



Algoritmos e Lógica de Programação I

Estrutura de Repetição

Prof. MSc. Rafael Staiger Bressan
rafael.bressan@unicesumar.edu.br



Cronograma

- Estrutura de Repetição
- Estrutura FOR
- Estrutura WHILE
- Estrutura DO-WHILE
- Exercícios



Estrutura de Repetição

- O real poder dos computadores está na sua **habilidade para repetir uma operação ou uma serie de operações muitas vezes.**
- Este repetição chamada laços (**loop**) é um dos conceitos básicos da programação estruturada



Estrutura de Repetição

- Uma estrutura de repetição permite que uma sequência de comandos seja executada repetidamente, enquanto determinadas condições são satisfeitas.



Estrutura de Repetição

- Essas condições são representadas por expressões lógicas (como, por exemplo, $A > B$; $C == 3$; $\text{Letra} == \text{'a'}$)
 - Repetição Contada (FOR)
 - Repetição com Teste no Início (WHILE)
 - Repetição com Teste no Final (DO-WILE)

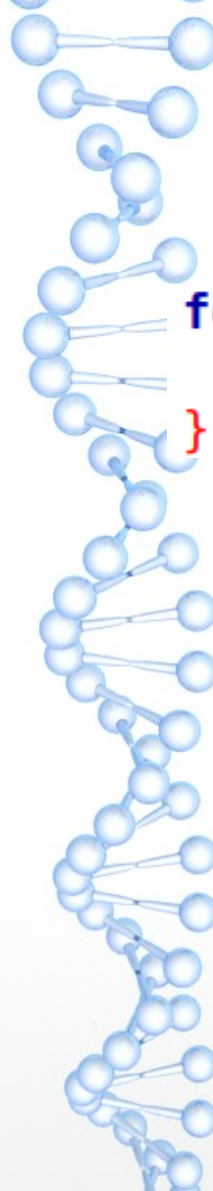


FOR

- O loop ou laço FOR é usado para repetir um comando, ou bloco de comandos, diversas vezes
 - Maior controle sobre o loop.
- Sua forma geral é

```
for("VALOR-INICIAL"; "CONDIÇÃO"; "PASSO-DO-INCREMENTO"){  
    "CODIGOS..."  
}
```

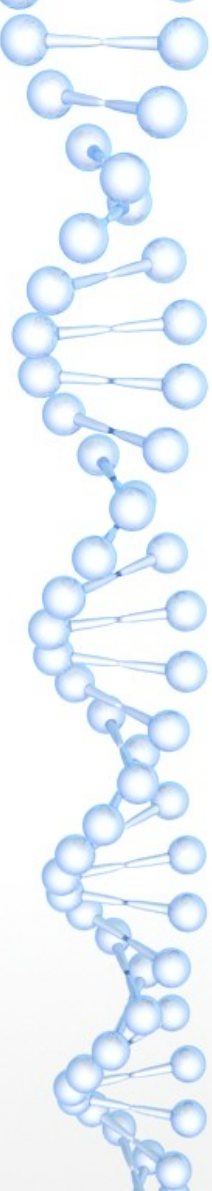
FOR



```
for( "VALOR-INICIAL"; "CONDIÇÃO"; "PASSO-DO-INCREMENTO" ){  
    "CODIGOS..."  
}
```

- **VALOR-INICIAL**: iniciar variáveis (contador).
- **CONDIÇÃO**: avalia a condição. Se verdadeiro, executa comandos do bloco, senão encerra laço.
- **PASSO-DO-INCREMENTO**: ao término do bloco de comandos, incrementa o valor do contador
- **repete o processo até que a condição seja falsa.**

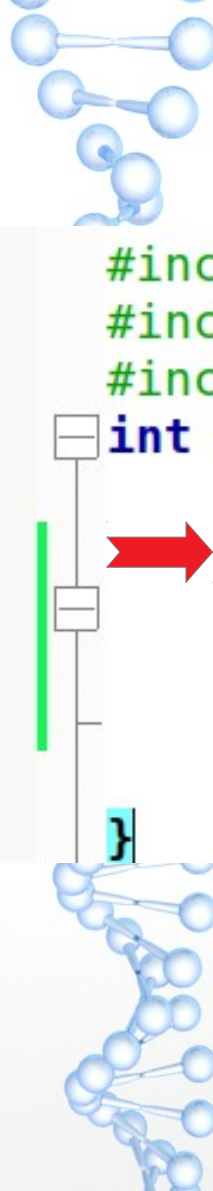
FOR - Exemplo



```
5  #include <stdio.h>
6  #include <stdlib.h>
7  #include <locale.h>
8  int main() {
9      setlocale(LC_ALL, "Portuguese");
10     int i;
11     for(i=0; i<10; i++){
12         printf(" %d \n", i);
13     }
14     return 0;
15 }
```

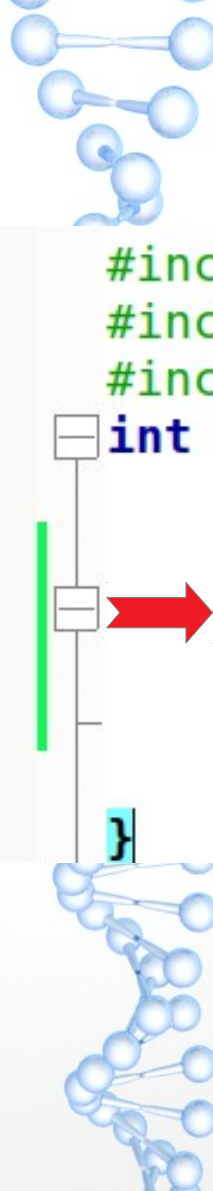

- Adiciona a opção de correção da linguagem.

```
5  #include <stdio.h>
6  #include <stdlib.h>
7  #include <locale.h>
8  int main() {
9      → setlocale(LC_ALL, "Portuguese");
10     int i;
11     for(i=0; i<10; i++){
12         printf(" %d \n", i);
13     }
14     return 0;
15 }
```



```
5  #include <stdio.h>
6  #include <stdlib.h>
7  #include <locale.h>
8  int main() {
9      setlocale(LC_ALL, "Portuguese");
10     int i;
11     for(i=0; i<10; i++){
12         printf(" %d \n", i);
13     }
14     return 0;
15 }
```

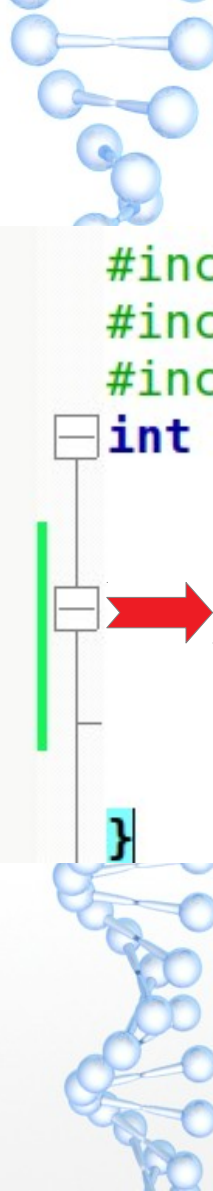
- Declara uma variável do tipo inteira (int) com nome de (i).
- A memória RAM reserva um espaço suficiente para armazenar um número inteiro.
- O valor de (i) até o momento é desconhecido.



```
5  #include <stdio.h>
6  #include <stdlib.h>
7  #include <locale.h>
8  int main() {
9      setlocale(LC_ALL, "Portuguese");
10     int i;
11     for(i=0; i<10; i++){
12         printf(" %d \n", i);
13     }
14     return 0;
15 }
```

A red arrow points from the line number 11 to the initialization part of the for loop, `i=0`, which is enclosed in a green box.

