# Al-Mustaqbal University
## College of Engineering Technology
Cybersecurity Techniques Engineering Department


# Programming Essential

## Lecture 6
Loops

PhD. BEng. Ahmed Hasan Janabi
PhD in Cybersecurity
Email: Ahmed.Janabi@uomus.edu.iq

# Objectives

## By the end of this lecture, students will be able to:

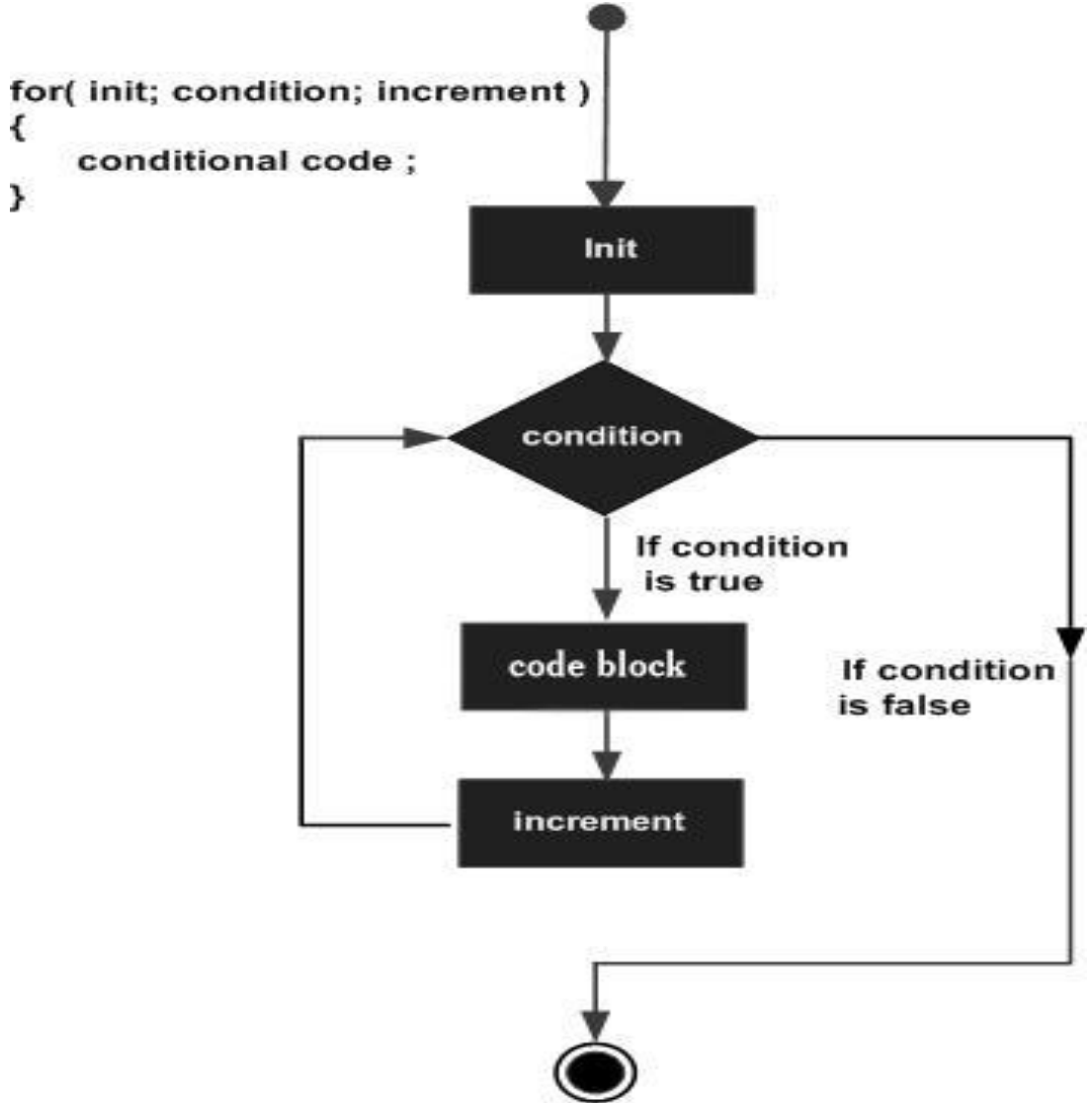
- Identify the purpose of **loops in C++ programming**
- Distinguish between **for, while, and do-while loops**
- Use loops to **repeat tasks automatically**
- Apply **break and continue statements** to control loop execution
- Implement **nested loops** for multi-level operations
- Design a simple **flowchart to represent loop behaviour**

# Think

- Should the system keep checking login attempts automatically?

# Part 1: The for Loop

- Used when we know **how many times** to repeat.



