# Python – Day 3 Notes

## ✅ 1. Conditional Statements

Conditional statements allow programs to make **decisions** based on conditions.

### if Statement

Executes a block only if the condition is True.

```
age = 18
if age >= 18:
    print("Eligible to vote")
```

### if...else Statement

Executes one block if the condition is True, another if it's False.

```
age = 16
if age >= 18:
    print("Eligible to vote")
else:
    print("Not eligible")
```

### if...elif...else Statement

Checks multiple conditions.

```
marks = 85
if marks >= 90:
    print("Grade A")
elif marks >= 75:
    print("Grade B")
else:
    print("Grade C")
```

### Nested if Statements

if statements inside another if.

```
num = 10
if num > 0:
    if num % 2 == 0:
        print("Positive Even Number")
```

# 2. Looping Statements

Used to **repeat tasks** until a condition is met.

### for Loop

Used when you **know the number of iterations**.

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

range(start, stop) creates numbers from start to stop - 1

### while Loop

Used when the **condition is checked repeatedly**.

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

Be careful: If condition never becomes False, loop runs forever.

### Nested Loops

A loop inside another loop.

```
for i in range(1, 4):
   for j in range(1, 4):
      print(i, j)
```

Useful in patterns, matrix operations, etc.

# 3. Loop Control Statements

These control how loops behave.

### break Statement

**Stops the loop** immediately.

```
for i in range(10):
   if i == 5:
      break
   print(i)
```

### `continue` Statement

**Skips** the current loop iteration.

```
for i in range(5):
    if i == 2:
        continue
    print(i)
```

### `pass` Statement

Does **nothing** – a placeholder.

```
for i in range(3):
    pass  # Placeholder for future code
```

# BONUS TIPS

- Indentation is **super important** in Python (usually 4 spaces).
- Always use a **condition** in loops, or you'll create an **infinite loop**.
- `break` and `continue` can be used in both `for` and `while` loops.
- Use `pass` when building logic step-by-step – it avoids syntax errors.

# Real-World Examples

### Traffic Light System:

```
light = "green"
if light == "red":
    print("Stop")
elif light == "yellow":
    print("Get Ready")
else:
    print("Go")
```

### Game Countdown:

```
for i in range(10, 0, -1):
    print(i)
print("Game Start!")
```