

# INTRODUÇÃO À PROGRAMAÇÃO

Prof.<sup>a</sup> Priscilla Abreu

[priscilla.braz@rj.senac.br](mailto:priscilla.braz@rj.senac.br)



# Introdução à Programação



## Roteiro de Aula

- Objetivo da aula
- Estruturas de repetição

# Introdução à Programação



## Objetivo da aula

Compreender o funcionamento das estruturas de repetição.

# REVISANDO...

# Introdução à Programação



## ESTRUTURAS DE REPETIÇÃO

Tipos de estruturas de repetição:

- FOR
- WHILE
- DO ... WHILE

# Introdução à Programação



## ESTRUTURA FOR

Exemplo1: Imprimir os números de 1 a 10.

```
#include <stdio.h>
int main(){
    int i;
    for (i= 1; i<=10; i++) {
        printf("%d\n",i);
    }
}
```

# Introdução à Programação



## ESTRUTURA WHILE

Exemplo 1: Faça um programa que imprima os números de 1 a 10.

```
#include <stdio.h>
int main(){
    int cont =1;
    while (cont<=10){
        printf("%d ",cont);
        cont++;
    }
}
```

# Introdução à Programação



## ESTRUTURA WHILE

```
#include <stdio.h>
int main () {
    int num, quad;
    printf("Digite um número – 0 (zero) para sair.\n");
    scanf("%d", &num);
    while (num != 0){
        quad = num * num ;
        printf ("O quadrado de %d = %d\n", num, quad);
        printf("Digite um número – 0 (zero) para sair.\n");
        scanf("%d", &num);
    }
}
```



# DO...WHILE

# Introdução à Programação



## ESTRUTURA DO ... WHILE

Nessa estrutura, um bloco de comandos é realizado enquanto uma determinada condição for verdadeira.

No entanto, ela se difere da estrutura while, pois testa a condição ao final do corpo da estrutura.

Sintaxe:

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

# Introdução à Programação



## ESTRUTURA DO ... WHILE

Funcionamento da estrutura:

- O bloco de instruções é executado;
- A condição é avaliada;
- Se o resultado da condição for verdadeiro, volta-se a executar os comandos da estrutura;
- Se o resultado da condição for falso, o laço é terminado e o programa continua na instrução seguinte.

# Introdução à Programação



## ESTRUTURA DO ... WHILE

```
#include <stdio.h>
int main() {
    int i;
    i = 10;
    do {
        printf ("\n Número: %d", i);
        i++;
    } while(i <=10);
}
```

# Introdução à Programação



## ESTRUTURA DO ... WHILE

```
#include <stdio.h>
int main() {
    int num;
    do {
        printf ("\nInforme um número positivo:");
        scanf("%d", &num);
    } while(num <0);
    printf("Número positivo: %d",num);
}
```

# Introdução à Programação



## ESTRUTURA DO ... WHILE

```
#include <stdio.h>
int main() {
    int num, i, s=0;
    do {
        printf ("\nInforme um número:");
        scanf("%d", &num);
        s = s + num;
        i++;
    } while(i <=10);
    printf("Soma dos 10 números lidos: %d", s);
}
```

# Introdução à Programação



## ESTRUTURA DO ... WHILE

EXEMPLO: Faça um programa que apresente ao usuário o seguinte menu de opções:

\*\*\*\* Cálculo de áreas \*\*\*\*

- 1- Quadrado
- 2- Triângulo
- 3- Retângulo
- 4- Sair

O programa deve permitir que o usuário faça qualquer uma das operações quantas vezes ele quiser.

# Introdução à Programação



## ESTRUTURA DO ... WHILE

```
#include <stdio.h>
int main(){
    int op, l1, l2;
    float area;
    do{
        printf("\n**** Cálculo de áreas ****\n");
        printf("1- Quadrado\n");
        printf("2- Triângulo\n");
        printf("3- Retângulo\n");
        printf("4- Sair\n");
        printf("Escolha sua opção: ");
        scanf("%d",&op);
```



# Introdução à Programação



## ESTRUTURA DO ... WHILE

```
switch(op){  
    case 1:{  
        printf("Informe o lado do quadrado: ");  
        scanf("%d",&l1);  
        area=l1*l1;  
        printf("Área: %.1f\n",area);  
        break;  
    }  
}
```