# Part 1: The for Loop - Example

❖ Example: Login Attempts Counter

```cpp
#include <iostream>
using namespace std;

int main() {

    for (int attempt = 1; attempt <= 3; attempt++) {
        cout << "Login attempt number: " << attempt << "\n";
    }

    return 0;
}
```

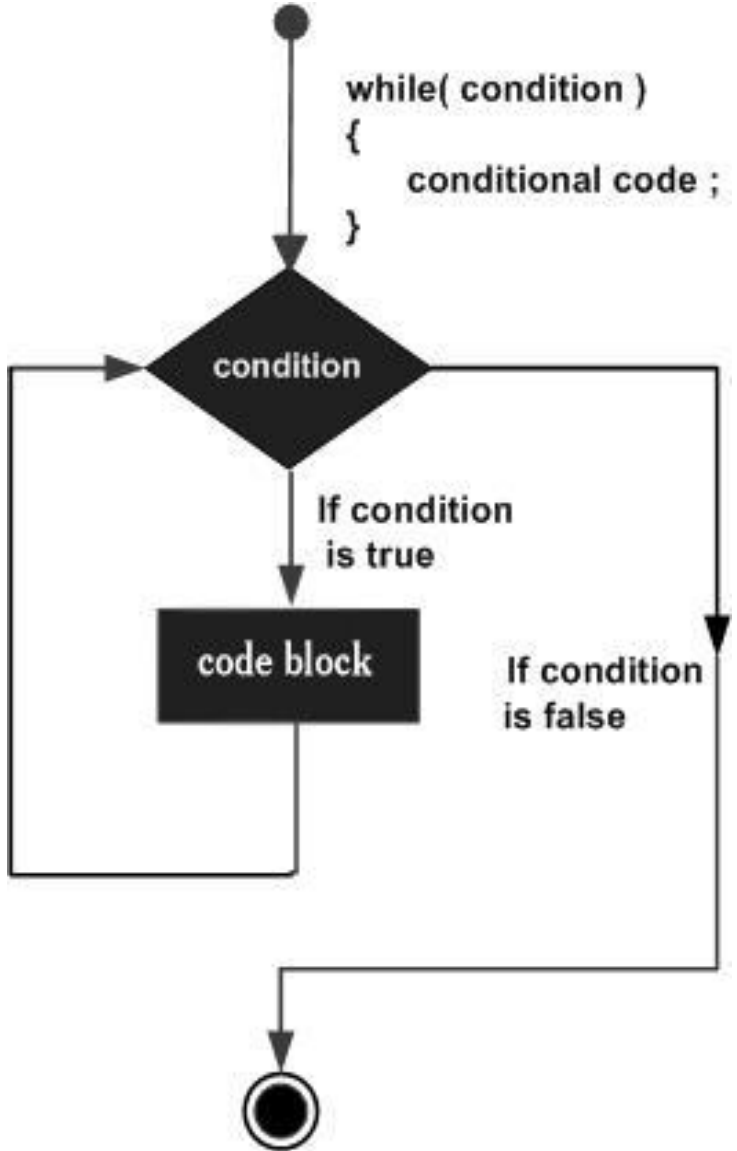
This is how a system allows only 3 login attempts.

**Output**

```
Login attempt number: 1
Login attempt number: 2
Login attempt number: 3
```

# Part 2: The while Loop

- Used when we repeat **until a condition becomes false**



# Part 2: The while Loop - Example

❖ Example: System Monitoring

```cpp
#include <iostream>
using namespace std;

int main() {

    int packets = 1;

    while (packets <= 5) {
        cout << "Monitoring packet: " << packets << "\n";
        packets++;
    }

    return 0;
}
```

