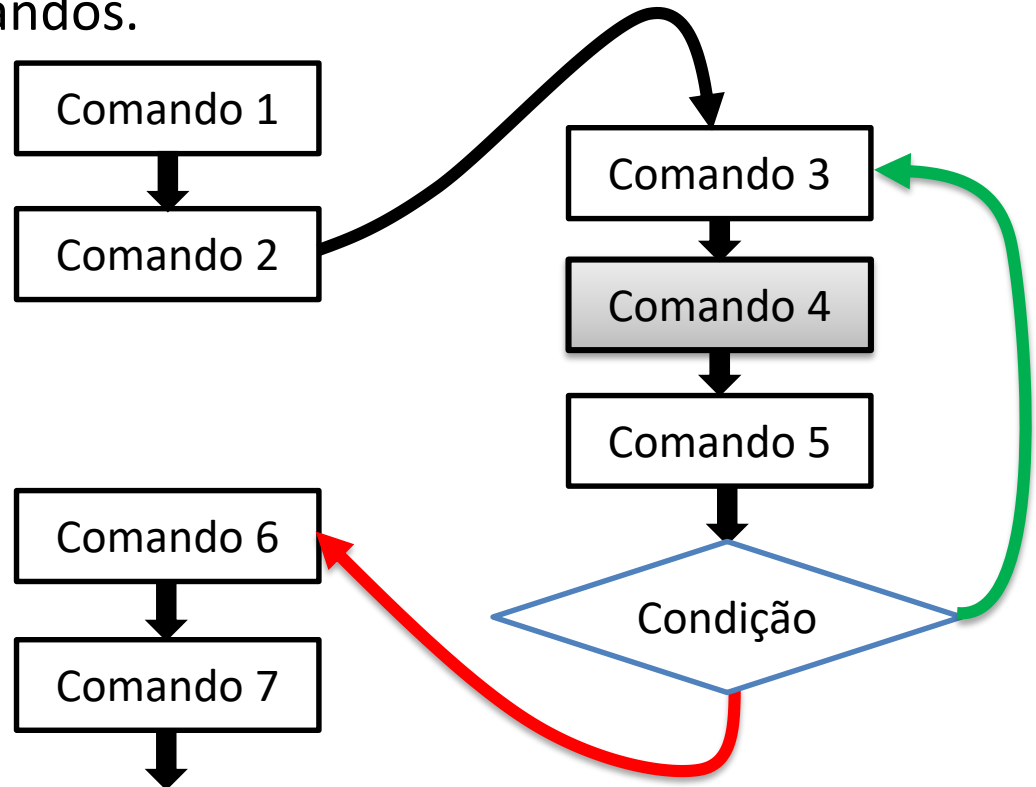


Estruturas de repetição

- **Comando “faca-enquanto”**

- O teste da condição é realizado no final do bloco, após a primeira execução dos seus comandos.

- O programa irá executar normalmente até encontrar o bloco “**faca-enquanto**”.
- O bloco será executado ao menos uma vez, até encontrar a condição.

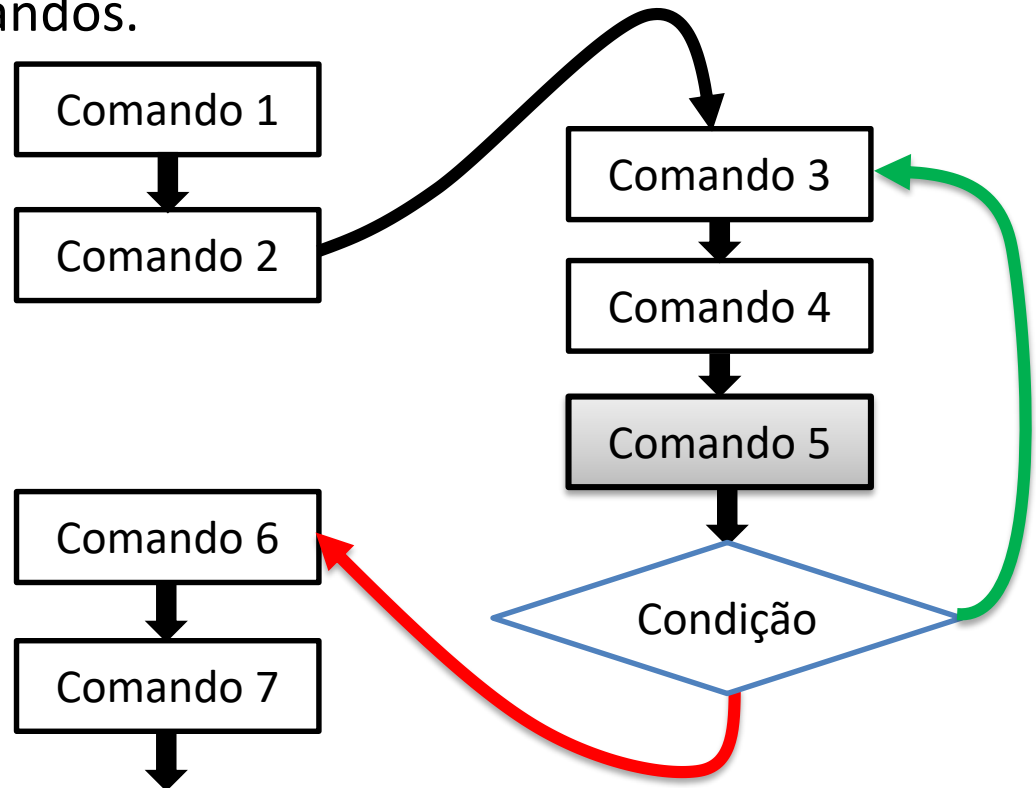


Estruturas de repetição

- **Comando “faca-enquanto”**

- O teste da condição é realizado no final do bloco, após a primeira execução dos seus comandos.

- O programa irá executar normalmente até encontrar o bloco “**faca-enquanto**”.
- O bloco será executado ao menos uma vez, até encontrar a condição.

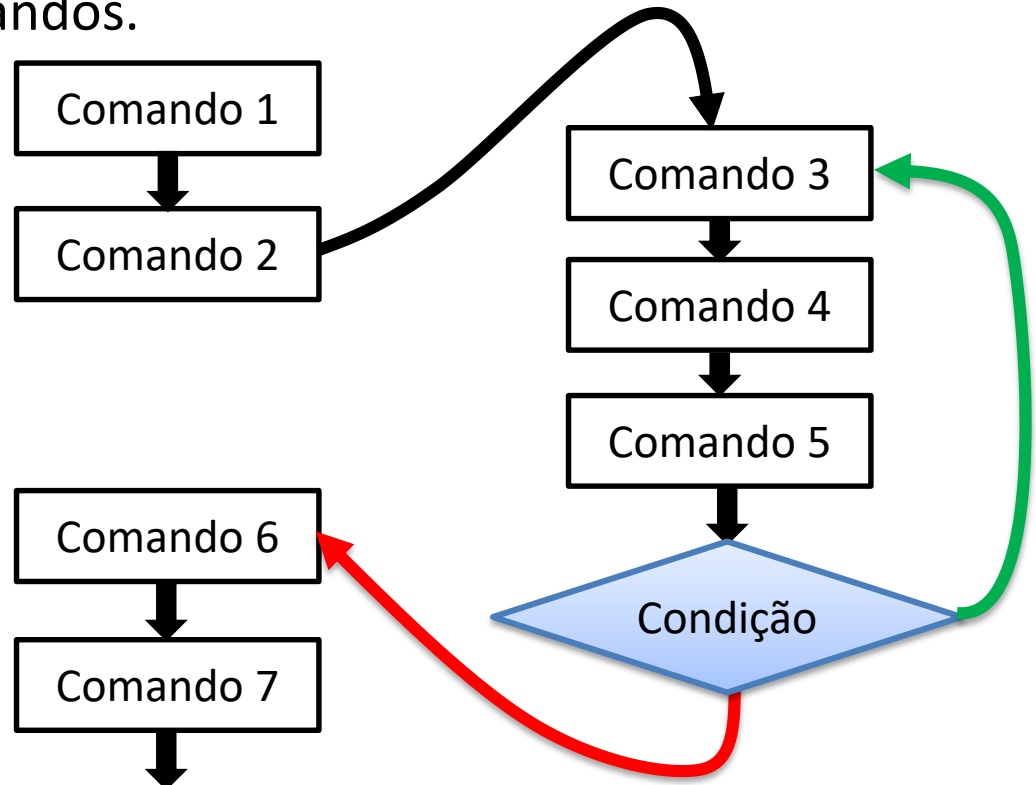


Estruturas de repetição

- **Comando “faca-enquanto”**

- O teste da condição é realizado no final do bloco, após a primeira execução dos seus comandos.

