

# 1. What Conditional Statements Are

Conditional statements allow a program to **make decisions**.

A decision in a program means:

- The program checks a **condition**
- If the condition is **true**, one block of code runs
- If the condition is **false**, a different block may run or nothing happens

This is similar to real life:

- If it rains → take an umbrella
- If it does not rain → do not take one

In Python, decisions are made using:

- **if**
- **else**
- **elif**

## 2. Conditions and True / False

A **condition** is a question that can only have **two possible results**:

- True
- False

Examples of conditions:

- Is a number greater than another number?
- Is a number equal to zero?
- Is a remainder zero or not?

Python checks the condition and then decides what code to run.

## 3. The if Statement

The if statement checks a condition.

**General structure:**

```
python

if condition:
    code to run if condition is true
```

 Copy code

**Important rules:**

- The colon `:` ends the condition line
- The code that belongs to the `if` statement must be **indented**
- Only indented code runs when the condition is true
- If the condition is false, the indented code is skipped

## 4. Indentation in Python

Python uses **indentation** to group code.

- Indentation shows which code belongs to which decision
- Incorrect indentation causes errors
- All lines in the same block must use the **same indentation**

Recommended rule:

- Use **4 spaces** for each level of indentation

Example:

```
python

if x > 0:
    print("Positive")
print("Done")
```

 Copy code

Only the first print statement belongs to the `if`.

## 5. Using else

The `else` statement runs when the `if` condition is **false**.

Structure:

```
python

if condition:
    code if true
else:
    code if false
```

 Copy code

Rules:

- `else` must come after an `if`
- Only one of the two blocks will run
- If `if` runs, `else` is skipped

## 6. Even and Odd Number Example

To check if a number is even or odd, the **modulus operator** `%` is used.

- `%` gives the remainder of a division
- If the remainder is 0, the number is even
- Otherwise, the number is odd

Example:

```
python

R = X % 2

if R == 0:
    print("Even")
else:
    print("Odd")
```

 Copy code

Only one message is printed.

## 7. Why if-else Is Better Than Two if Statements

Using if-else is more efficient than writing two separate if statements.

With if-else:

- Once a condition is true, Python skips the other option
- This avoids unnecessary checks

## 8. Nested if Statements

A **nested if** is an if statement inside another if.

This is used when:

- One decision depends on another decision

Example idea:

- First check if a number is even
- Then check another condition inside that block

Indentation shows which if belongs to which block.

## 9. elif (Else If)

elif is used when there are **multiple possible conditions**, but only **one** should run.

### Structure:

```
python

if condition1:
    code
elif condition2:
    code
elif condition3:
    code
else:
    code
```

 Copy code

### Rules:

- Python checks conditions from top to bottom
- As soon as one condition is true, Python stops checking
- `else` runs if none of the conditions are true

### Example:

```
python

if X == 1:
    print("One")
elif X == 2:
    print("Two")
elif X == 3:
    print("Three")
else:
    print("Wrong input")
```

 Copy code

## 10. Key Terms

Term	Meaning
Conditional statement	Code that makes decisions
if	Runs code when a condition is true
else	Runs code when the if condition is false
elif	Checks another condition if previous ones are false
Indentation	Spaces used to define code blocks
Modulus %	Operator that returns the remainder of division
Nested if	An if inside another if