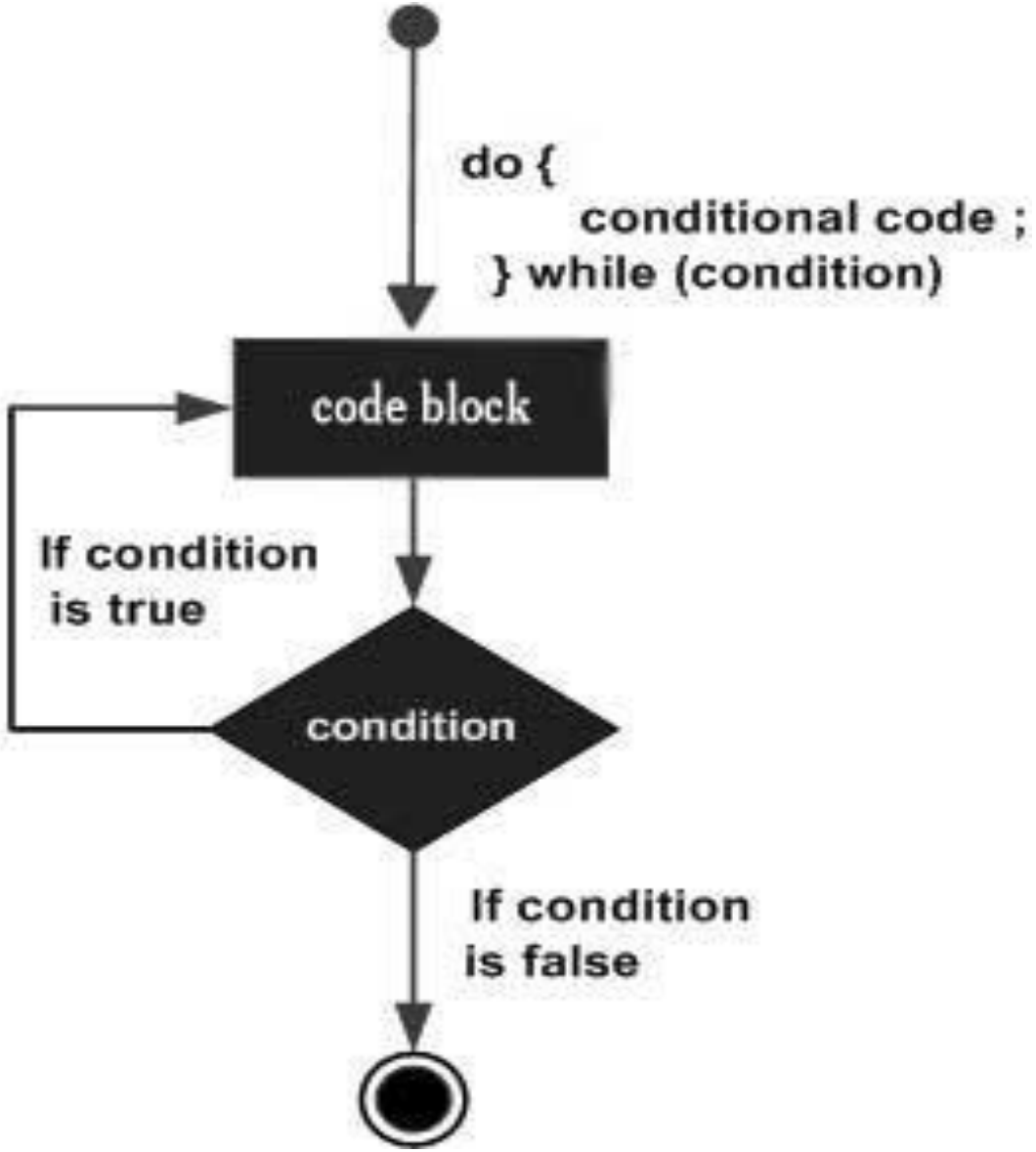
The system keeps running while packets are still being monitored.

**Output**

```
Monitoring packet: 1
Monitoring packet: 2
Monitoring packet: 3
Monitoring packet: 4
Monitoring packet: 5
```

# Part 3: The Do - while Loop

- **Always runs at least once**



# Part 3: The Do - while Loop - Example

❖ Example: Password Check

```cpp
#include <iostream>
using namespace std;

int main() {

    int tries = 1;

    do {
        cout << "Checking password attempt: " << tries << "\n";
        tries++;
    } while (tries <= 3);

    return 0;
}
```

Even if the password is correct, the system checks at least once.

**Output**

```
Checking password attempt: 1
Checking password attempt: 2
Checking password attempt: 3
```

# Part 4: break with Loops

- Used to **stop the loop immediately**
- Example: Stop When Suspicious Activity Detected

```cpp
#include <iostream>
using namespace std;

int main() {

    for (int user = 1; user <= 5; user++) {

        if (user == 3) {
            cout << "Suspicious activity detected – stopping monitoring\n";
            break;
        }

        cout << "Monitoring user: " << user << "\n";
    }

    return 0;
}
```

When the system finds something dangerous, it stops immediately.

```
Monitoring user: 1
Monitoring user: 2
Suspicious activity detected – stopping monitoring
```