- O programa irá executar normalmente até encontrar o bloco “**faca-enquanto**”.
- O bloco será executado ao menos uma vez, até encontrar a condição.
- Enquanto a condição for verdadeira, o bloco será repetido.

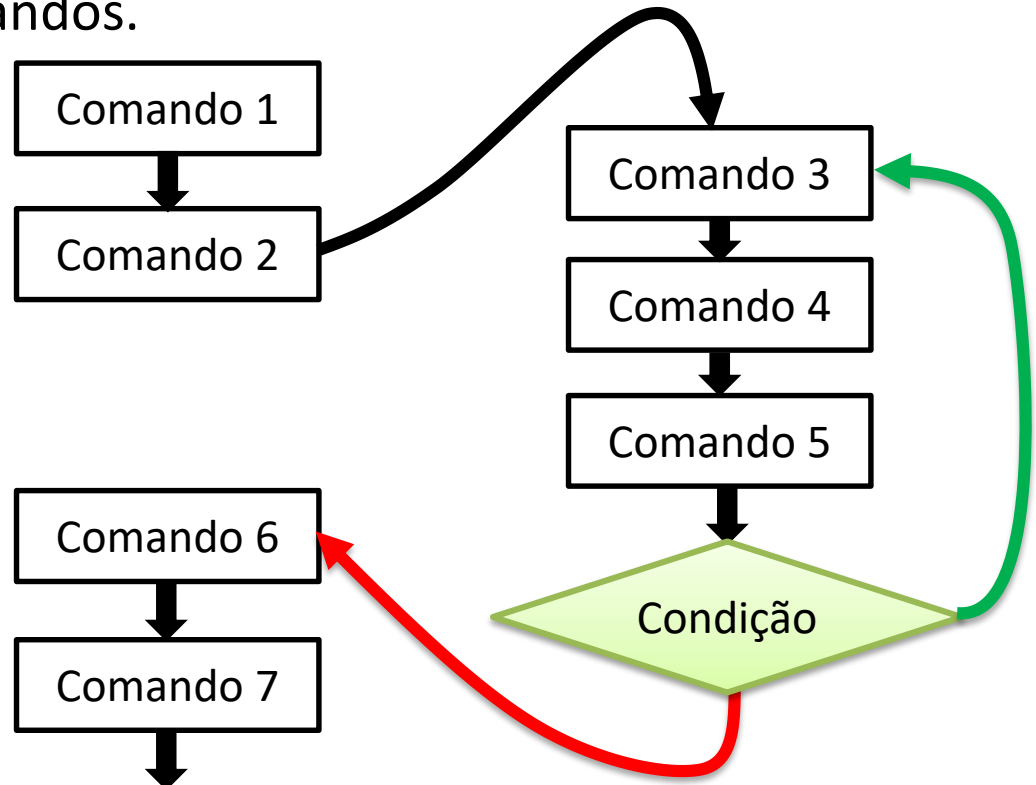


Estruturas de repetição

- **Comando “faca-enquanto”**

- O teste da condição é realizado no final do bloco, após a primeira execução dos seus comandos.

- O programa irá executar normalmente até encontrar o bloco “**faca-enquanto**”.
- O bloco será executado ao menos uma vez, até encontrar a condição.
- Enquanto a condição for verdadeira, o bloco será repetido.

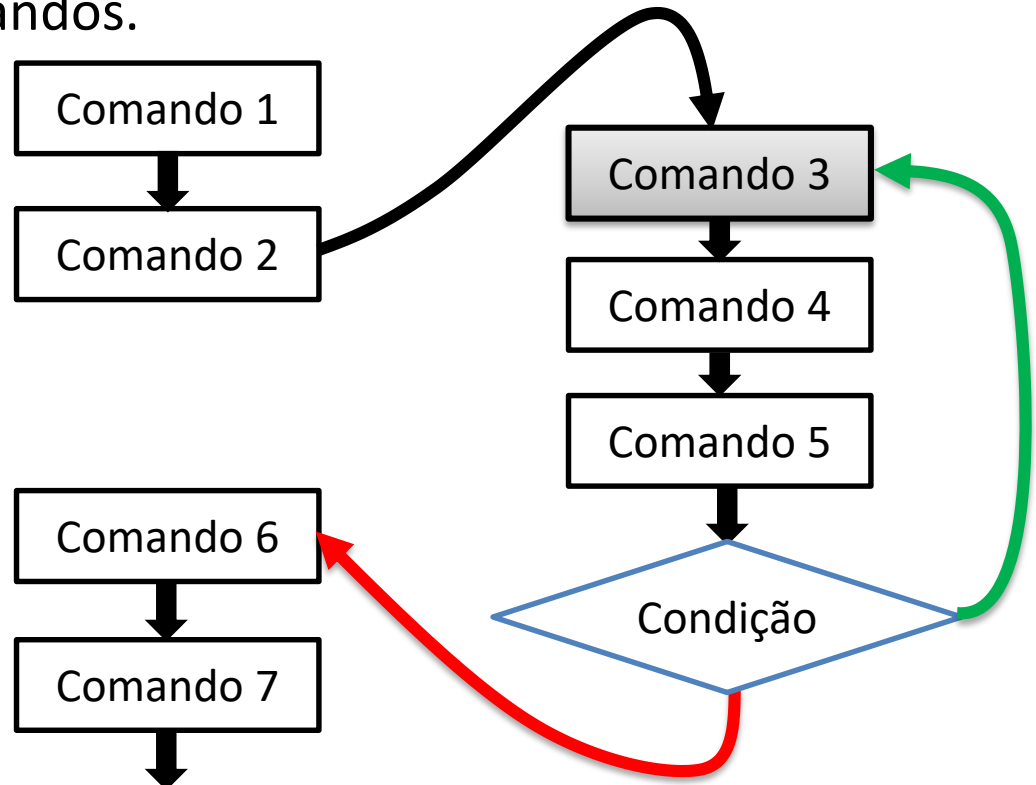


Estruturas de repetição

- **Comando “faca-enquanto”**

- O teste da condição é realizado no final do bloco, após a primeira execução dos seus comandos.

- O programa irá executar normalmente até encontrar o bloco “**faca-enquanto**”.
- O bloco será executado ao menos uma vez, até encontrar a condição.
- Enquanto a condição for verdadeira, o bloco será repetido.

