



CICLOS

if

Executa uma instrução ou bloco de instruções **se** uma determinada condição for verdadeira.

Ciclos

Executam uma instrução ou bloco de instruções **enquanto** uma determinada condição for verdadeira.

WHILE

while

```
while ( condição )  
    instrução;
```

avalia condição antes de
executar instrução

```
int counter = 5;  
int factorial = 1;  
while (counter > 1)  
{  
    factorial *= counter--;  
}  
printf("%d", factorial);
```

WHILE

while

```
while ( condição )  
    instrução;
```

avalia condição antes de executar instrução

```
int counter = 5;  
int factorial = 1;  
while (counter > 1)  
{  
    factorial *= counter--;  
}  
printf("%d", factorial);
```

do while

```
do  
    instrução  
while ( condição );
```

avalia condição depois de executar instrução


```
int counter = 5;  
int factorial = 1;  
do {  
    factorial *= counter--;  
} while (counter > 0);  
printf("%d", factorial);
```

```
printf("introduza um inteiro: ");  
scanf("%d", &b);  
switch(b){  
    case 1:  
        printf("Alan"); break;  
    case 2:  
        printf("Mathison"); break;  
    case 3:  
        printf("Turing"); break;  
    default:  
        printf("numero invalido");  
}
```

comportamento desejado

```
introduza um inteiro: 2  
Mathison  
introduza um inteiro: 1  
Alan  
introduza um inteiro: 50  
numero invalido  
introduza um inteiro: -1
```

programa termina depois de se
introduzir -1



Altere o programa para continuar a pedir um inteiro até que seja introduzido -1.

for

```
for (inicializações ; condição ; pós-instrução)  
    instrução;
```

tipicamente usado em situações em que o número de iterações é conhecido à partida;

```
int counter = 5;  
int factorial;  
for(factorial=1; counter > 1; counter--)  
    factorial *= counter;  
printf("%d", factorial);
```

counter=5

factorial=1

```
int counter = 5;
int factorial;
for (factorial=1; counter > 1; counter--)
    factorial *= counter;
printf("%d", factorial);
```



```
int counter = 5;
int factorial;
for (factorial=1; counter > 1; counter--)
    factorial *= counter;
printf("%d", factorial);
```

counter=5

factorial=1

counter > 1



```
int counter = 5;
int factorial;
for (factorial=1; counter > 1; counter--)
    factorial *= counter;
printf("%d", factorial);
```

counter=5

factorial=5

counter > 1

factorial *= counter;

factorial=1; counter > 1; counter--

factorial *= counter;



```
int counter = 5;
int factorial;
for (factorial=1; counter > 1; counter--)
    factorial *= counter;
printf("%d", factorial);
```

counter=4

factorial=5

counter > 1

factorial *= counter;

counter--



```
int counter = 5;
int factorial;
for (factorial=1; counter > 1; counter--)
    factorial *= counter;
printf("%d", factorial);
```

counter=4

factorial=5

counter > 1

factorial *= counter;

counter--

counter > 1

```
int counter = 5;  
int factorial;  
for (factorial=1; counter > 1; counter--)  
    factorial *= counter;  
printf("%d", factorial);
```

counter=3

factorial=20

counter > 1

factorial *= counter;

counter--

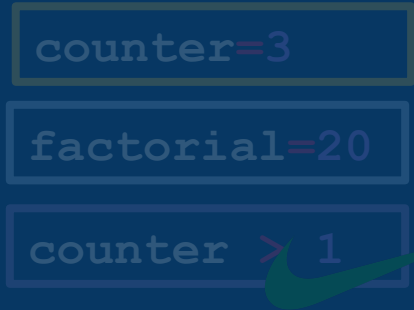
counter > 1

factorial *= counter;

counter--



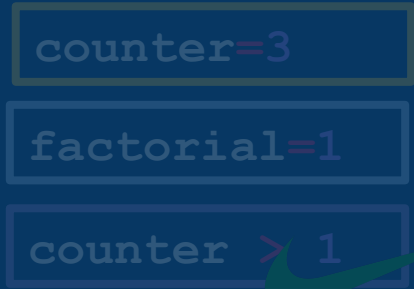
```
int counter = 5;
int factorial;
for (factorial=1; counter > 1; counter--)
    factorial *= counter;
printf("%d", factorial);
```



continua esta sequência até a condição ser falsa



```
int counter = 5;
int factorial;
for (factorial=1; counter > 1; counter--)
    factorial *= counter;
printf("%d", factorial);
```



```
factorial *= counter;
```

```
counter--
```



```
factorial *= counter;
```

```
counter--
```



continua esta sequência até a condição ser falsa



```
printf("%d", factorial);
```

FOR

Escreva um programa que escreva a tabuada de um número qualquer, usando um ciclo `for`.

Escreva um programa que escreva a tabuada de dos primeiros 5 números inteiros;

break

Terminar uma sequência de instruções dentro de um **switch** ou um **ciclo**;

continue

Avançar para a seguinte iteração;

```
int i;
printf("os numeros pares de 1 a 10 são:\n");
for(i=1; i<=10; i++){
    if (i % 2 == 0) printf("%d ,", i);
    else continue;
}
```

break

Terminar uma sequência de instruções dentro de um **switch** ou um **ciclo**;

continue

Avançar para a seguinte iteração;

```
int i;
printf("os numeros pares de 1 a 10 são:\n");
for(i=1; i<=10; i++){
    if (i % 2 == 0) printf("%d ,", i);
    else continue;
}
```

```
os numeros pares de 1 a 10 são:
2 ,4 ,6 ,8 ,10 ,
```

Ciclos infinitos

```
while (1)
    instrução;
```

```
do
    instrução;
while (1);
```

```
for ( ; ; )
    instrução;
```

A instrução `break` ou `return` terminam um ciclo infinito.

Ciclos infinitos

```
while (1)
    instrução;

do
    instrução;
while (1);

for ( ; ; )
    instrução;
```

**Quando é que um ciclo infinito
pode ser útil?**

A instrução `break` ou `return` terminam um ciclo infinito.