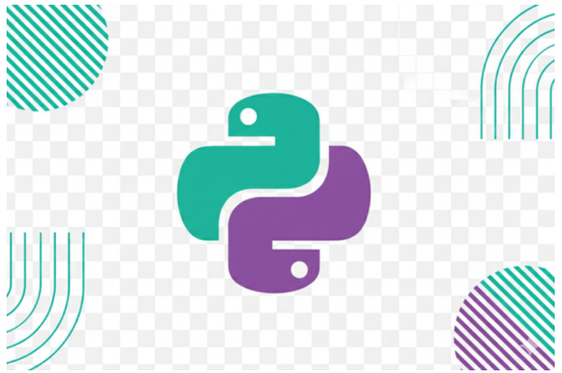
Python



Python Basics

OUTLINE

1 Overview of python

2 Variables and Data Types

3 Basic Data structures

4 Control Flow

5 Functions

6 simple project

7 Common Beginner Mistakes

8 Practice Exercises

What is python?

- A high-level, easy-to-read programming language
- Used for web development, data science, AI, and automation
- Simple syntax that looks like English
- Great for beginners and professionals alike

Why Learn Python?

- Easy to learn and read
- Versatile – used in many fields
- Strong community and resources
- High demand in job market

Getting Started: Your First Program

Lets write our first python program:

```
12
13    print("Hello World!")
```

- This prints text to screen.

Variables & Data Types

Variables store information. Python has key data types:

- Whole numbers: 42, -5
- Decimal numbers: 3.14, -2.5
- Text: "Hello", "Python"
- Boolean: True or False

Creating Variables

```
12
13    name= "Abebe"
14    age= 24
15    height=1.85
16    is_student= True
17
18    print(name)
19    print(age)
```

Getting User Input

Ask users for information using :

```
10
11
12
13  name = input("What is your name? ")
14  print("Hello, " + name)
15
```

- The program waits for the user to type something

Lists: Storing Multiple Items

Lists hold multiple values in order:

```
fruits = ["apple", "banana", "orange"]
numbers = [1, 2, 3, 4, 5]
print(fruits[0]) # Output: apple
```

- Lists start counting at 0(index)

Dictionaries: Key-Value Pairs

Dictionaries store data with labels(key):

```
student = {"name": "Bob", "age": 21, "grade": "A"}
print(student["name"]) # Output: Bob
```

- Access values using their keys, not positions

Tuples & Sets

Tuples: immutable list(can’t change)

```
colors = ("red", "green", "blue")
```

Sets: Unique items, unordered

```
unique_nums = {1, 2, 3, 2, 1}  # Output: {1, 2, 3}
```

Making Decisions: If Statements

Run code only when a condition is true:

```
age = 18
if age >= 18:
    print("You are an adult")
```

If-Else Statements

Execute different code for different conditions:

If-Elif-Else: Multiple Conditions

```python
score = 85
if score >= 90:
    print("A")
elif score >= 80:
    print("B")
else:
    print("C")
# This code evaluates a score and prints the corresponding grade.
```

Repeating Code: For Loops

Run the same code multiple times:

Loop Through Lists

```
fruits = ["apple", "banana", "orange"]
for fruit in fruits:
    print(fruit)
```

The loop repeats for each item in this list

While Loops: Loop Until Condition is False

```
count = 0
while count < 5:
    print(count)
    count = count + 1
```

Loop Control: Break & Continue

Break - Exit the loop early
Continue - Skip to the next iteration

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

Functions: Reusable Code Blocks

Functions let you reuse code without repeating it:

```
8
9
10
11
12  def greet():
13      print("Hello!")
14
15  greet() # Call the function
16
17
```

Functions with Parameters

```
def greet(name):
    print("Hello, " + name)

greet("Abebe")
```

Functions that Return Values

Functions can send data back:

```
def add(a, b):
    return a + b

result = add(3, 5)
print(result) # Output:
```

Functions with Multiple Parameters

```
10
11
12
13  def calculate_avg(a, b, c):
14      avg = (a + b + c) / 3
15      return avg
16
17  print(calculate_avg(10, 20, 30))
18
19
```

Project: Simple Calculator

Combine what you've learned:

- Get two numbers from the user
- Use if-elif to choose operation
- Create functions for each operation
- Return and print the result

Common Beginner Mistakes

- Forgetting colons after if, for, while, def
- Indentation errors – Python is sensitive!
- Using = instead of == for comparison
- Forgetting to convert input() to numbers

Practice Exercises

- Create a program that asks for your age and prints your birth year
- Write a program that checks if a number is even or odd
- Create a function that calculates factorial
- Build a program to find the maximum number in a list

THANK YOU