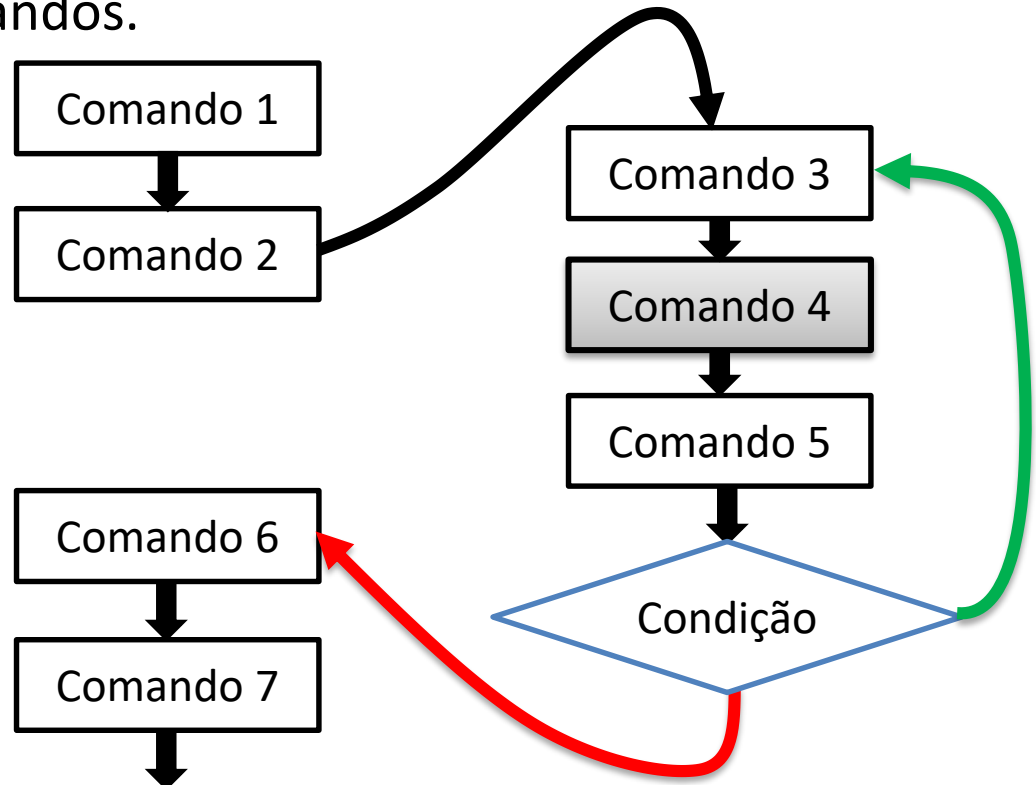


Estruturas de repetição

- **Comando “faca-enquanto”**

- O teste da condição é realizado no final do bloco, após a primeira execução dos seus comandos.

- O programa irá executar normalmente até encontrar o bloco “**faca-enquanto**”.
- O bloco será executado ao menos uma vez, até encontrar a condição.
- Enquanto a condição for verdadeira, o bloco será repetido.

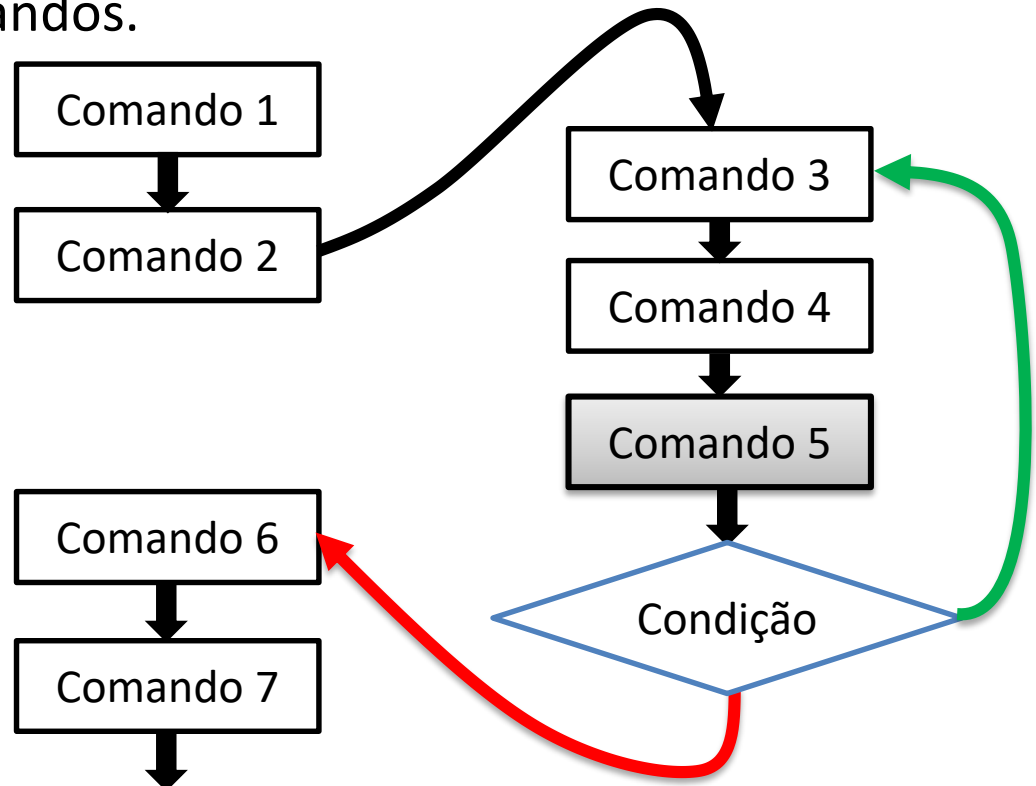


Estruturas de repetição

- **Comando “faca-enquanto”**

- O teste da condição é realizado no final do bloco, após a primeira execução dos seus comandos.

- O programa irá executar normalmente até encontrar o bloco “**faca-enquanto**”.
- O bloco será executado ao menos uma vez, até encontrar a condição.
- Enquanto a condição for verdadeira, o bloco será repetido.

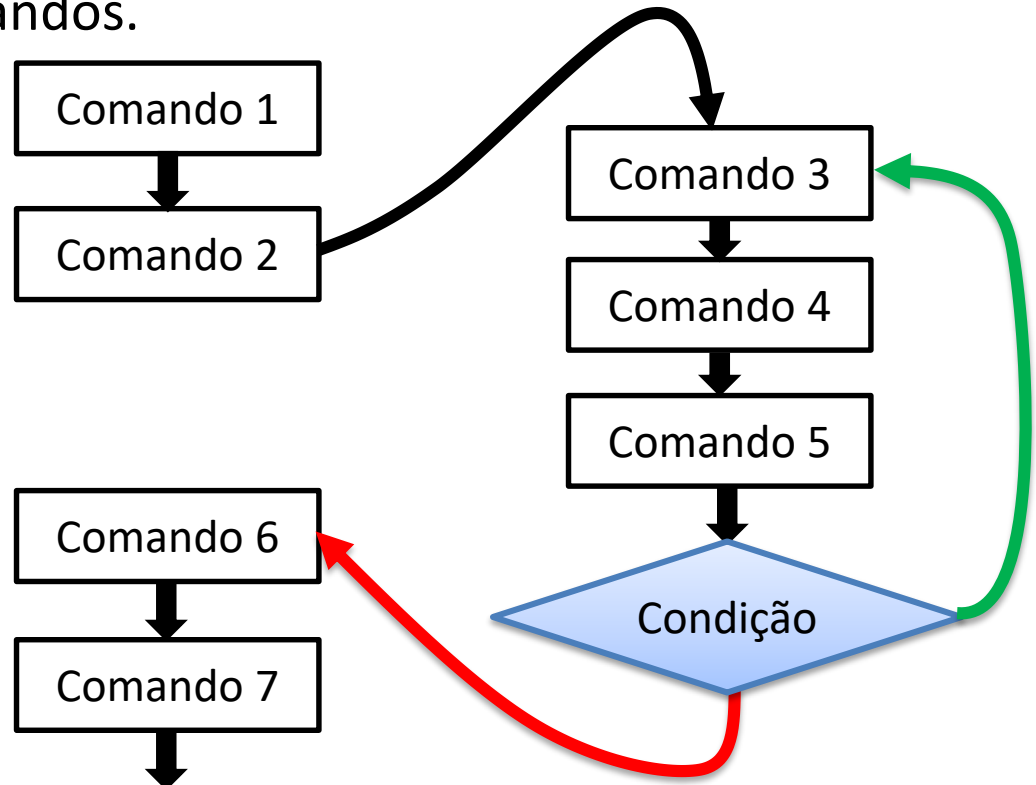


Estruturas de repetição

- **Comando “faca-enquanto”**

- O teste da condição é realizado no final do bloco, após a primeira execução dos seus comandos.