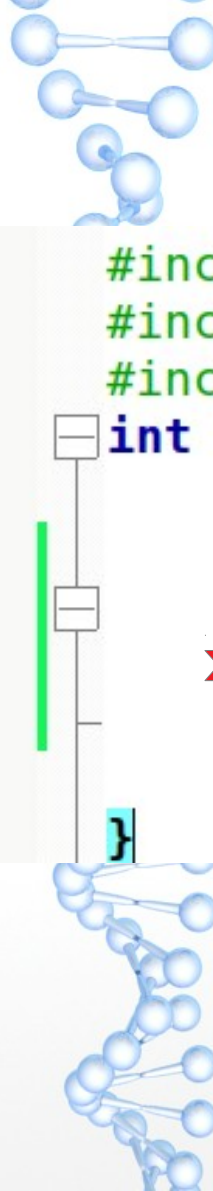
- Na linha 11, (i) recebe 0.
 - `i = 0;`
- Ou seja, a variável (i) nesse momento está com o valor 0 atribuído.



```
5  #include <stdio.h>
6  #include <stdlib.h>
7  #include <locale.h>
8  int main() {
9      setlocale(LC_ALL, "Portuguese");
10     int i;
11     for(i=0; i<10; i++){
12         printf(" %d \n", i);
13     }
14     return 0;
15 }
```

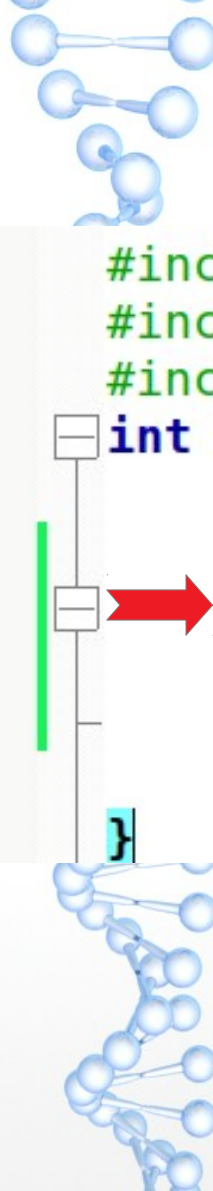
A red arrow points to the condition `i < 10` in the for loop on line 11. A green box highlights the expression `i < 10`.

- `i = 0;`
- Verifica-se a condição proposta.
 - `i < 10` ?
- Se a condição proposta é verdadeira, executa-se os comandos dentro do laço. { }



```
5  #include <stdio.h>
6  #include <stdlib.h>
7  #include <locale.h>
8  int main() {
9      setlocale(LC_ALL, "Portuguese");
10     int i;
11     for(i=0; i<10; i++){
12         printf(" %d \n", i);
13     }
14     return 0;
15 }
```

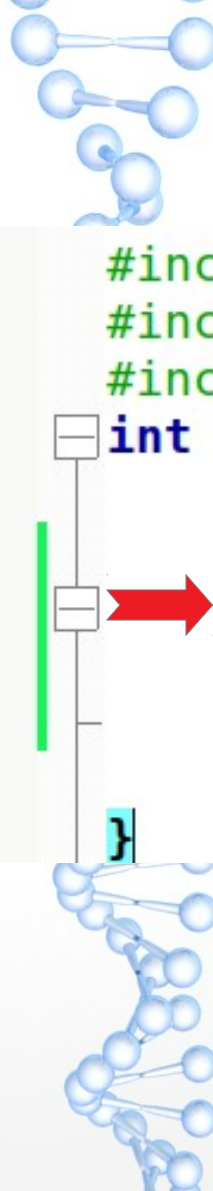
- **i = 0;**
- Verifica-se a condição proposta.
 - **i < 10** ?
- Se a condição proposta é verdadeira, executa-se os comandos dentro do laço. { }
- **Escreva na tela (i) que no momento esta valendo 0.**



```
5  #include <stdio.h>
6  #include <stdlib.h>
7  #include <locale.h>
8  int main() {
9      setlocale(LC_ALL, "Portuguese");
10     int i;
11     for(i=0; i<10; i++){
12         printf(" %d \n", i);
13     }
14     return 0;
15 }
```

A red arrow points to the `i++` expression in the for loop condition on line 11. A green box highlights the `i++` expression. A green vertical bar is on the left margin next to lines 10 and 11.

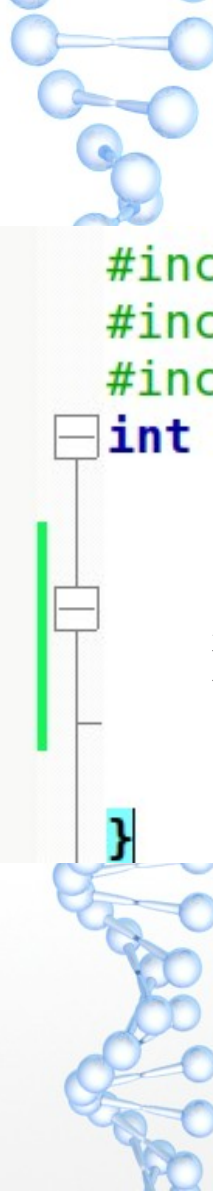
- `i = 0;`
- Após executar os comandos do laço, o passo do incremento ou decremento é executado.
 - `i++;`
- `i++` é igual a `i = i + 1;`
- Como o valor de `i` no momento é 0, ele passa a valer 1.
 - `i = 0 + 1;`



```
5  #include <stdio.h>
6  #include <stdlib.h>
7  #include <locale.h>
8  int main() {
9      setlocale(LC_ALL, "Portuguese");
10     int i;
11     for(i=0; i<10; i++){
12         printf(" %d \n", i);
13     }
14     return 0;
15 }
```

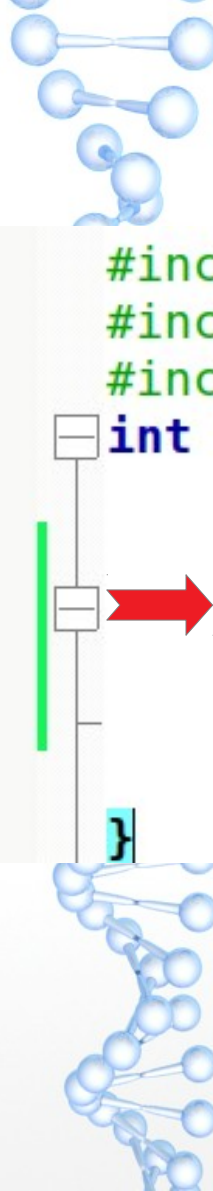
A red arrow points from the opening curly brace of the `main` function to the `for` loop. A green box highlights the condition `i < 10` in the loop header.

- `i = 1;`
- Verifica-se a condição proposta.
 - `i < 10` ?
- Se a condição proposta é verdadeira, executa-se os comandos dentro do laço. { }



```
5  #include <stdio.h>
6  #include <stdlib.h>
7  #include <locale.h>
8  int main() {
9      setlocale(LC_ALL, "Portuguese");
10     int i;
11     for(i=0; i<10; i++){
12         printf(" %d \n", i);
13     }
14     return 0;
15 }
```

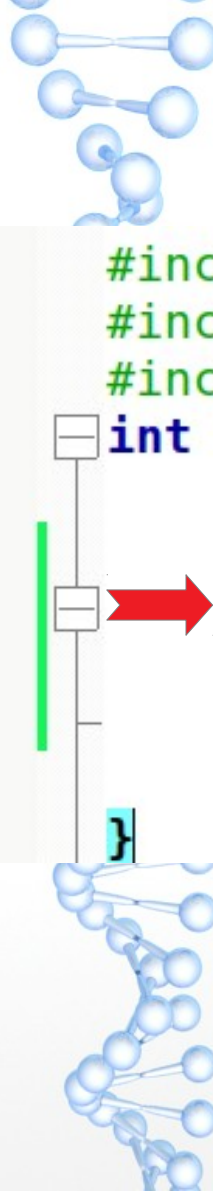
- **i = 1;**
- Verifica-se a condição proposta.
 - **i < 10** ?
- Se a condição proposta é verdadeira, executa-se os comandos dentro do laço. { }
- **Escreva na tela (i) que no momento esta valendo 1.**



```
5  #include <stdio.h>
6  #include <stdlib.h>
7  #include <locale.h>
8  int main() {
9      setlocale(LC_ALL, "Portuguese");
10     int i;
11     for(i=0; i<10; i++){
12         printf(" %d \n", i);
13     }
14     return 0;
15 }
```

A red arrow points from the left margin to the `i++` expression in the for loop on line 11. A green box highlights the `i++` expression.

