Lecture: 6

Nested For Loops

Introduction

A nested loop means a loop statement inside another loop statement.

In C, you can nest any type of loop (while, do-while, or for), but nesting for loops is the most common practice, especially when the number of iterations is known.

Basic Syntax

The structure consists of an Outer Loop and an Inner Loop.

```
for (initialization; condition; increment) {
    // Outer loop body
    for (initialization; condition; increment) {
        // Inner loop body (runs completely for every single outer iteration)
    }
}
```


تعديل من خلال WPS Office

How It Works (The Logic)

The execution follows a specific order:

The Outer Loop starts its first iteration.

The Inner Loop starts and runs from beginning to end (until its condition becomes false).

Once the inner loop finishes, control goes back to the outer loop for its second iteration.

The process repeats until the outer loop's condition becomes false.

Key Rule: If the outer loop runs n times and the inner loop runs m times, the code inside the inner loop will execute a total of n x m times.

تعديل من خلال WPS Office

Practical Example: Printing a Grid

Printing a rectangle of stars (3 rows and 5 columns).

```
#include <stdio.h>

int main() {
    int rows = 3;
    int cols = 5;
    for (int i = 1; i <= rows; i++) { // Outer loop for rows
        for (int j = 1; j <= cols; j++) { // Inner loop for columns
            printf("* ");
        }
        printf("\n"); // Move to next line after each row
    }
    return 0;
}
```


تعديل من خلال WPS Office

The out put :

```
* * * * *

* * * * *

* * * * *
```


تعديل من خلال WPS Office

Example 2:

```
#include <stdio.h>

int main() {

    int rows = 5;

    for (int i = rows; i >= 1; i--) { // Outer loop starts at 5 and decreases

        for (int j = 1; j <= i; j++) { // Inner loop runs 'i' times

            printf("* ");

        }

        printf("\n");

    }


    return 0;

}
```


تعديل من خلال WPS Office

* * * * *

* * * *

* * *

* *

*

تعديل من خلال WPS Office

EXAMPLE 3:

```
#include <stdio.h>
int main() {
for(int i = 1; i <= 4; i++) {
for(int j = 1; j <= 4; j++) {
printf("* ");
}
printf("\n");
}
return 0;
}
```


تعديل من خلال WPS Office

* * * *

* * * *

* * * *

* * * *

تعديل من خلال WPS Office

Example 4:

```
#include <stdio.h>
int main() {
for(int i = 1; i <= 5; i++) {
for(int j = 1; j <= i; j++) {
printf("%d ", j);
}
printf("\n");
}
return 0;
}
```


تعديل من خلال WPS Office

Output

1

2 1

3 2 1

4 3 2 1

5 4 3 2 1

تعديل من خلال WPS Office

while Loop

The while loop provides a mechanism for repeating C statements while a condition is true.

Its format is

While (control expression)

program statement;

The while statement works as follows:

1) Control expression is evaluated ("entry condition")

2) If it is FALSE,skip over the loop.

3) If it is TRUE,loop body is executed.

4) Go back to step 1

تعديل من خلال WPS Office

Example 1 :

The Standard Counter (Basic Logic)

This is the simplest form, used when you want to repeat an action until a specific limit is reached.

It requires three parts: Initialization, Condition, and Update.

```
#include <stdio.h>
int main() {
    int count = 1; // 1. Initialization
    while (count <= 5) { // 2. Condition
        printf("Iteration number: %d\n", count);

        count++; // 3. Update (Increment)
    }
    printf("Loop finished!\n");
    return 0;
}
```


تعديل من خلال WPS Office

Iteration number: 1

Iteration number: 2

Iteration number: 3

Iteration number: 4

Iteration number: 5

Loop finished!

تعديل من خلال WPS Office

Example 2:

Sentinel-Controlled Loop (User Input)

In professional C programming, while is often used when you don't know how many times the loop will run.

A Sentinel Value (like -1) is used to tell the loop when to stop.

تعديل من خلال WPS Office

```
#include <stdio.h>
int main() {
    int number = 0;
    int sum = 0;
    printf("Enter numbers to add (Enter -1 to stop):\n");
// سحب أول رقم قبل الدخول في الحلقة
    scanf("%d", &number);
    while (number != -1) {
        sum += number;
        printf("Current Sum: %d. Enter next number: ", sum);
// سحب الرقم التالي في نهاية الحلقة
        scanf("%d", &number);
    }
    printf("Final Sum: %d\n", sum);
    return 0;
}
```


تعديل من خلال WPS Office

Example 3:

```
#include <stdio.h>
int main() {
    int i = 2;
    while (i <= 10) {
        printf("%d\n", i);
        i += 2;
    }

    return 0;
}
```


تعديل من خلال WPS Office

Example 4:

```
#include <stdio.h>
int main() {
    int i = 1;
    int sum = 0;
    while (i <= 5) {
        sum += i;
        i++;
    }
    printf("Sum = %d\n", sum);
    return 0;
}
```


تعديل من خلال WPS Office

Example 5:

Read numbers until user enters 0

```
#include <stdio.h>

int main() {
    int num = 1;

    while (num != 0) {
        printf("Enter a number (0 to stop): ");
        scanf("%d", &num);
    }

    printf("Program ended\n");

    return 0;
}
```


تعديل من خلال WPS Office