- O programa irá executar normalmente até encontrar o bloco “**faca-enquanto**”.
- O bloco será executado ao menos uma vez, até encontrar a condição.
- Enquanto a condição for verdadeira, o bloco será repetido.

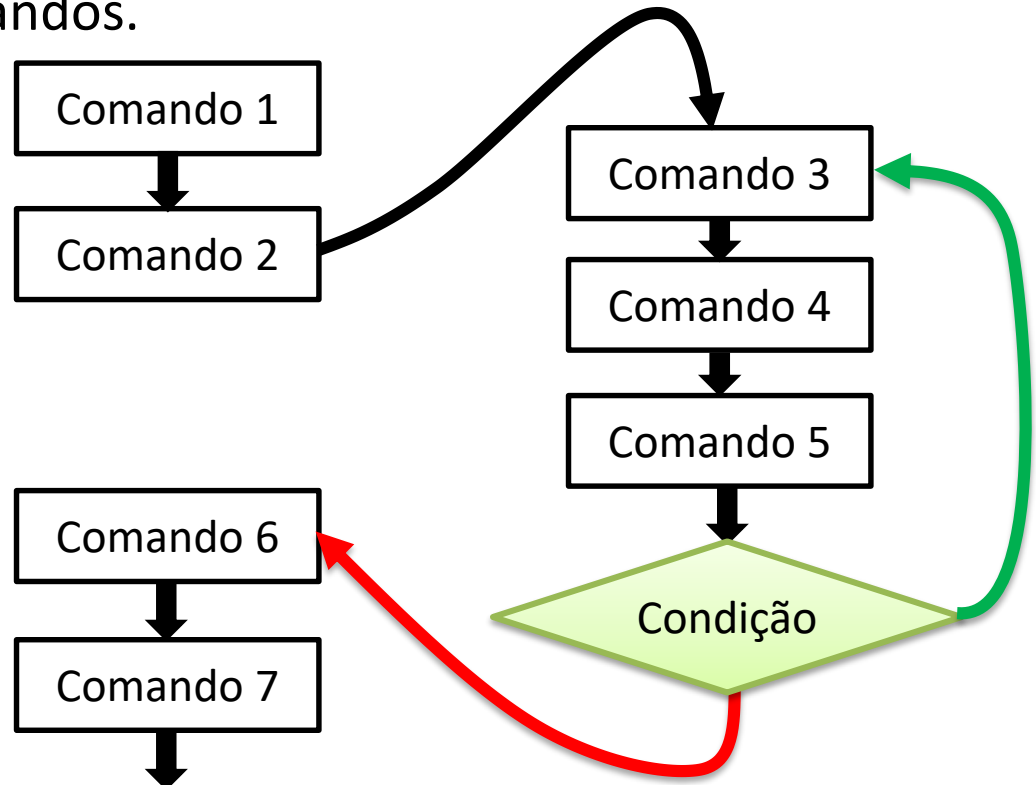


Estruturas de repetição

- **Comando “faca-enquanto”**

- O teste da condição é realizado no final do bloco, após a primeira execução dos seus comandos.

- O programa irá executar normalmente até encontrar o bloco “**faca-enquanto**”.
- O bloco será executado ao menos uma vez, até encontrar a condição.
- Enquanto a condição for verdadeira, o bloco será repetido.

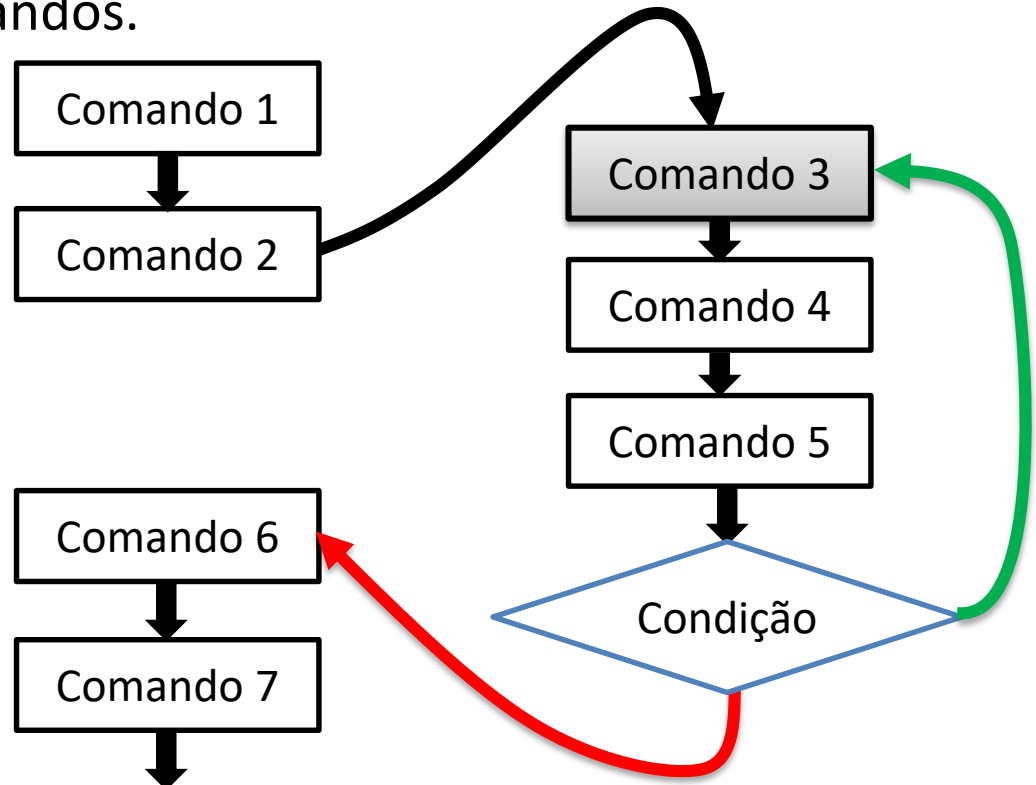


Estruturas de repetição

- **Comando “faca-enquanto”**

- O teste da condição é realizado no final do bloco, após a primeira execução dos seus comandos.

- O programa irá executar normalmente até encontrar o bloco “**faca-enquanto**”.
- O bloco será executado ao menos uma vez, até encontrar a condição.
- Enquanto a condição for verdadeira, o bloco será repetido.