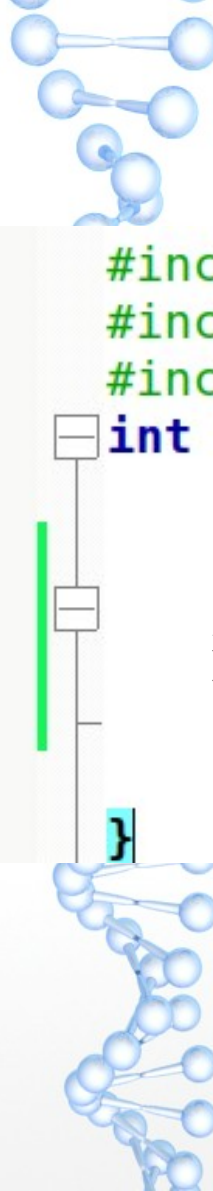
- `i = 1;`
- Após executar os comandos do laço, o passo do incremento ou decremento é executado.
 - `i++;`
- `i++` é igual a `i = i + 1;`
- Como o valor de `i` no momento é 1, ele passa a valer 2.
 - `i = 1 + 1;`



```
5  #include <stdio.h>
6  #include <stdlib.h>
7  #include <locale.h>
8  int main() {
9      setlocale(LC_ALL, "Portuguese");
10     int i;
11     for(i=0; i<10; i++){
12         printf(" %d \n", i);
13     }
14     return 0;
15 }
```

A red arrow points from the vertical line of the for loop to the condition `i < 10`, which is highlighted with a green box.

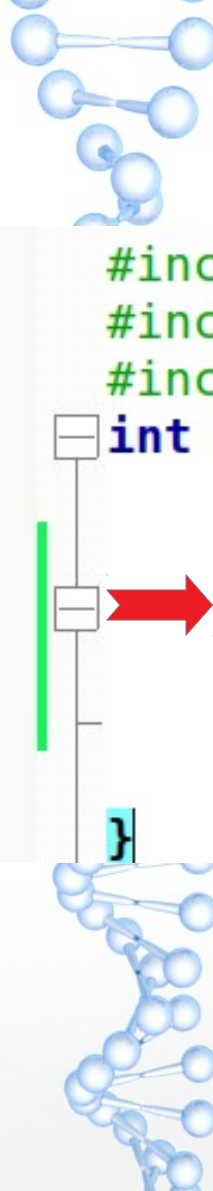
- `i = 2;`
- Verifica-se a condição proposta.
 - `i < 10` ?
- Se a condição proposta é verdadeira, executa-se os comandos dentro do laço. { }



```
5  #include <stdio.h>
6  #include <stdlib.h>
7  #include <locale.h>
8  int main() {
9      setlocale(LC_ALL, "Portuguese");
10     int i;
11     for(i=0; i<10; i++){
12         printf(" %d \n", i);
13     }
14     return 0;
15 }
```

A red arrow points to the `printf` statement on line 12, which is enclosed in a green box. A vertical green bar is positioned to the left of the code block, spanning from line 10 to line 13.

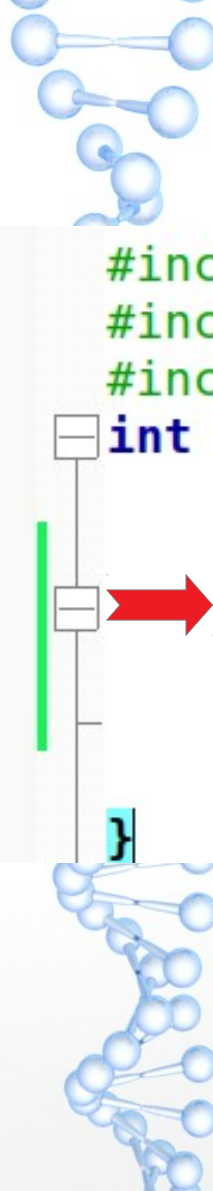
- `i = 2;`
- Verifica-se a condição proposta.
 - `i < 10` ?
- Se a condição proposta é verdadeira, executa-se os comandos dentro do laço. { }
- Escreva na tela (i) que no momento esta valendo 2.



```
5  #include <stdio.h>
6  #include <stdlib.h>
7  #include <locale.h>
8  int main() {
9      setlocale(LC_ALL, "Portuguese");
10     int i;
11     for(i=0; i<10; i++){
12         printf(" %d \n", i);
13     }
14     return 0;
15 }
```

A red arrow points to the `i++` expression in the for loop on line 11. A green box highlights the `i++` expression.

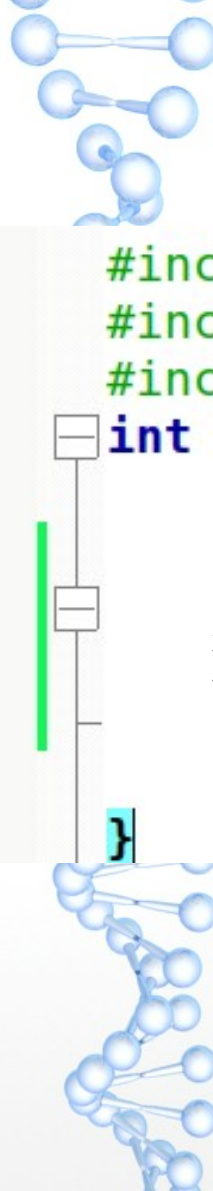
- `i = 2;`
- Após executar os comandos do laço, o passo do incremento ou decremento é executado.
 - `i++;`
- `i++` é igual a `i = i + 1;`
- Como o valor de `i` no momento é 2, ele passa a valer 3.
 - `i = 2 + 1;`



```
5  #include <stdio.h>
6  #include <stdlib.h>
7  #include <locale.h>
8  int main() {
9      setlocale(LC_ALL, "Portuguese");
10     int i;
11     for(i=0; i<10; i++){
12         printf(" %d \n", i);
13     }
14     return 0;
15 }
```

A red arrow points from the line number 11 to the condition `i < 10` in the for loop. A green vertical bar is positioned to the left of the code block, spanning from line 10 to line 13.

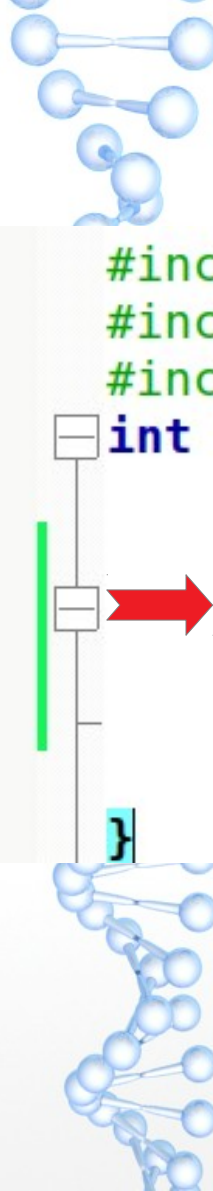
- `i = 3;`
- Verifica-se a condição proposta.
 - `i < 10` ?
- Se a condição proposta é verdadeira, executa-se os comandos dentro do laço. { }



```
5  #include <stdio.h>
6  #include <stdlib.h>
7  #include <locale.h>
8  int main() {
9      setlocale(LC_ALL, "Portuguese");
10     int i;
11     for(i=0; i<10; i++){
12         printf(" %d \n", i);
13     }
14     return 0;
15 }
```

A red arrow points to the `printf` statement on line 12, which is enclosed in a green box. A green vertical bar is positioned to the left of the code block.

