Chapter 21: IF, FOR, WHILE

Introduction

In the world of Artificial Intelligence and programming, the ability to control the flow of instructions is essential. Just like humans make decisions and repeat tasks based on conditions, computer programs must also be able to do the same. This chapter introduces three important programming constructs used in decision-making and repetition:

- **IF** (Decision Making)
- **FOR** (Fixed Looping)
- **WHILE** (Conditional Looping)

These constructs are the foundation of logic in any AI or software system and are used to add intelligence to machines by helping them decide, repeat, and react based on inputs or data.

21.1 The IF Statement

What is the IF Statement?

The `if` statement is used for **decision-making**. It allows a program to **perform a specific action only when a certain condition is true**.

Syntax:

```
if condition:
    statement(s)
```

Example:

```
age = 18
if age >= 18:
    print("You are eligible to vote.")
```

IF-ELSE Statement:

If the condition is false, an alternative set of instructions can be executed using **else**.

```
if condition:
    # Executes if condition is true
else:
    # Executes if condition is false
```

Example:

```
age = 16
if age >= 18:
    print("You are eligible to vote.")
else:
    print("You are not eligible to vote.")
```

IF-ELIF-ELSE Ladder:

To check multiple conditions in a sequence, use `elif`.

```
marks = 85

if marks >= 90:
    print("Grade A")
elif marks >= 75:
    print("Grade B")
elif marks >= 50:
    print("Grade C")
else:
    print("Grade D")
```

21.2 The FOR Loop

What is the FOR Loop?

The **`for`** loop is used to **repeat a block of code a specific number of times**. It is typically used when the number of iterations is known.

Syntax:

```
for variable in range(start, stop, step):
    statement(s)
```

Explanation:

- `start`: Starting value (default is 0)
- `stop`: Loop runs till one less than this value
- `step`: Interval (default is 1)

Example:

```
for i in range(1, 6):
    print("Hello", i)
```

Output:

```
Hello 1
Hello 2
Hello 3
Hello 4
Hello 5
```

Using FOR with Lists:

```
colors = ['red', 'blue', 'green']
for color in colors:
    print(color)
```

21.3 The WHILE Loop

What is the WHILE Loop?

The **`while`** loop is used for **repeating a block of code as long as a condition remains true**. It is suitable when the number of repetitions is **not known beforehand**.

Syntax:

```
while condition:
    statement(s)
```

Example:

```
i = 1
while i <= 5:
    print("Count:", i)
    i += 1
```

Output:

```
Count: 1
Count: 2
Count: 3
Count: 4
Count: 5
```

21.4 Difference Between FOR and WHILE

Feature	FOR Loop	WHILE Loop
Use Case	Known number of iterations	Unknown number of iterations
Syntax	`for variable in range()`	`while condition:`
Control	Controlled by sequence	Controlled by condition
Example Use	Counting from 1 to 10	Wait until a password is correct

---

21.5 Nested IF and Loops

You can **combine IF statements inside loops** or **loops inside IF statements**.

Example – Nested IF inside FOR:

```
for i in range(1, 6):
    if i % 2 == 0:
        print(i, "is even")
    else:
        print(i, "is odd")
```

Summary

- **IF** is used for decision-making based on conditions.
- **IF-ELSE** and **IF-ELIF-ELSE** allow checking multiple conditions.
- **FOR** is used when the number of repetitions is known.
- **WHILE** is used when repetition continues until a condition becomes false.
- Loops can be nested or combined with conditions for complex logic.

These control structures form the **core logic** of intelligent systems and help in creating smarter, decision-making programs in AI and beyond.