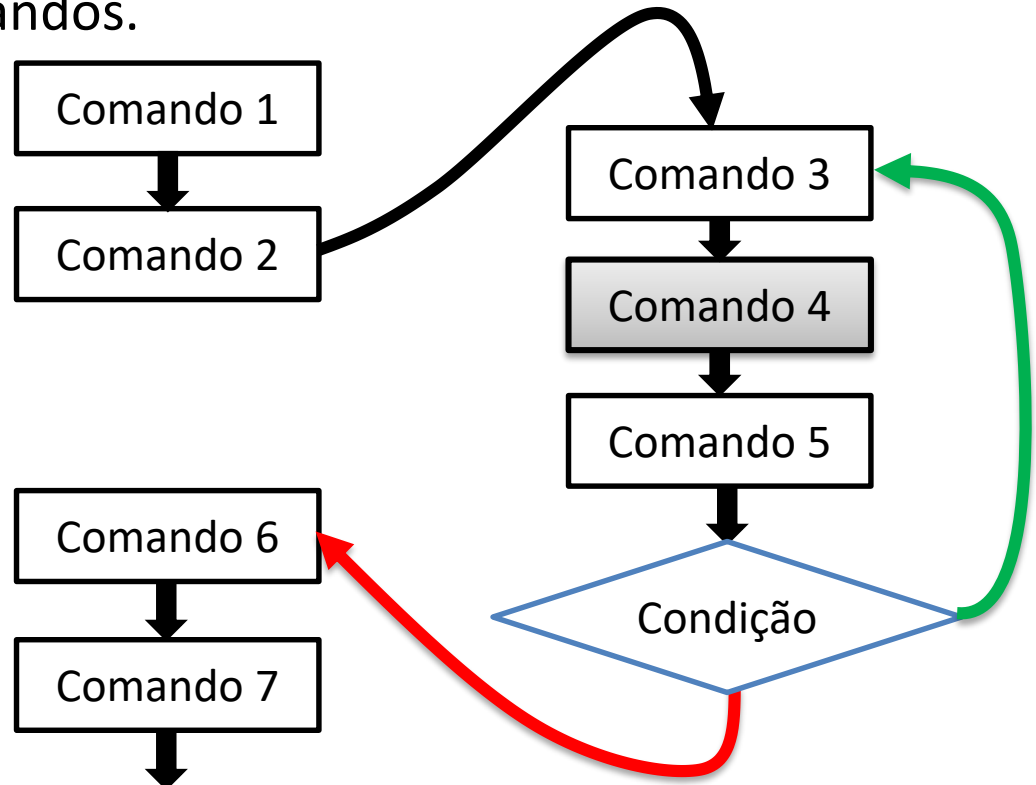


Estruturas de repetição

- **Comando “faca-enquanto”**

- O teste da condição é realizado no final do bloco, após a primeira execução dos seus comandos.

- O programa irá executar normalmente até encontrar o bloco “**faca-enquanto**”.
- O bloco será executado ao menos uma vez, até encontrar a condição.
- Enquanto a condição for verdadeira, o bloco será repetido.

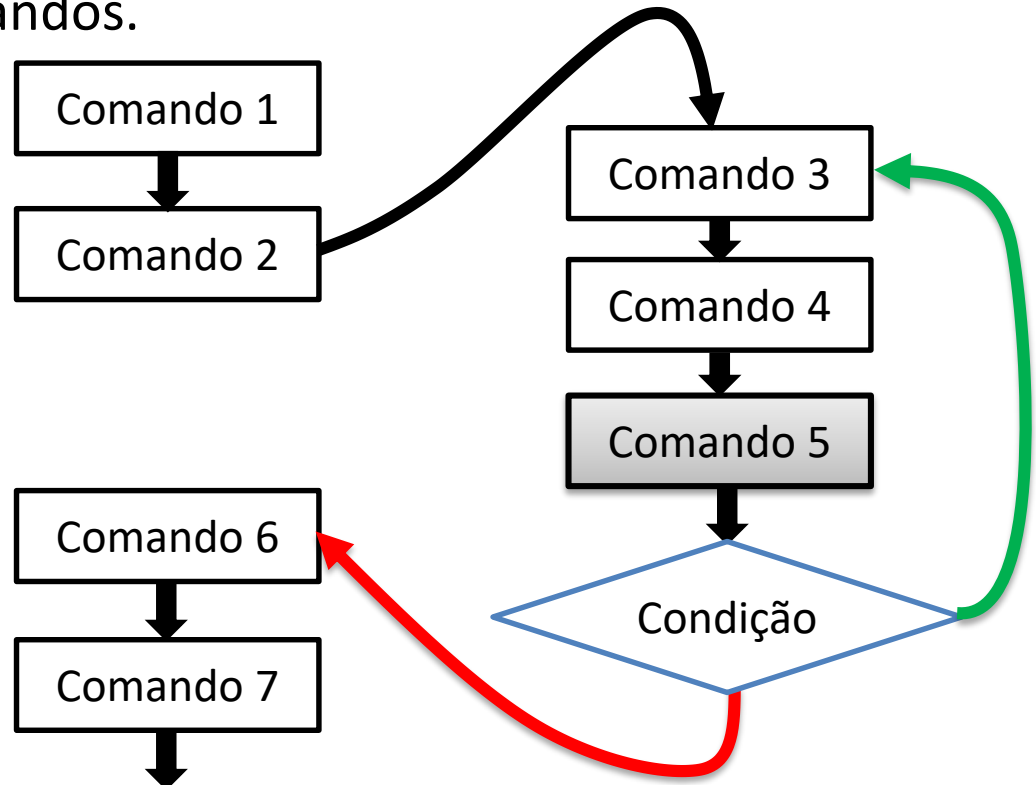


Estruturas de repetição

- **Comando “faca-enquanto”**

- O teste da condição é realizado no final do bloco, após a primeira execução dos seus comandos.

- O programa irá executar normalmente até encontrar o bloco “**faca-enquanto**”.
- O bloco será executado ao menos uma vez, até encontrar a condição.
- Enquanto a condição for verdadeira, o bloco será repetido.

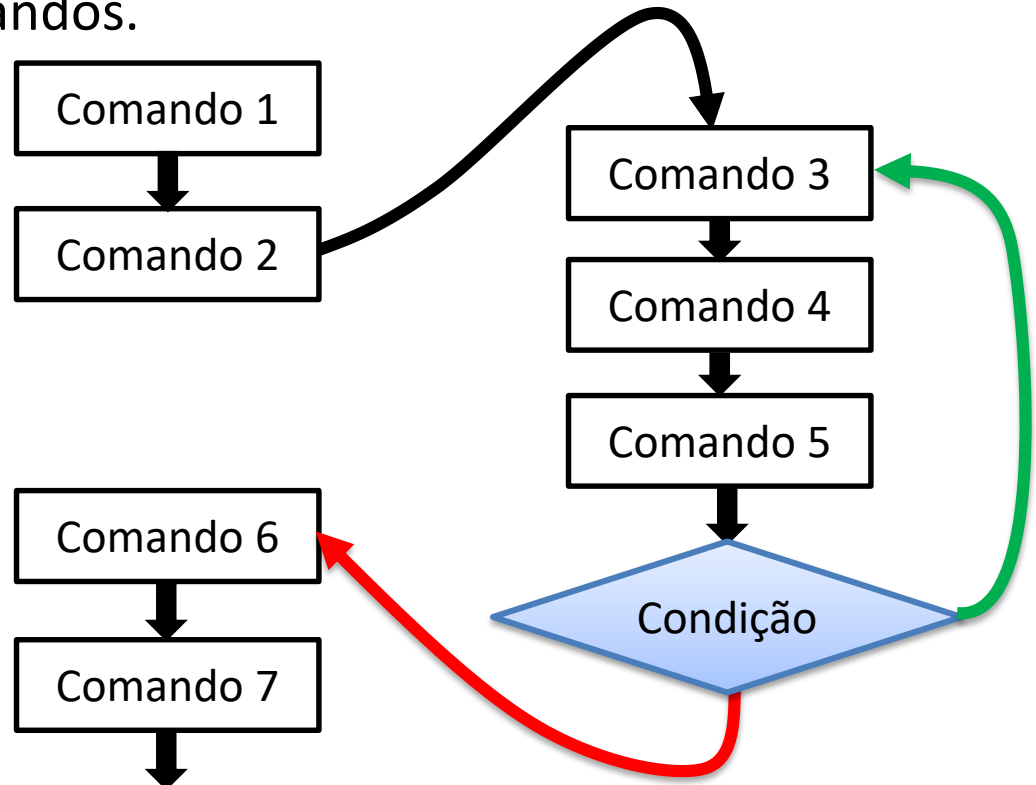


Estruturas de repetição

- **Comando “faca-enquanto”**

- O teste da condição é realizado no final do bloco, após a primeira execução dos seus comandos.

- O programa irá executar normalmente até encontrar o bloco “**faca-enquanto**”.
- O bloco será executado ao menos uma vez, até encontrar a condição.
- Enquanto a condição for verdadeira, o bloco será repetido.