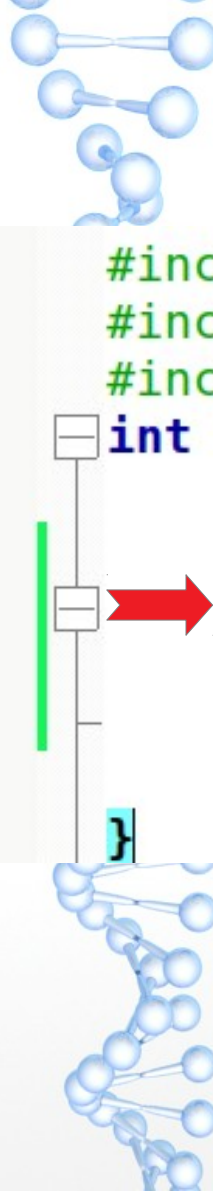
- **i = 3;**
- Verifica-se a condição proposta.
 - **i < 10** ?
- Se a condição proposta é verdadeira, executa-se os comandos dentro do laço. { }
- **Escreva na tela (i) que no momento esta valendo 3.**



```
5  #include <stdio.h>
6  #include <stdlib.h>
7  #include <locale.h>
8  int main() {
9      setlocale(LC_ALL, "Portuguese");
10     int i;
11     for(i=0; i<10; i++){
12         printf(" %d \n", i);
13     }
14     return 0;
15 }
```

A red arrow points to the `i++` expression in the for loop on line 11. A green box highlights the `i++` expression. A green vertical bar is on the left margin, spanning from line 10 to line 13.

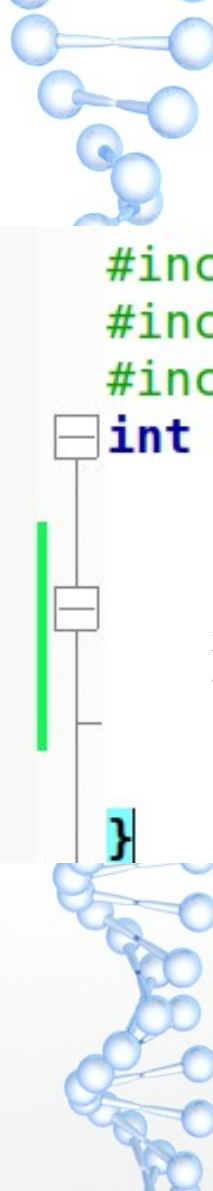
- `i = 3;`
- Após executar os comandos do laço, o passo do incremento ou decremento é executado.
 - `i++;`
- `i++` é igual a `i = i + 1;`
- Como o valor de `i` no momento é 3, ele passa a valer 4.
 - `i = 3 + 1;`



```
5  #include <stdio.h>
6  #include <stdlib.h>
7  #include <locale.h>
8  int main() {
9      setlocale(LC_ALL, "Portuguese");
10     int i;
11     for(i=0; i<10; i++){
12         printf(" %d \n", i);
13     }
14     return 0;
15 }
```

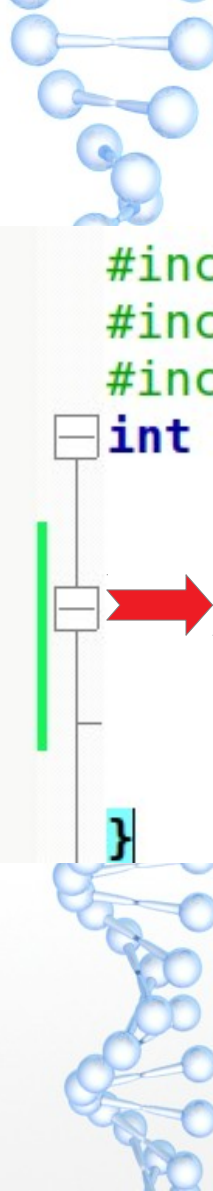
A red arrow points from the opening curly brace of the `main` function on line 8 to the `for` loop on line 11. A green box highlights the condition `i < 10` in the `for` loop.

- `i = 4;`
- Verifica-se a condição proposta.
 - `i < 10` ?
- Se a condição proposta é verdadeira, executa-se os comandos dentro do laço. { }



```
5  #include <stdio.h>
6  #include <stdlib.h>
7  #include <locale.h>
8  int main() {
9      setlocale(LC_ALL, "Portuguese");
10     int i;
11     for(i=0; i<10; i++){
12         printf(" %d \n", i);
13     }
14     return 0;
15 }
```

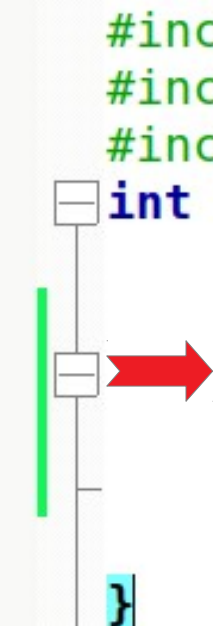
- **i = 4;**
- Verifica-se a condição proposta.
 - **i < 10** ?
- Se a condição proposta é verdadeira, executa-se os comandos dentro do laço. { }
 - **Escreva na tela (i) que no momento esta valendo 4.**



```
5  #include <stdio.h>
6  #include <stdlib.h>
7  #include <locale.h>
8  int main() {
9      setlocale(LC_ALL, "Portuguese");
10     int i;
11     for(i=0; i<10; i++){
12         printf(" %d \n", i);
13     }
14     return 0;
15 }
```

A red arrow points to the `i++` expression in the for loop on line 11. A green box highlights the `i++` expression. A green vertical bar is on the left margin next to lines 10 and 11.

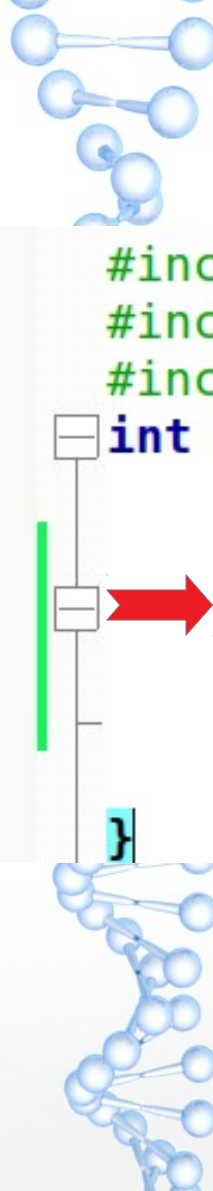
- `i = 4;`
- Após executar os comandos do laço, o passo do incremento ou decremento é executado.
 - `i++;`
- `i++` é igual a `i = i + 1;`
- Como o valor de `i` no momento é 4, ele passa a valer 5.
 - `i = 4 + 1;`



```
5  #include <stdio.h>
6  #include <stdlib.h>
7  #include <locale.h>
8  int main() {
9      setlocale(LC_ALL, "Portuguese");
10     int i;
11     for(i=0; i<10; i++){
12         printf(" %d \n", i);
13     }
14     return 0;
15 }
```

A red arrow points to the `i++` increment in the for loop on line 11. A green box highlights the `i++` expression. A green vertical bar is positioned to the left of the code, spanning from line 10 to line 13.

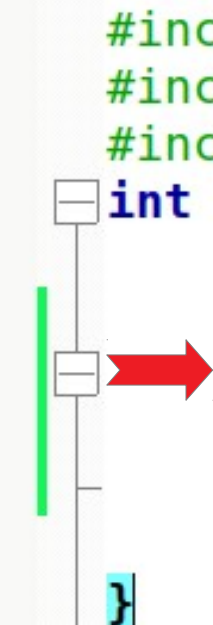
- Esse processo continua até que a condição proposta se torne falsa.
- Quando (i) ganhar o valor 10 o laço não será executado.



```
5  #include <stdio.h>
6  #include <stdlib.h>
7  #include <locale.h>
8  int main() {
9      setlocale(LC_ALL, "Portuguese");
10     int i;
11     for(i=0; i<10; i++){
12         printf(" %d \n", i);
13     }
14     return 0;
15 }
```

A red arrow points to the `i++` expression in the for loop on line 11. A green box highlights the `i++` expression. A green vertical bar is on the left margin next to lines 10 and 11.

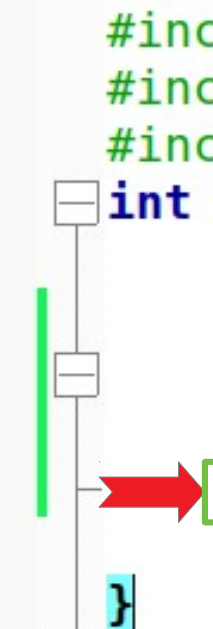
- `i = 9;`
- Após executar os comandos do laço, o passo do incremento ou decremento é executado.
 - `i++;`
- `i++` é igual a `i = i + 1;`
- Como o valor de `i` no momento é 9, ele passa a valer 10.
 - `i = 9 + 1;`



```
5  #include <stdio.h>
6  #include <stdlib.h>
7  #include <locale.h>
8  int main() {
9      setlocale(LC_ALL, "Portuguese");
10     int i;
11     for(i=0; i<10; i++){
12         printf(" %d \n", i);
13     }
14     return 0;
15 }
```

A red arrow points from the opening curly brace of the `main` function on line 8 to the `for` loop on line 11. A green box highlights the condition `i < 10` in the `for` loop.

