Lesson 6 For Loop

Among the so many features a basic computer can have, **looping** is a powerful one.

If you have got 100 papers to sign, it is impossible for the first and last signatures to be the same. As a human, you get tired and hence are unable to correctly repeat the same action. The computer, on the other hand, isn't the same. It has the ability to **repeat** a sequence of instructions precisely. This is usually done by the printer. If we want to print 200, 300, or even 1000 copies of a paper, all of the copies will be exactly the same.

How can we do so using programming?

Here's the **For loop**. If we want to execute a block of code a certain number of times.

For example, if we want to execute a target statement 10 times, we put it inside a **loop** that repeats itself 10 times.

```
for (initialization; condition; increment or decrement){
//the target statement(s)
}
```

How does it function?

1) **The initialization step**: is executed first, and only once. It must be followed by a semicolon.

2) **The condition**: is the second step to be executed. It determines when the loop terminates. In each iteration, the condition is evaluated. If it is true, the body of the loop is executed. However, if it's false, the loop terminates. Again, It must be followed by a semicolon.

3) **The target statement(s)**: the code block inside the curly braces { } to be executed.

4) **Increment or decrement**: In this step, the loop control variable is changed whether by incrementing or decrementing it.

For example:

```
#include <stdio.h>
int main() {
   int i;
   for ( i = 0; i < 5; i++) {
         printf("Gammal ");
```

```
        printf("Tech\n");
    }
}
```

output:

Gammal Tech

Gammal Tech

Gammal Tech

Gammal Tech

Gammal Tech

Try to code yourself:

- - > click here: [Lesson 6 for loop c1 - Replit]

The statements enclosed within the pair of curly braces **{ }** will be executed a certain number of times.

Example:

```
#include <stdio.h>
int main() {
    int i;
    //we declared a variable
    for (i = 0; i < 10; i++) {
            printf("%d ", i);
    }
}
```

output: 0 1 2 3 4 5 6 7 8 9

Try to code yourself:

- - > click here: [Lesson 6 for loop c2 - Replit]

In the first iteration:

i = 0;

- we assigned the variable i a value of zero

i<10;

- Is the value of i less than 10?

  yes

- The code block inside the curly braces is executed.
- **i++** (increment operator)

- The value of i will be increased by 1 : **i = 2**

In the second iteration:

i = 2;

- Is the value of i less than 10?

  yes

- The code block inside the curly braces is executed.

In the third iteration:

- The value of i will be increased by 1 : **i = 3**
- Is the value of i less than 10?

  yes

- The code block inside the curly braces is executed
- The same operation is repeated until **i = 11**
- Is the value of i less than 10?

  No

Thus, the **loop** terminates and the code block inside the braces **{ }** will not be executed again.