## ESTRUTURA DO ... WHILE

```
case 2:{  
    printf("Informe a base do triângulo: ");  
    scanf("%d",&l1);  
    printf("Informe a altura do triângulo: ");  
    scanf("%d",&l2);  
    area=(l1*l2)/2;  
    printf("Área: %.1f\n",area);  
    break;  
}
```

# Introdução à Programação



## ESTRUTURA DO ... WHILE

```
case 3:{  
    printf("Informe a base do retângulo: ");  
    scanf("%d",&l1);  
    printf("Informe a altura do retângulo: ");  
    scanf("%d",&l2);  
    area=l1*l2;  
    printf("Área: %.1f\n",area);  
    break;  
}
```

# Introdução à Programação



## ESTRUTURA DO ... WHILE

```
        case 4:{
            printf("\nSaindo...\n");
            break;
        }
        default:{
            printf("\nOpção inválida!\n");
            break;
        }
    }
}while(op!=4);
}
```

# COMANDOS DE DESVIO

# Introdução à Programação



## COMANDOS DE DESVIO

### Break

O comando break pode ser tanto usado para terminar um teste case dentro de um comando switch quanto para interromper a execução de um laço.

Quando for encontrado no corpo do comando:  
término imediato;  
controle do programa muda o fluxo para o código imediatamente após o loop.

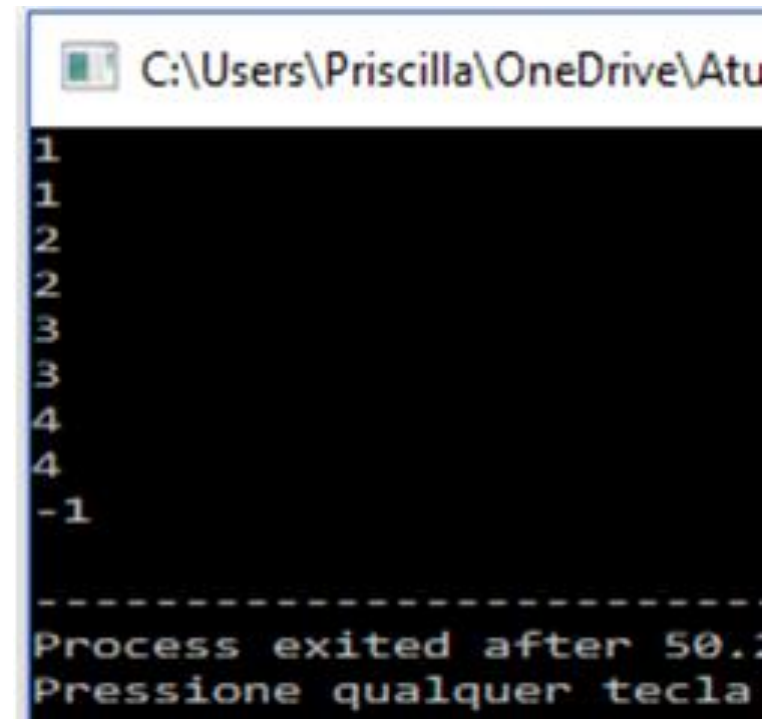
# Introdução à Programação



## COMANDOS DE DESVIO

### Break – exemplo:

```
for (i = 0; i < 10; i++){  
    scanf("%d", &num );  
    if (num < 0)  
        break;  
    printf ("%d\n", num );  
}
```

A screenshot of a Windows command prompt window. The title bar shows the path "C:\Users\Priscilla\OneDrive\Atu...". The command prompt has a black background with white text. It shows the output of a C program: a loop that reads integers until a negative number is entered. The input sequence shown is 1, 1, 2, 2, 3, 3, 4, 4, and then -1. After the -1, there is a dashed line and the text "Process exited after 50.3" and "Pressione qualquer tecla".

```
C:\Users\Priscilla\OneDrive\Atu...  
1  
1  
2  
2  
3  
3  
4  
4  
-1  
-----  
Process exited after 50.3  
Pressione qualquer tecla
```

# Introdução à Programação



## COMANDOS DE DESVIO

### Continue

O comando continue é parecido com o comando break.

