Trisha

AI/ML Enthusiast | Intern @CodroidHub | First-Year Engineering Student

Artificial Intelligence/Machine Learning UpSkills Notebook

From Basics to Real-World — Starts Your ML Journey

Agenda — Control Flow Statements in Python

This notebook will help you learn and practice Python's basic control flow statements.
Here's what we'll cover:

Conditional Statements

- `if`, `elif`, `else` — how to make decisions

The `pass` Statement

- Use as a placeholder when you don't want to write code yet

Loops

- `while` loop — repeat code while a condition is True
- `for` loop — repeat code for each item in a sequence

Loop Control Statements

- `break` — exit a loop early
- `continue` — skip to the next loop iteration

The `range()` Function

- Generate number sequences for loops

Optional: `match-case`

- Pattern matching in Python 3.10+ (like switch-case)

Summary

- Recap what you learned and key points to remember

By the end of this notebook, you'll know how to control **when** and **how many times** your Python code runs — an essential skill for every programmer!

What are Control Flow Statements?

They are statements that **control the order** in which Python code runs.

The main control flow statements are:

- **Conditional Statements** → `if`, `elif`, `else`
- **Loops** → `while` and `for`
- **`break` and `continue`** → control how loops behave
- **`pass`** → do nothing placeholder
- **`range()`** → useful with `for` loops
- **Optional:** `match-case` (Python's version of `switch-case`, Python 3.10+)

Conditional Statements

- `if` : Runs code if a condition is True.
- `elif` : Checks another condition if `if` is False.
- `else` : Runs if no other condition is True.

Example:

In [1]:
```python
age = 18

if age >= 18:
    print("You are an adult.")
elif age >= 13:
    print("You are a teenager.")
else:
    print("You are a child.")
```

```
You are an adult.
```

"pass" Statement

- `pass` is a placeholder — it does **nothing**.
- Useful when you need a block of code syntactically but don't want to write it yet.

Example:

In [2]:
```python
x = 5

if x > 0:
    pass  # Do nothing for now

print("This runs anyway.")
```

```
This runs anyway.
```

"while" Loop

- Runs code **while a condition is True**.
- Be careful — you can create infinite loops if the condition never becomes False.

Example:

In [3]:
```python
count = 1

while count <= 3:
    print("Count:", count)
    count += 1
```

```
Count: 1
Count: 2
Count: 3
```

"for" Loop

- Repeats code **for each item in a sequence**.
- Commonly used with `range()` for a sequence of numbers.

Example:

In [4]:
```python
for i in range(1, 4):
    print("Number:", i)
```

```
Number: 1
Number: 2
Number: 3
```

"break" and "continue"

- `break` → Exit the loop early.
- `continue` → Skip the rest of the loop body and start the next iteration.

Example with `break`:

In [5]:
```
for num in range(1, 6):
    if num == 3:
        break
    print("Number:", num)
```

```
Number: 1
Number: 2
```

Example with `continue`:

In [6]:
```
for num in range(1, 6):
    if num == 3:
        continue
    print("Number:", num)
```

```
Number: 1
Number: 2
Number: 4
Number: 5
```

range() Function

- Generates a sequence of numbers.
- Commonly used with `for` loops.

Example:

In [16…
```
for i in range(5):
    print(i)    # Prints 0 to 4
```

```
0
1
2
3
4
```

In [11…
```
for i in range(2, 6):
    print(i)  # Prints 2 to 5
```

```
2
3
4
5
```

In [12…
```
for i in range(1, 10, 2):
    print(i)  # Prints 1, 3, 5, 7, 9
```

```
1
3
5
7
9
```

"match-case" Statement

- Similar to `switch-case` in other languages.
- Matches a variable's value to patterns.

Example:

In [1]:
```
command = input("Enter a command (start/stop): ")

match command:
    case "start":
        print("Starting...")
    case "stop":
        print("Stopping...")
    case _:
        print("Unknown command.")
```

```
Starting...
```

Summary: What You Learned

- `if`, `elif`, `else` → make decisions
- `while` and `for` → repeat tasks
- `break` and `continue` → control loop flow
- `pass` → do nothing for now
- `range()` → generate number sequences
- `match-case` → advanced pattern matching (Python 3.10+)

Keep practicing! Mastering control flow is key to writing smart Python programs.

Connect @
Mail (Trisha Dhiman): dhimantrisha1812@gmail.com
Contact: +91-9729832340

Trisha Dhiman

AI/ML Enthusiast | Intern @CodroidHub | First-Year Engineering Student

```
In [ ]:
```

Loading [MathJax]/jax/output/CommonHTML/fonts/TeX/fontdata.js