Lecture – 11-

The **for** loop

- A **for** loop is a repetition control structure that allows you to efficiently write a loop that executes a specific number of times.

Syntax:

```
for ( init; condition; increment ) {
        statement(s);
}
```

- The **init** step is executed first and does not repeat.
- Next, the **condition** is evaluated, and the body of the loop is executed if the condition is true.
- In the next step, the **increment** statement updates the loop control variable.
- Then, the loop's body repeats itself, only stopping when the condition becomes **false**.
- **Example:**

```
for (int x = 1; x < 10; x++) {
        // some code
}
```

- The **init** and **increment** statements may be left out, if not needed, but remember that the **semicolons** are mandatory.
- The example below uses a **for** loop to print numbers from 0 to 9.

```
for (int a = 0; a < 10; a++) {
        cout << a << endl;
}

/* Outputs
0
1
2
3
4
5
```

```
6
7
8
9
*/
```

- In the **init** step, we declared a variable **a** and set it to equal 0. a < 10 is the **condition**.
- After each iteration, the a++ **increment** statement is executed.
- When **a** increments to 10, the condition evaluates to **false**, and the loop stops.
- It's possible to change the increment statement.

```
for (int a = 0; a < 50; a+=10) {
    cout << a << endl;
    }
/* Outputs
0
10
20
30
40
 */
```

- You can also use decrement in the statement.

```
for (int a = 10; a >= 0; a -= 3) {
    cout << a << endl;
    }

/* Outputs
10
7
4
1
*/
```

- When using the for loop, don't forget the **semicolon** after the **init** and **condition** statements.

The **do...while** Loop

- Unlike **for** and **while** loops, which test the loop condition at the top of the loop, the **do...while** loop checks its condition at the bottom of the loop.
- A **do...while** loop is like a **while** loop. The one difference is that the **do...while** loop is guaranteed to execute **at least one time**.

Syntax:

```
do {
    statement(s);
} while (condition);
```

- Example:

```
int a = 0;
do {
    cout << a << endl;
    a++;
} while(a < 5);

/* Outputs
    0
    1
    2
    3
    4
*/
```

- Don't forget the **semicolon** after the while statement.

while vs. do...while

- If the condition evaluated to **false**, the statements in the **do** would still run once:

```
int a = 42;
do {
    cout << a << endl;
    a++;
} while(a < 5);

// Outputs 42
```

- The **do...while** loop executes the statements at least once, and then tests the condition.
- The **while** loop executes the statement after testing condition.

The **do...while** Loop

- As with other loops, if the condition in the loop never evaluates to **false**, the loop will run forever.
- Example:

```
int a = 42;
do {
cout << a << endl;
} while (a > 0);
```

- This will print 42 to the screen **forever**.
- Always test your loops, so you know that they operate in the manner you expect.