### **1. Introduction to Conditions**

In Python, **conditional statements** allow you to execute specific blocks of code based on whether a condition is **True** or **False**.

The primary conditional statements in Python are:

* if
* elif
* else

### **2. Syntax of Conditional Statements**

#### **if Statement**

The if statement is used to check if a condition is **True**. If it is, the code block under the if statement runs.

x = 10

if x > 5:

print("x is greater than 5")

Output:

x is greater than 5

If the condition is **False**, the code block under the if statement is skipped.

#### **elif (else if) Statement**

elif stands for "else if." It lets you check multiple conditions. If the first if condition is **False**, Python will check the conditions for elif.

x = 3

if x > 5:

print("x is greater than 5")

elif x == 3:

print("x is equal to 3")

Output:

x is equal to 3

#### **else Statement**

The else statement is used when all if and elif conditions are **False**. If no other condition is true, the code in the else block will run.

x = 2

if x > 5:

print("x is greater than 5")

elif x == 3:

print("x is equal to 3")

else:

print("x is less than or not equal to 3 or 5")

Output:

x is less than or not equal to 3 or 5

### **3. Multiple Conditions with Logical Operators**

You can combine conditions using logical operators like:

* and: returns **True** if both conditions are **True**.
* or: returns **True** if at least one of the conditions is **True**.
* not: reverses the Boolean value of the condition.

#### **Using and**

x = 10

y = 20

if x > 5 and y < 30:

print("Both conditions are true")

Output:

Both conditions are true

#### **Using or**

x = 10

y = 40

if x > 5 or y < 30:

print("At least one condition is true")

Output:

At least one condition is true

#### **Using not**

x = 10

if not x > 5:

print("This will not print because x > 5")

else:

print("x is greater than 5")

Output:

x is greater than 5

### **4. Nested Conditions**

You can also nest if statements inside one another. This allows more complex logic.

x = 10

y = 20

if x > 5:

if y < 30:

print("x is greater than 5 and y is less than 30")

else:

print("x is greater than 5 but y is not less than 30")

else:

print("x is not greater than 5")

Output:

x is greater than 5 and y is less than 30

### **5. The if Statement with Lists**

In Python, you can also use conditions to check elements in a list, string, or other collections.

#### **Check if an element is in a list**

fruits = ["apple", "banana", "cherry"]

if "banana" in fruits:

print("Banana is in the list!")

Output:

Banana is in the list!

#### **Check if an element is not in a list**

if "grape" not in fruits:

print("Grape is not in the list!")

Output:

Grape is not in the list!

### **6. Ternary Operator (Conditional Expression)**

In Python, you can also use a **ternary operator** (also known as a conditional expression) to assign values based on a condition.

x = 5

result = "Yes" if x > 0 else "No"

print(result)

Output:

Yes

The ternary operator follows this pattern:

value\_if\_true if condition else value\_if\_false

### **7. Summary of Conditionals in Python**

* **if**: Used to execute code if a condition is **True**.
* **elif**: Used to check another condition if the previous if or elif is **False**.
* **else**: Used for code that runs when all conditions are **False**.
* **Logical Operators** (and, or, not): Combine multiple conditions.
* **Ternary Operator**: A shorthand for if-else for simple conditions.