- `i = 10;`
- Verifica-se a condição proposta.
 - `i < 10` ?
- Se a condição proposta é verdadeira, executa-se os comandos dentro do laço. { }

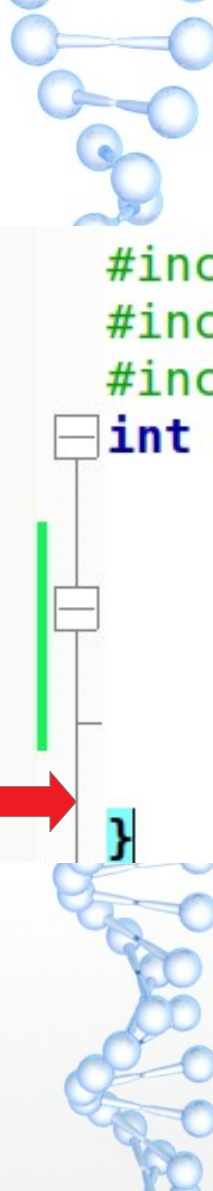


```
5  #include <stdio.h>
6  #include <stdlib.h>
7  #include <locale.h>
8  int main() {
9      setlocale(LC_ALL, "Portuguese");
10     int i;
11     for(i=0; i<10; i++){
12         printf(" %d \n", i);
13     }
14     return 0;
15 }
```


A vertical green bar is positioned to the left of the code, spanning from line 10 to line 13. A red arrow points from the closing curly brace of the for loop on line 13 to the right.

- **i = 10;**
- Verifica-se a condição proposta.
 - **i < 10** ?
- Se a condição proposta é verdadeira, executa-se os comandos dentro do laço. { }
- Como a condição é falsa, o laço é finalizado.

- O programa retorna 0 para o sistema operacional e é finalizado.



```
5  #include <stdio.h>
6  #include <stdlib.h>
7  #include <locale.h>
8  int main() {
9      setlocale(LC_ALL, "Portuguese");
10     int i;
11     for(i=0; i<10; i++){
12         printf(" %d \n", i);
13     }
14     return 0;
15 }
```



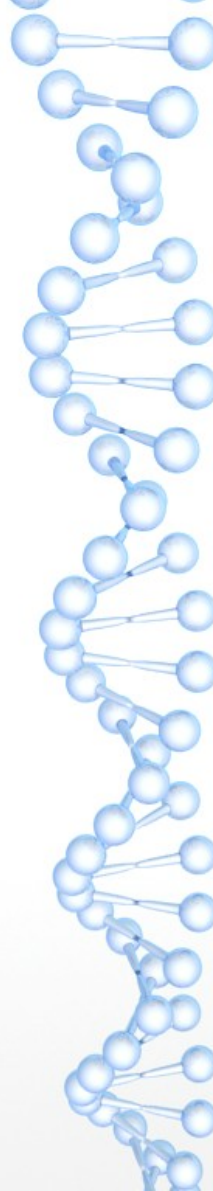


FOR

```
for ("VALOR-INICIAL"; "CONDIÇÃO"; "PASSO-DO-INCREMENTO") {  
    "CODIGOS..."  
}
```

- Os comandos podem ser ajustados para cada problema.
- Como exemplo, escreva um programa que apresente na tela a sequência numérica
 - 0 2 4 6 8 10 ... 100

FOR



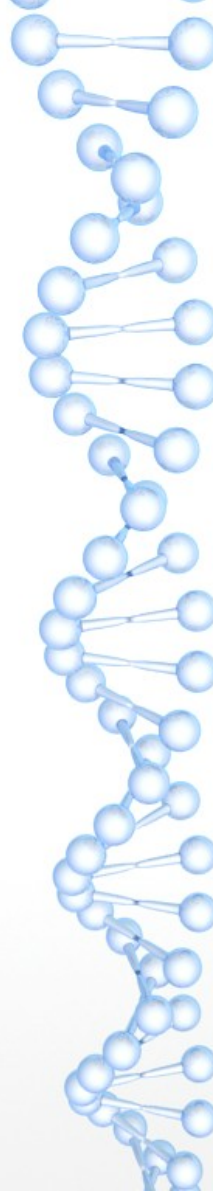
```
7  #include <stdio.h>
8  #include <stdlib.h>
9  #include <locale.h>
10 int main() {
11     setlocale(LC_ALL, "Portuguese");
12     int i;
13     for(i=0; i<=100; i=i+2){
14         printf(" %d \n", i);
15     }
16     return 0;
17 }
```



FOR

- Desenvolva um programa que apresente na tela os valores de 100 a 1.
 - Exemplo: 100 99 98 97 96 ... 1

FOR



```
7  #include <stdio.h>
8  #include <stdlib.h>
9  #include <locale.h>
10 int main() {
11     setlocale(LC_ALL, "Portuguese");
12     int i;
13     for(i=100; i>0; i--){
14         printf(" %d \n", i);
15     }
16     return 0;
17 }
```



FOR

- Desenvolva uma programa que apresente uma sequencia numérica de x a y. O usuário deve informar o valor inicial e o valor final da sequencia.
 - Exemplo:
 - Entrada → $x = 1$ e $y = 5$
 - Saída → 1 2 3 4 5

FOR

```
9  #include <stdio.h>
10 #include <stdlib.h>
11 #include <locale.h>
12 int main() {
13     setlocale(LC_ALL, "Portuguese");
14     int i, x, y;
15     printf("Digite o valor inicial : ");
16     scanf("%d", &x);
17     printf("Digite o valor final   : ");
18     scanf("%d", &y);
19     for(i=x; i<=y; i++){
20         printf(" %d \n", i);
21     }
22     return 0;
23 }
```

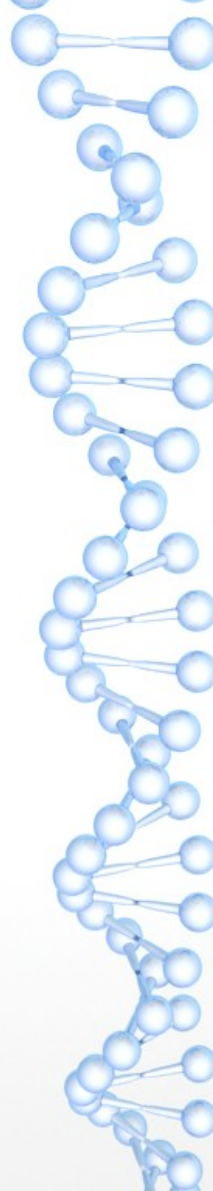
FOR

```
9  #include <stdio.h>
10 #include <stdlib.h>
11 #include <locale.h>
12 int main() {
13     setlocale(LC_ALL, "Portuguese");
14     int i, x, y;
15     printf("Digite o valor inicial : ");
16     scanf("%d", &x);
17     printf("Digite o valor final   : ");
18     scanf("%d", &y);
19     if (x < y){
20         for(i=x; i<=y; i++){
21             printf(" %d \n", i);
22         }
23     }else{
24         printf(" Valor inicial deve ser menor que valor final! \n");
25     }
26     return 0;
27 }
```



Exercícios

- Escreva um programa que apresente todos os números ímpares entre 0 e 1000.
- Escreva um programa que apresente todos os números múltiplos de 8 entre 0 e 1000.



```
6  #include <stdio.h>
7  #include <stdlib.h>
8  #include <locale.h>
9  int main() {
10     setlocale(LC_ALL, "Portuguese");
11     int i;
12     for(i=0; i<=1000; i++){
13         if(i%2 != 0){
14             printf(" %d - Ímpar \n", i);
15         }
16     }
17     return 0;
18 }
```