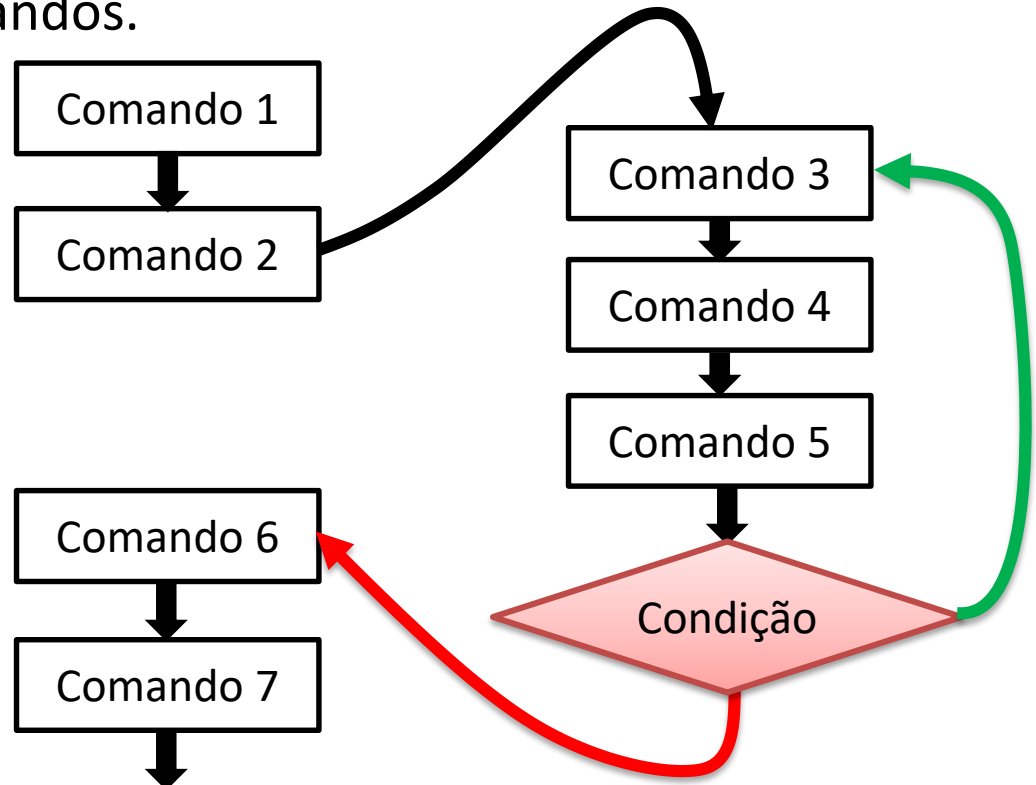


Estruturas de repetição

- **Comando “faca-enquanto”**

- O teste da condição é realizado no final do bloco, após a primeira execução dos seus comandos.

- O programa irá executar normalmente até encontrar o bloco “**faca-enquanto**”.
- O bloco será executado ao menos uma vez, até encontrar a condição.
- Enquanto a condição for verdadeira, o bloco será repetido.

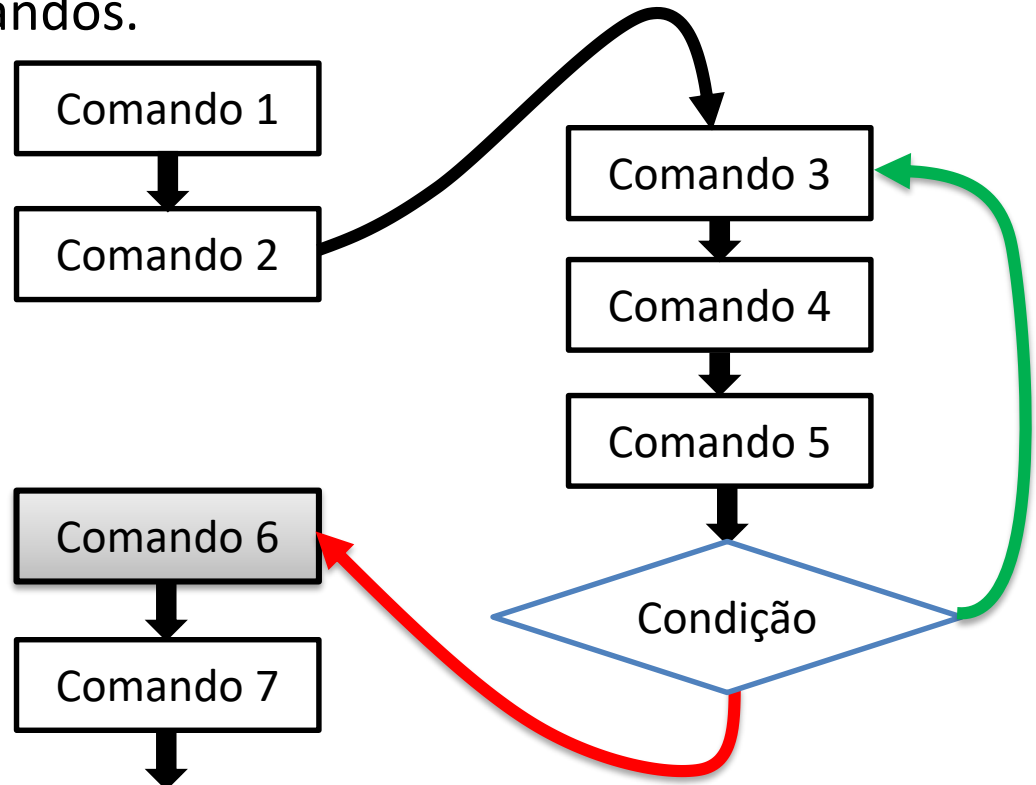


Estruturas de repetição

- **Comando “faca-enquanto”**

- O teste da condição é realizado no final do bloco, após a primeira execução dos seus comandos.

- O programa irá executar normalmente até encontrar o bloco “**faca-enquanto**”.
- O bloco será executado ao menos uma vez, até encontrar a condição.
- Enquanto a condição for verdadeira, o bloco será repetido.

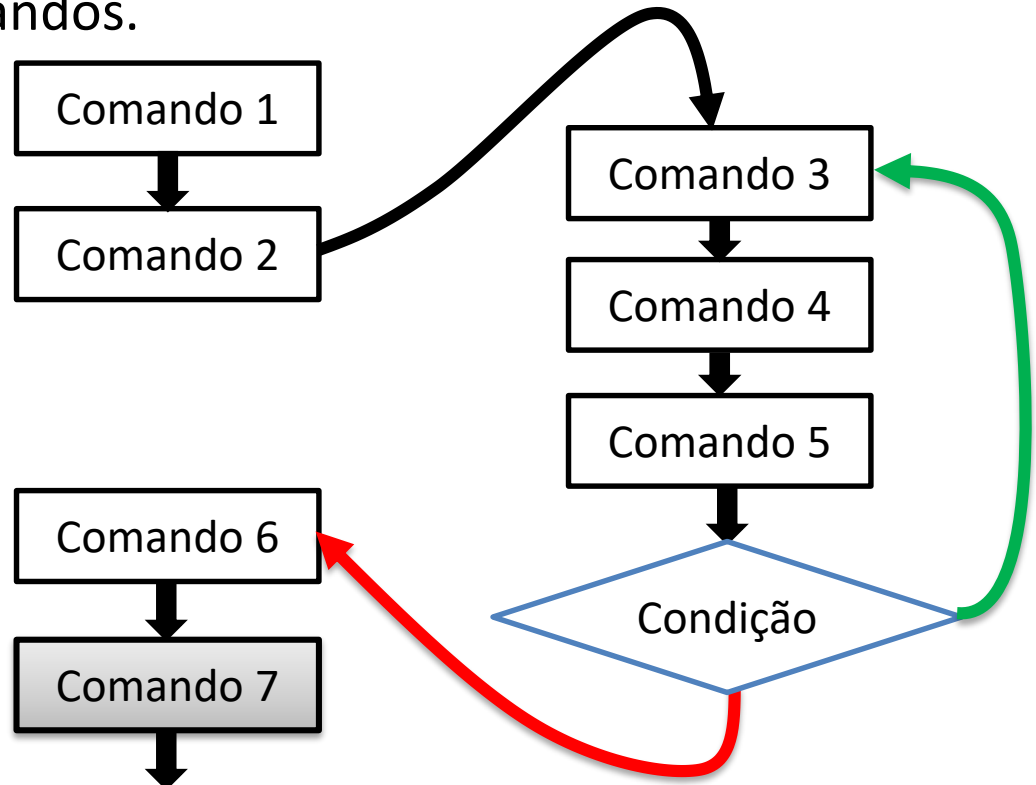


Estruturas de repetição

- **Comando “faca-enquanto”**

- O teste da condição é realizado no final do bloco, após a primeira execução dos seus comandos.