A diferença é que o comando continue interrompe apenas a execução da iteração corrente, passando para a próxima iteração do laço, se houver uma.



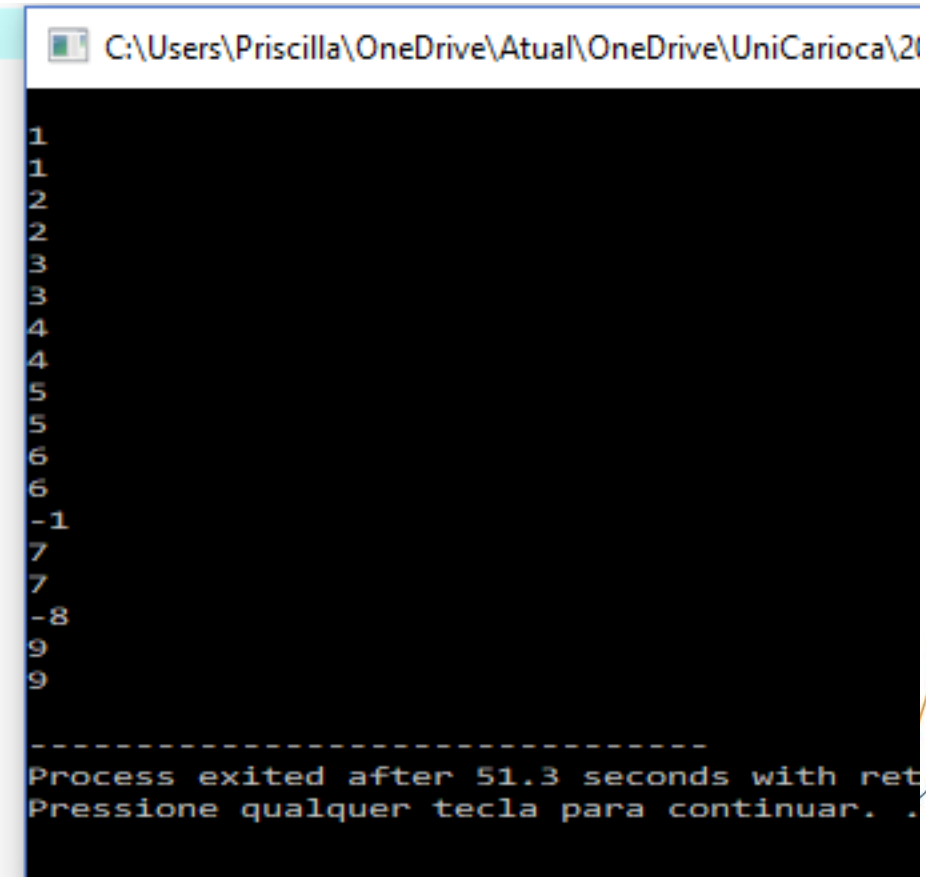
# Introdução à Programação



## COMANDOS DE DESVIO

### Continue – exemplo:

```
for (i = 0; i < 10; i++){  
    scanf("%d", &num );  
    if (num < 0)  
        continue ;  
    printf("%d\n", num );  
}
```



```
C:\Users\Priscilla\OneDrive\Atual\OneDrive\UniCarioca\20  
1  
1  
2  
2  
3  
3  
4  
4  
5  
5  
6  
6  
-1  
7  
7  
-8  
9  
9  
-----  
Process exited after 51.3 seconds with return code 0.  
Pressione qualquer tecla para continuar.
```

# Introdução à Programação



## COMANDOS DE DESVIO

### Exit()

Provoca a terminação de um programa, retornando o controle ao sistema operacional. O protótipo da função é a seguinte:

```
void exit (int codigo);
```

O código é usado para indicar qual condição causou a interrupção do programa. Usualmente utiliza-se 0.

# Introdução à Programação



## COMANDOS DE DESVIO

### Return

É usado para interromper a execução de uma função e retornar um valor ao programa que chamou esta função. Caso haja algum valor associado ao comando return, este é devolvido para a função, caso contrário um valor qualquer é retornado.

Formato geral:

```
return expressão;
```

# DÚVIDAS?