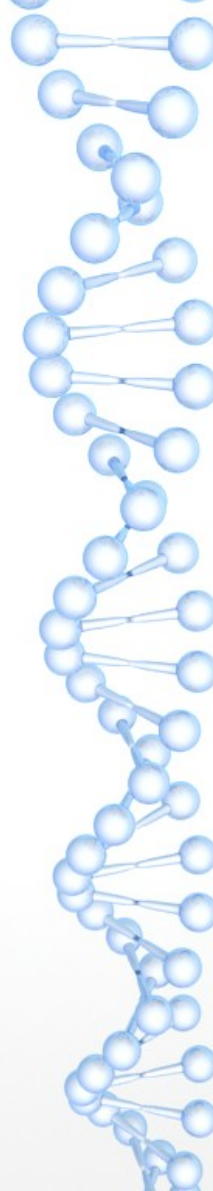



```
6  #include <stdio.h>
7  #include <stdlib.h>
8  #include <locale.h>
9  int main() {
10     setlocale(LC_ALL, "Portuguese");
11     int i;
12     for(i=0; i<=1000; i++){
13         if(i%8 == 0){
14             printf(" %d - Múltiplo de 8 \n", i);
15         }
16     }
17     return 0;
18 }
```



Exercícios

- Escreva um programa que leia 10 números e apresente o maior número.



```
6  #include <stdio.h>
7  #include <stdlib.h>
8  #include <locale.h>
9  int main() {
10     setlocale(LC_ALL, "Portuguese");
11     int i;
12     float numero, maior;
13     printf("Digite o 1 número : ");
14     scanf("%f", &numero);
15     maior = numero;
16     for(i=1; i<10; i++){
17         printf("Digite o %d número : ", i+1);
18         scanf("%f", &numero);
19         if (numero > maior){
20             maior = numero;
21         }
22     }
23     printf("Maior número = %.2f", maior);
24     return 0;
25 }
```



WHILE

- Vimos até o momento a estrutura de repetição FOR, que é utilizada quando conhecemos a quantidade de vezes necessária para “rodar” o laço.
- E se não soubermos a quantidade de vezes e sim a condição de parada?



WHILE

- O loop ou laço WHILE é usado para repetir um comando, ou bloco de comandos, n vezes enquanto a condição for verdadeira.
- Sua forma geral é

```
while( <CONDIÇÃO> ){  
    <COMANDOS>...  
}
```

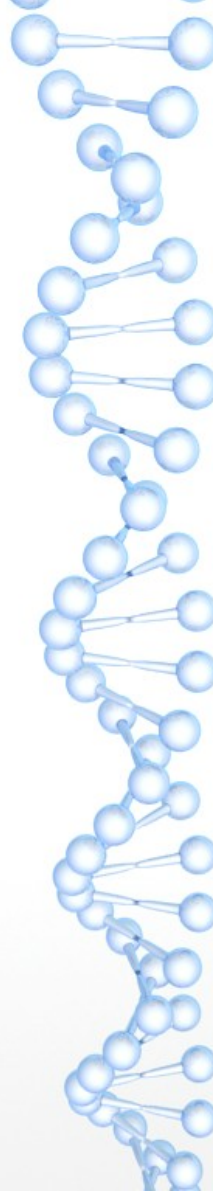


WHILE

```
while( <CONDIÇÃO> ){  
    <COMANDOS>...  
}
```

- CONDIÇÃO → Teste lógico realizado. Enquanto for verdade os comandos dentro do laço serão executados.

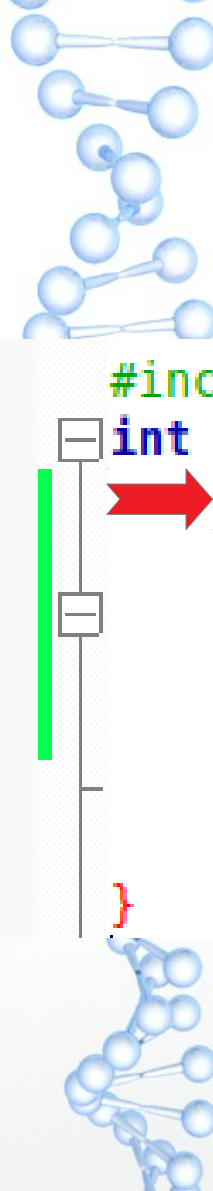
WHILE - Exemplo



```
6      #include <stdio.h>
7      int main() {
8          int i;
9          i = 1;
10         while(i<=10){
11             printf("  %d  ", i);
12             i++;
13         }
14         return 0;
15     }
```

A vertical green bar is positioned to the left of the code, spanning from line 7 to line 13. To the left of this bar, there are two square boxes: one at line 7 and another at line 10. A vertical line connects these two boxes, with a horizontal tick mark on the line at line 13.

WHILE - Exemplo




```
6  #include <stdio.h>
7  int main() {
8      int i;
9      i = 1;
10     while(i<=10){
11         printf(" %d ", i);
12         i++;
13     }
14     return 0;
15 }
```



- Declara uma variável do tipo inteira (int) com nome de (i).
- A memória RAM reserva um espaço suficiente para armazenar um número inteiro.
- O valor de (i) até o momento é desconhecido.

WHILE - Exemplo


- i recebe 1.
 - A variável (i) armazena o valor 1.



```
6  #include <stdio.h>
7  int main() {
8      int i;
9      i = 1;
10     while(i<=10){
11         printf(" %d ", i);
12         i++;
13     }
14     return 0;
15 }
```



WHILE - Exemplo



```
6  #include <stdio.h>
7  int main() {
8      int i;
9      i = 1;
10     while(i<=10){
11         printf(" %d ", i);
12         i++;
13     }
14     return 0;
15 }
```

A green vertical bar highlights the left margin of the code. A red arrow points from the line number 10 to the `while` loop. The condition `i<=10` is enclosed in a green box.

- **i = 1**
- Enquanto (i) for menor ou igual a 10 o laço será executado.
 - O laço (while) é executado enquanto a condicional de teste é verdadeira.
- **i<=10 ?**

WHILE - Exemplo

- **i = 1**
- Escreve na tela (1)

```
6  #include <stdio.h>
7  int main() {
8      int i;
9      i = 1;
10     while(i<=10){
11         printf(" %d ", i);
12         i++;
13     }
14     return 0;
15 }
```

WHILE - Exemplo

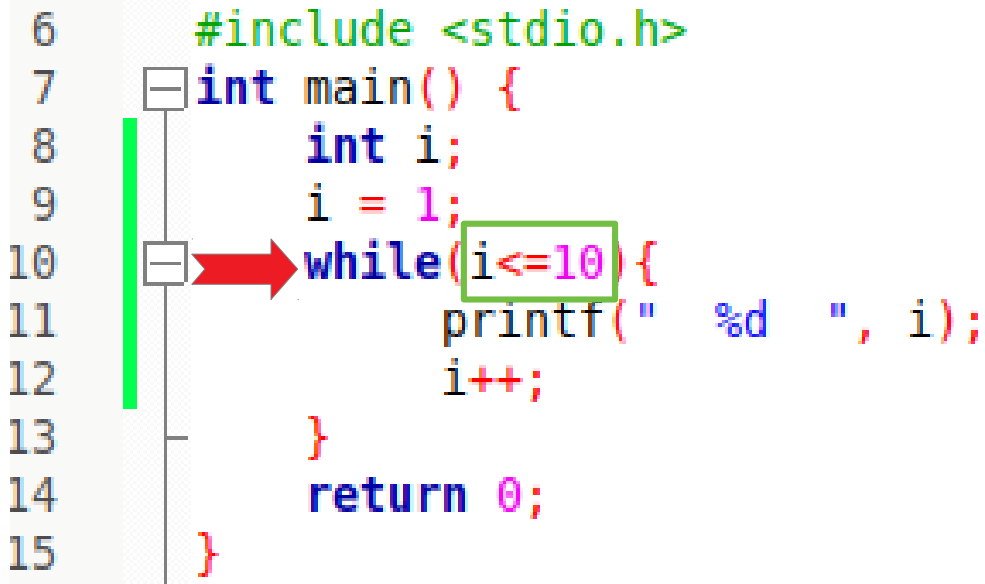
- **i = 1**
- **i = i + 1**
 - (i) recebe o valor de (i) atual, que é igual a 1, somado com 1.
 - **i = 1 + 1;**
 - **i = 2;**

```
6  #include <stdio.h>
7  int main() {
8      int i;
9      i = 1;
10 while(i<=10){
11     printf(" %d ", i);
12     ➔ i++;
13 }
14 return 0;
15 }
```