- O programa irá executar normalmente até encontrar o bloco “**faca-enquanto**”.
- O bloco será executado ao menos uma vez, até encontrar a condição.
- Enquanto a condição for verdadeira, o bloco será repetido.



Estruturas de repetição

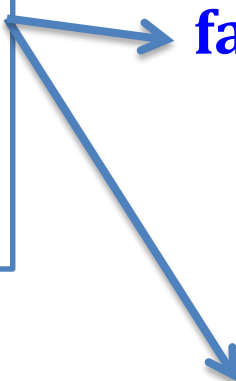
- Comando “faca-enquanto”
 - Em português, a sintaxe para o comando “faca-enquanto” é:

```
faça {  
    comando1  
    comando2  
    comando3  
    ...  
} enquanto(condição)
```

Estruturas de repetição

- Comando “faca-enquanto”
 - Em português, a sintaxe para o comando “faca-enquanto” é:

Palavras reservadas
que indicam um laço
de repetição “faca-
enquanto”.



```
faça {  
    comando1  
    comando2  
    comando3  
    ...  
} enquanto(condição)
```

Estruturas de repetição

- Comando “faca-enquanto”
 - Em português, a sintaxe para o comando “faca-enquanto” é:

Palavras reservadas
que indicam um laço
de repetição “faca-
enquanto”.

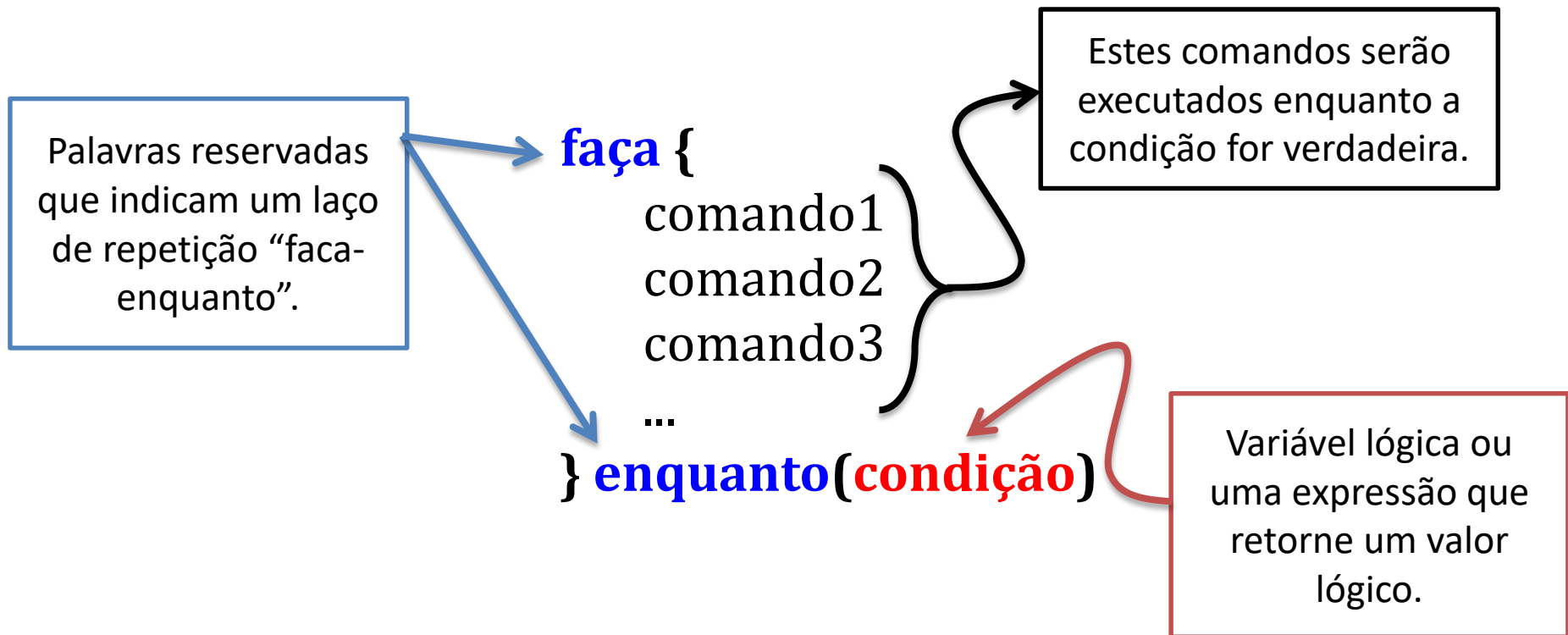
faça {
 comando1
 comando2
 comando3
 ...
} **enquanto**(**condição**)

Variável lógica ou
uma expressão que
retorne um valor
lógico.

Estruturas de repetição

- Comando “faca-enquanto”

- Em português, a sintaxe para o comando “faca-enquanto” é:



Estruturas de repetição

- Comando “faca-enquanto”
 - Por exemplo, veja o seguinte trecho de código:

```
inteiro a = 1

faca {
    escreva(a, "\n")
    a = a + 1
} enquanto (a < 10)
```

- O que será impresso na tela se executarmos esse código?

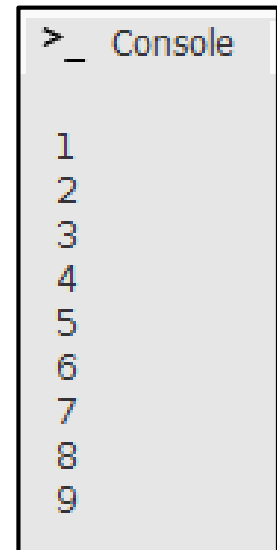
Estruturas de repetição

- Comando “faca-enquanto”

- Por exemplo, veja o seguinte trecho de código:

```
inteiro a = 1

faca {
    escreva(a, "\n")
    a = a + 1
} enquanto (a < 10)
```



```
>_ Console

1
2
3
4
5
6
7
8
9
```

- O que será impresso na tela se executarmos esse código?

Estruturas de repetição

- Comando “faca-enquanto”

- Por exemplo, veja o seguinte trecho de código:

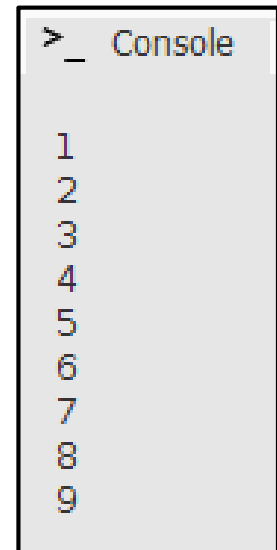
Variável de controle?

Condição?

Alteração da variável de controle?

```
inteiro a = 1

faca {
    escreva(a, "\n")
    a = a + 1
} enquanto (a < 10)
```



```
>_ Console

1
2
3
4
5
6
7
8
9
```

- O que será impresso na tela se executarmos esse código?