# Part 5: continue with Loops

- **Used to skip one iteration**

❖Example: Ignore Blocked Users

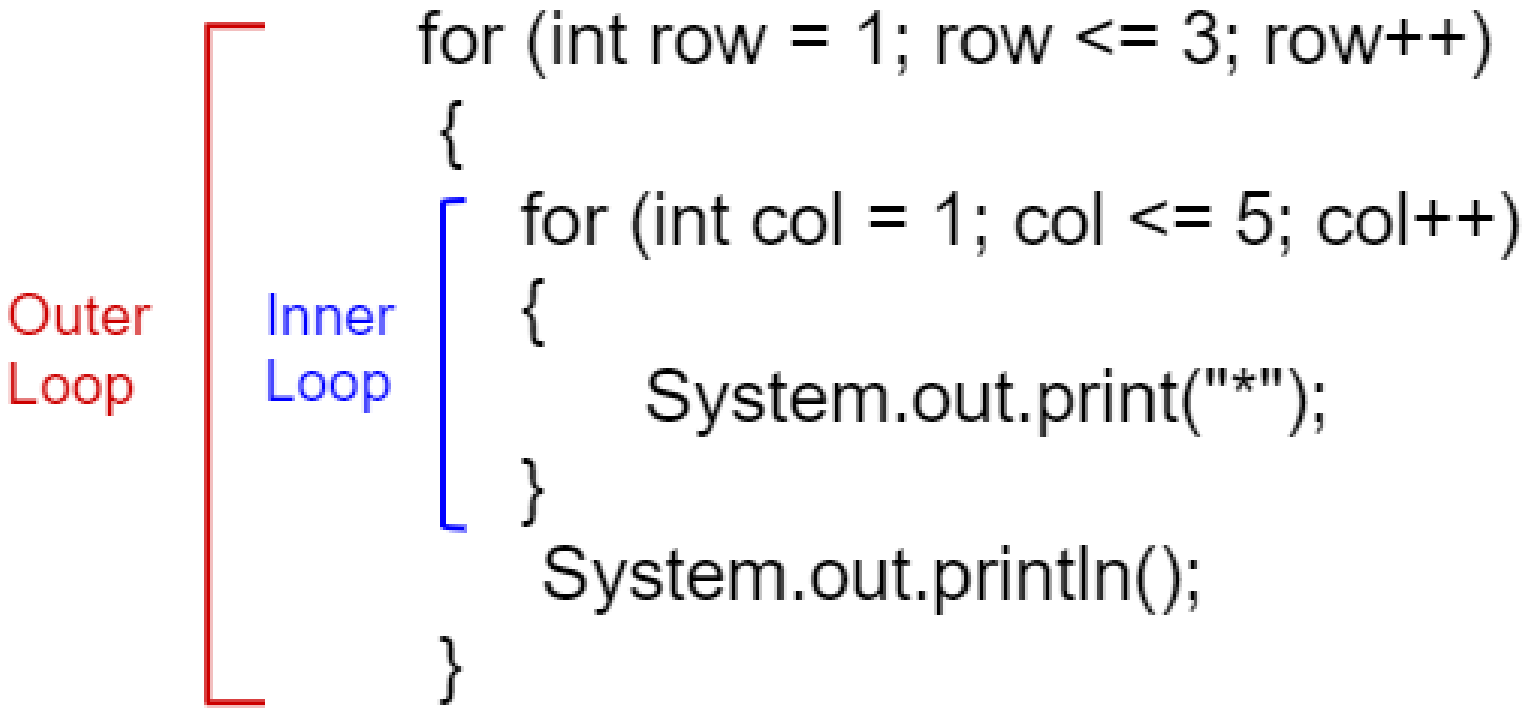
```cpp
#include <iostream>
using namespace std;

int main() {

    for (int user = 1; user <= 5; user++) {

        if (user == 2) {
            continue;
        }

        cout << "Scanning user: " << user << "\n";
    }

    return 0;
}
```

User 2 is blocked — the system skips scanning.

```
Scanning user: 1
Scanning user: 3
Scanning user: 4
Scanning user: 5
```

# Part 6: NESTED LOOP

- One loop runs **inside** another loop.
- **Some systems repeat tasks at more than one level.**



# Part 6: NESTED LOOP - Example

❖Example: Ignore Blocked Users

```cpp
#include <iostream>
using namespace std;

int main() {

    for (int user = 1; user <= 3; user++) {

        cout << "Monitoring User " << user << "\n";

        for (int attempt = 1; attempt <= 2; attempt++) {
            cout << "  Attempt " << attempt << "\n";
        }
    }

    return 0;
}
```

The outer loop handles users, the inner loop handles attempts.

```
Monitoring User 1
  Attempt 1
  Attempt 2
Monitoring User 2
  Attempt 1
  Attempt 2
Monitoring User 3
  Attempt 1
  Attempt 2
```

# Part 7: Exercise 1

**Write a program that:**

1. Uses a for loop from 1 to 5
2. Skips printing when the number = 3 using continue

# Part 7: Exercise 2

**Write a program that:**

1. Loops over **3 users**
2. For each user, prints **2 login attempts**

# Summary

| Loop | When to Use | Cyber Example |
|---|---|---|
| for | Known number of repetitions | Login attempts |
| while | Repeat while condition is true | Monitoring packets |
| do-while | Execute at least once | Password check |
| break | Stop immediately | Threat detected |
| continue | Skip current step | Blocked user |

# THANK YOU