WHILE - Exemplo

- $i = 2$
- Enquanto (i) for menor ou igual a 10 o laço será executado.
 - $i \leq 10$?

```
6  #include <stdio.h>
7  int main() {
8      int i;
9      i = 1;
10     while(i <= 10) {
11         printf(" %d ", i);
12         i++;
13     }
14     return 0;
15 }
```




WHILE - Exemplo

- **i = 2**
- Escreve na tela (2)

```
6  #include <stdio.h>
7  int main() {
8      int i;
9      i = 1;
10     while(i<=10){
11         printf(" %d ", i);
12         i++;
13     }
14     return 0;
15 }
```

WHILE - Exemplo

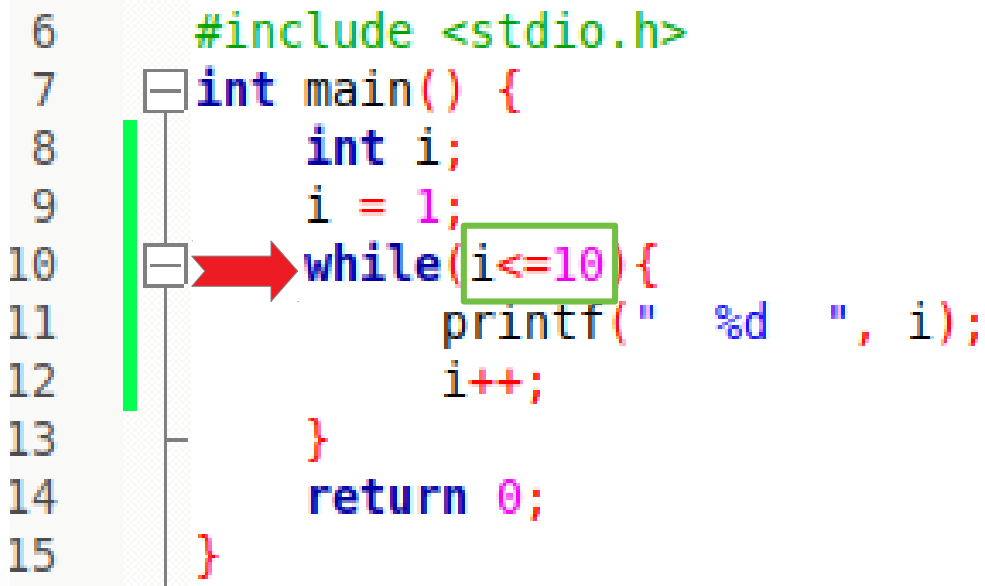
- **i = 2**
- **i = i + 1**
 - (i) recebe o valor de (i) atual, que é igual a 2, somado com 1.
 - **i = 2 + 1;**
 - **i = 3;**

```
6  #include <stdio.h>
7  int main() {
8      int i;
9      i = 1;
10     while(i<=10){
11         printf(" %d ", i);
12          i++;
13     }
14     return 0;
15 }
```

WHILE - Exemplo

- **i = 3**
- Enquanto (i) for menor ou igual a 10 o laço será executado.
 - **i <= 10 ?**

```
6  #include <stdio.h>
7  int main() {
8      int i;
9      i = 1;
10     while(i <= 10) {
11         printf(" %d ", i);
12         i++;
13     }
14     return 0;
15 }
```




WHILE - Exemplo

- **i = 3**
- Escreve na tela (3)

```
6  #include <stdio.h>
7  int main() {
8      int i;
9      i = 1;
10     while(i<=10){
11         printf(" %d ", i);
12         i++;
13     }
14     return 0;
15 }
```

WHILE - Exemplo





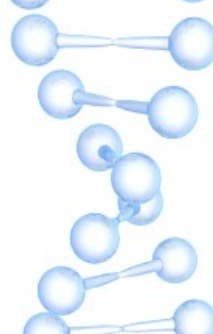
- **i = 3**
- **i = i + 1**
 - (i) recebe o valor de (i) atual, que é igual a 3, somado com 1.
 - $i = 3 + 1$;
 - $i = 4$;

```
6  #include <stdio.h>
7  int main() {
8      int i;
9      i = 1;
10 while(i<=10){
11     printf(" %d ", i);
12      i++;
13 }
14 return 0;
15 }
```


WHILE - Exemplo

- **i = 4**
- Enquanto (i) for menor ou igual a 10 o laço será executado.
 - **i <= 10 ?**

```
6  #include <stdio.h>
7  int main() {
8      int i;
9      i = 1;
10     while(i <= 10) {
11         printf(" %d ", i);
12         i++;
13     }
14     return 0;
15 }
```



WHILE - Exemplo

- **i = 4**
- Escreve na tela (4)

```
6  #include <stdio.h>
7  int main() {
8      int i;
9      i = 1;
10     while(i<=10){
11         printf(" %d ", i);
12         i++;
13     }
14     return 0;
15 }
```

WHILE - Exemplo

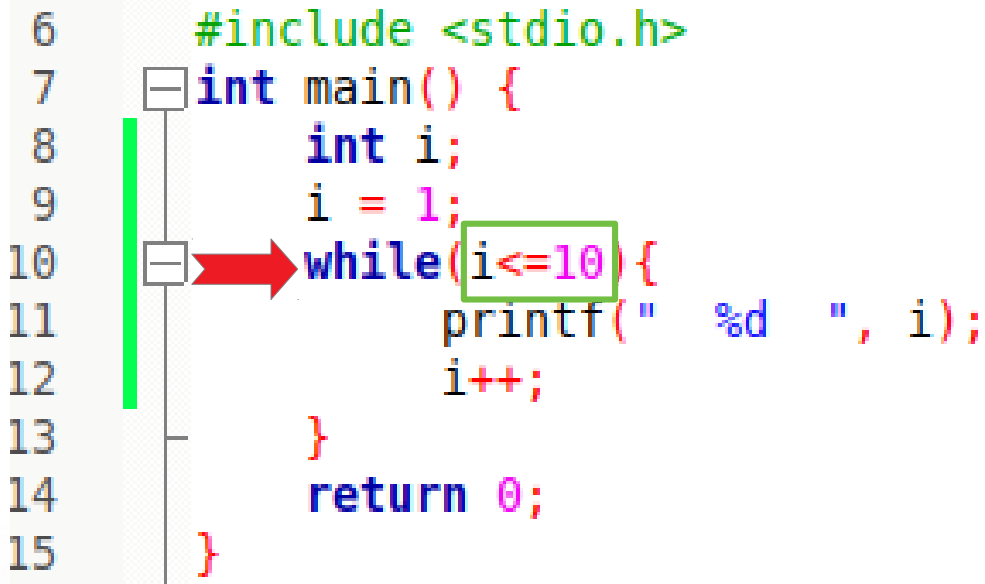
- **i = 4**
- **i = i + 1**
 - (i) recebe o valor de (i) atual, que é igual a 4, somado com 1.
 - **i = 4 + 1;**
 - **i = 5;**

```
6  #include <stdio.h>
7  int main() {
8      int i;
9      i = 1;
10     while(i<=10){
11         printf(" %d ", i);
12         ➔ i++;
13     }
14     return 0;
15 }
```

WHILE - Exemplo


- **i = 5**
- Enquanto (i) for menor ou igual a 10 o laço será executado.
 - **i <= 10 ?**

```
6  #include <stdio.h>
7  int main() {
8      int i;
9      i = 1;
10     while(i <= 10) {
11         printf(" %d ", i);
12         i++;
13     }
14     return 0;
15 }
```



WHILE - Exemplo

- Esse processo continua enquanto a condição proposta for verdadeira.
- Quando (i) ganhar o valor 11 o laço não será executado.