Estruturas de repetição

- Comando “faca-enquanto”

- Por exemplo, veja o seguinte trecho de código:

Variável de controle?

Condição?

Alteração da variável de controle?

```
inteiro a = 1

faca {
    escreva(a, "\n")
    a = a + 1
} enquanto (a < 10)
```

```
>_ Console
1
2
3
4
5
6
7
8
9
```

- O que será impresso na tela se executarmos esse código?

Estruturas de repetição

- Comando “faca-enquanto”

- Por exemplo, veja o seguinte trecho de código:

Variável de controle?

```
inteiro a = 1
```

Condição?

```
faca {  
    escreva(a, "\n")  
    a = a + 1  
} enquanto (a < 10)
```

Alteração da variável de controle?

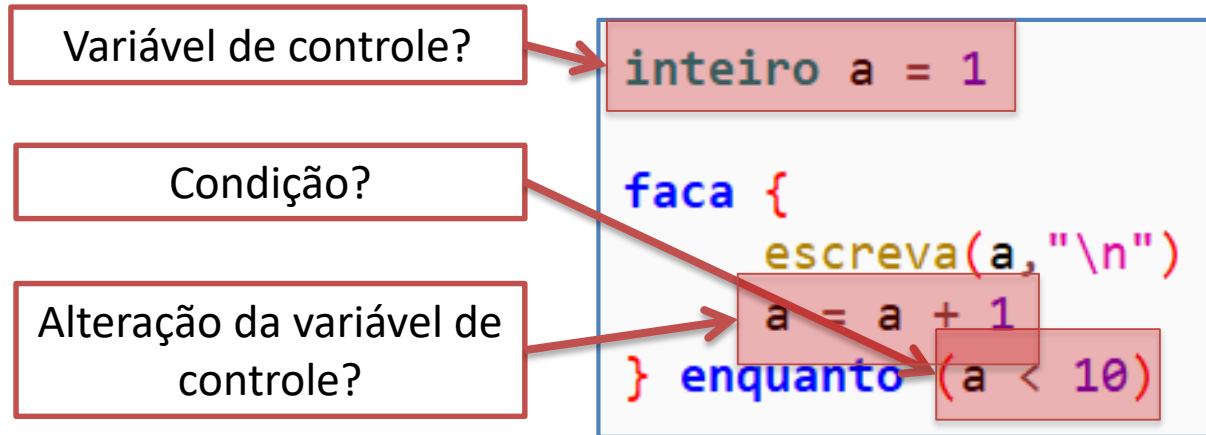
```
>_ Console  
  
1  
2  
3  
4  
5  
6  
7  
8  
9
```

- O que será impresso na tela se executarmos esse código?

Estruturas de repetição

- Comando “faca-enquanto”

- Por exemplo, veja o seguinte trecho de código:



```
>_ Console  
  
1  
2  
3  
4  
5  
6  
7  
8  
9
```

- O que será impresso na tela se executarmos esse código?

- Comando “faca-enquanto”

- A estrutura “faca-enquanto” é particularmente **útil** em casos em que precisamos **colocar um programa em loop** até que o **usuário decida encerrá-lo**.
- Exemplo:

```
real n1, n2, media
caracter resp
faca{
    escreva("Cálculo da média entre dois números\n")
    escreva("Digite o primeiro número: ")
    leia(n1)
    escreva("Digite o segundo número: ")
    leia(n2)
    media = (n1+n2)/2
    escreva("A média entre ", n1, " e ", n2, " é igual a ", media, "...\n\n")
    escreva("Deseja repetir o programa?\n")
    escreva("Entre [S] para sim ou outra tecla para não: ")
    leia(resp)
} enquanto ((resp == 'S') ou (resp == 's'))
```


Estruturas de repetição

- **Exercícios**

1. Faça um programa para gerar e exibir os números inteiros de 20 até 10, decrescendo de 1 em 1.
2. Faça uma função para receber um número inteiro positivo n , somar todos os números compreendidos entre 1 e n , e retornar o resultado obtido. Teste sua função com uma função “início”.
3. Faça um programa para ler a quantidade n de números que devam ser fornecidos e, em seguida, ler também estes números. Este algoritmo deve, a seguir, mostrar o maior deles.

Estruturas de repetição

- **Exercícios**

4. Um número natural é primo, por definição, se ele não tem divisores, exceto 1 e ele próprio. Escreva uma função para receber um número inteiro e determinar se ele é ou não primo. Teste sua função com uma função “início”.

5. Escreva uma função para calcular o fatorial de um número inteiro recebido, sabendo que:

$$n! = 1 \times 2 \times \cdots \times (n - 1) \times n$$
$$0! = 1$$

Teste sua função com uma função “início”.

Estruturas de repetição

- **Exercícios**

6. Faça uma função que recebe um número inteiro e determina o número de dígitos que ele possui.

7. Faça uma função que recebe um número inteiro e retorna a soma de seus dígitos pares.

8. Faça uma função que recebe um número inteiro e retorna o seu maior dígito ímpar.

9. Um número é **deboriano** se a soma de seus dígitos pares é maior do que o seu maior dígito ímpar. Escreva uma função para determinar se um número é **deboriano**.

Estruturas de repetição

• Exercícios

10. Um número é **maluco** se o seu dígito mais significativo for igual ao seu dígito menos significativo e, além disso, o total de seus dígitos é igual ao seu dígito menos significativo. Escreva uma função para determinar se um número é **maluco**.

11. Um número é **podre** se só possui dígitos ímpares que não sejam múltiplos de cinco. Escreva uma função para determinar se um número é **podre**.

12. Um número é **suíno** se não possui os dígitos 1 e 7 e o seu dígito menos significativo é maior do que o seu dígito mais significativo. Escreva uma função para determinar se um número é **suíno**.

13. Um número é **perdedor** se é podre, suíno e maluco. Escreva uma função para determinar se um número é **perdedor**.

- **Exercícios**

14. A sequência de Fibonacci é formada inicialmente pelos valores 0 e 1. A partir de então, cada novo elemento desta sequência é obtido pela soma dos dois elementos imediatamente anteriores (0, 1, 1, 2, 3, 5, 8, 13, ...). Faça uma função para receber um número n, calcular e retornar o n-ésimo termo da sequência de Fibonacci. Teste sua função com uma função “início”.

15. Uma financeira empresta dinheiro a seus clientes sob pena de juros fixos a serem cobrados a cada mês. Veja um exemplo em que um cliente faz um empréstimo de R\$10.000,00 para pagá-los em 3 meses com juros fixos de 1,0 %.

Mês 1: $R\$10.000,00 + R\$10.000,00 \times 1\% = R\$10.100,00$

Mês 2: $R\$10.100,00 + R\$10.100,00 \times 1\% = R\$10.201,00$

Mês 3: $R\$10.201,00 + R\$10.201,00 \times 1\% = R\$10.303,01$

Faça uma função para receber o valor a ser emprestado, a taxa de juros que será cobrada mensalmente e o período em meses para o cliente pagar sua dívida e, em seguida, calcular e retornar o valor a ser pago pelo cliente ao final do prazo estabelecido no empréstimo. Teste sua função com uma função “início”.

- **Exercícios**

16. Escreva um programa que apresente o menu de opções abaixo:

1 – SAUDAÇÃO 2 – BRONCA 3 – FELICITAÇÃO 0 – FIM

O programa deve ler a opção do usuário e exibir, para cada opção, a respectiva mensagem:

1 - Olá. Como vai?

2 - Vamos estudar mais.

3 - Meus Parabéns!

0 - Fim de serviço.

Enquanto a opção for diferente de 0 (zero) deve-se continuar apresentando as opções. Use o comando “faca-enquanto” como estrutura de repetição.

Estruturas de repetição

FIM