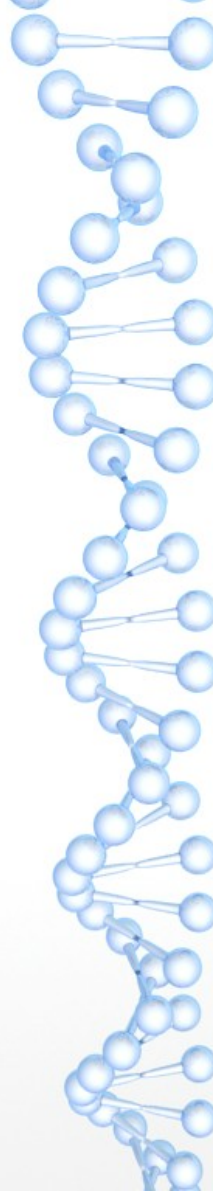


```
6  #include <stdio.h>
7  int main() {
8      int i;
9      i = 1;
10     while(i<=10){
11         printf("%d ", i);
12         i++;
13     }
14     return 0;
15 }
```

The code is presented in a monospaced font with syntax highlighting. A vertical green bar is on the left of the code lines. A red arrow points from the opening curly brace of the while loop on line 10 to the condition `i<=10`, which is enclosed in a green rectangular box. The code ends with a closing curly brace on line 15.

WHILE

- Percebemos então que o teste de validação do laço é realizado no início do laço.



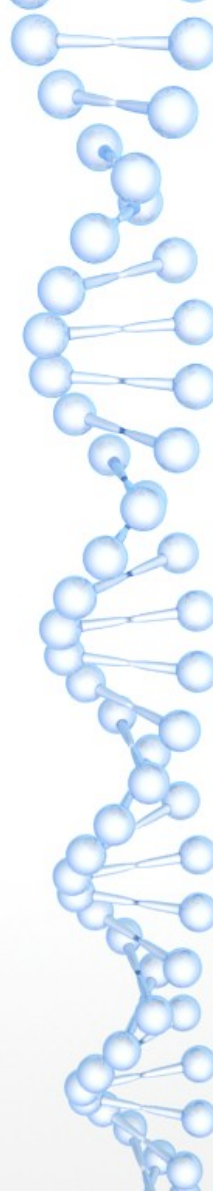
```
6      #include <stdio.h>
7      int main() {
8          int i;
9          i = 1;
10         while(i<=10){
11             printf("  %d  ", i);
12             i++;
13         }
14         return 0;
15     }
```

A diagram illustrating the execution of a C program. A vertical line on the left represents the program's flow. A green vertical bar highlights the loop condition `while(i<=10)`, which is enclosed in a green box. A red arrow points from the start of the loop body to the condition, indicating that the condition is checked before the first iteration.



WHILE

- Imagine o seguinte exemplo.
 - Construa um programa que receba indefinidos números inteiros e positivos e apresente o maior número.
 - O programa deve ser finalizado quando informado qualquer valor negativo.



```
7  #include <stdio.h>
8  #include <stdlib.h>
9  #include <locale.h>
10 int main() {
11     setlocale(LC_ALL, "Portuguese");
12     int numero, maior;
13     printf("Digite o número : ");
14     scanf("%d", &numero);
15     maior = numero;
16     while(numero >= 0){
17         printf("Digite o número : ");
18         scanf("%d", &numero);
19         if (numero > maior){
20             maior = numero;
21         }
22     }
23     printf("Maior número = %d", maior);
24     return 0;
25 }
```

DO-WHILE

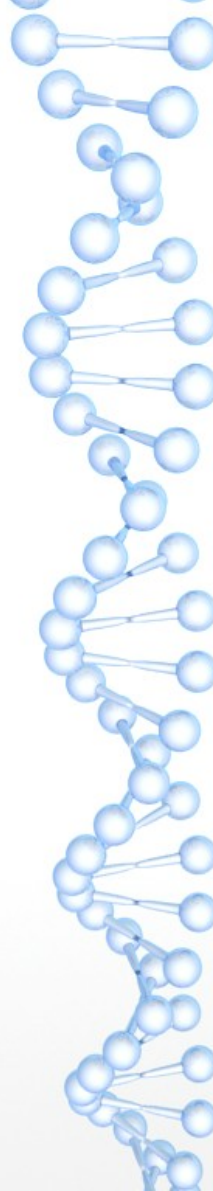
- Podemos realizar o teste no final do laço utilizando a estrutura DO-WHILE

```
7  [ ] int main() {  
8  [ ]     int i;  
9  [ ]     i = 1;  
10 [ ]     do{  
11 [ ]         printf("  %d  ", i);  
12 [ ]         i++;  
13 [ ]     }while(i<=10);  
14 [ ]     return 0;  
15 [ ] }
```



DO-WHILE

- Essa estrutura é utilizada quando não temos o valor inicial para “testar”, como no exemplo do exercício anterior.
 - Construa um programa que receba indefinidos números inteiros e positivos e apresente o maior número.
 - O programa deve ser finalizado quando informado qualquer valor negativo.
- A solução ideal para o problema proposto é utilizando o teste no final do laço, pois não sabemos o valor que será digitado pelo usuário.

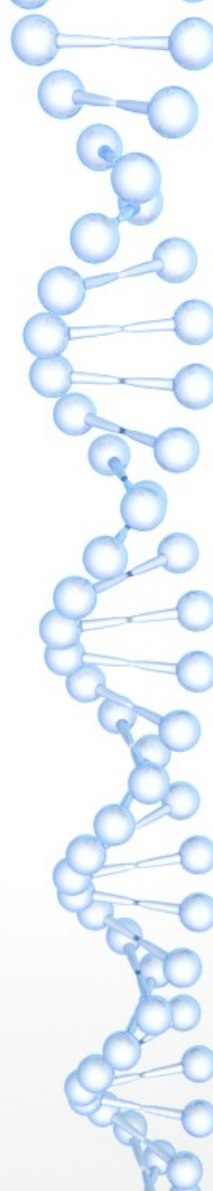


```
7  #include <stdio.h>
8  #include <locale.h>
9  int main() {
10     setlocale(LC_ALL, "Portuguese");
11     int numero, maior;
12     maior = -1;
13     do{
14         printf("Digite o número : ");
15         scanf("%d", &numero);
16         if (numero > maior){
17             maior = numero;
18         }
19     }while(numero>=0);
20     printf("Maior número = %d", maior);
21     return 0;
22 }
```



Exercício 1

- Faça um programa que verifique e mostre os números entre 1000 e 2000

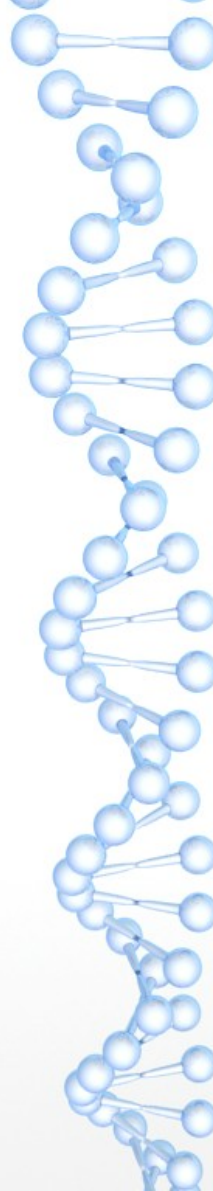


```
7      #include <stdio.h>
8      #include <locale.h>
9      - int main() {
10         setlocale(LC_ALL, "Portuguese");
11         int i;
12         - for(i=1000; i<=2000; i++){
13             - if(i % 11 == 5){
14                 printf(" %i \n", i);
15             }
16         }
17         return 0;
18     }
```




Exercício 2

- Desenvolva um programa que leia indefinidos números diferentes de zero, calcule e apresente na tela a média dos números lidos.



```
7  #include <stdio.h>
8  #include <locale.h>
9  int main() {
10     setlocale(LC_ALL, "Portuguese");
11     float numero, media;
12     int contador;
13     contador = 0;
14     do{
15         printf("Digite o número : ");
16         scanf("%f", &numero);
17         if (numero != 0){
18             media += numero;
19             contador++;
20         }
21     }while(numero!=0);
22     if (contador != 0){
23         printf("Média = %.2f", (media/contador));
24     }
25     return 0;
26 }
```



Exercícios

- Livro
 - Fundamentos da Programação de Computadores
 - Estrutura de Repetição
 - Exercícios Propostos

