Python Basics

Python Overview: Python is a versatile, easy-to-learn programming language that stands out for its readability and simplicity. It's popular among developers for tasks ranging from web development to data analysis and machine learning, as well as scripting and automation.

Example Use Case: Imagine you want to automate your daily tasks like checking the weather and sending a reminder email. Python makes it simple with a few lines of code and existing libraries.

Installing Python

How to Install Python:

- **Windows:** Download from the official site, run the installer, and ensure you check "Add Python to PATH."
- **macOS:** Download the installer, run it, and follow the prompts.
- **Linux:** Use your package manager (e.g., sudo apt-get install python3 on Ubuntu).

Verification and Setup: Check the installation by typing python --version in your terminal. Create a virtual environment with python -m venv myenv to keep your projects isolated and organized.

Python Syntax and Semantics

Hello, World! Example:

print("Hello, World!")

This simple line of code demonstrates Python's straightforward syntax. The print function outputs text to the console.

Data Types and Variables

Key Data Types:

- **Integer (int)**: Whole numbers (e.g., 42).
- **Float (float)**: Numbers with decimals (e.g., 3.14).
- **String (str)**: Text (e.g., "hello").
- **Boolean (bool)**: True or False.
- **List**: An ordered collection (e.g., [1, 2, 3]).
- **Dictionary (dict)**: Key-value pairs (e.g., {"name": "Alice", "age": 25}).

Example Script:

name = "Alice"

age = 30

is_student = False

print(f"{name} is {age} years old. Is she a student? {is_student}")

Control Structures

Conditional Statements and Loops:

- **if Statements:** Control flow based on conditions.
- **Loops:** Repeat tasks, like iterating over a list.

Examples:

age = 20

if age < 18:

print("Minor")

else:

print("Adult")

python

for i in range(5):

print(i)

Functions in Python

What are Functions? Functions are reusable blocks of code that perform specific tasks. They help keep your code DRY (Don't Repeat Yourself).

Example Function:

def add(a, b):

return a + b

print(add(2, 3)) # Output: 5

Lists and Dictionaries

Lists vs. Dictionaries:

- **Lists:** Ordered collections accessible by index.
- **Dictionaries:** Collections of key-value pairs accessible by keys.

Example Script:

```
numbers = [1, 2, 3]

person = {"name": "Alice", "age": 30}

numbers.append(4)

print(numbers)  # Output: [1, 2, 3, 4]

print(person["name"])  # Output: Alice
```

Exception Handling

Handling Errors: Exception handling in Python allows you to manage errors gracefully using try, except, and finally blocks.

Example:

```
try:
    x = 1 / 0
except ZeroDivisionError:
    print("Cannot divide by zero!")
finally:
    print("This will always execute.")
```

Modules and Packages

Modules and Packages: Modules are Python files with reusable code. Packages are collections of modules. You can import them to use their functions.

Example:

```
import math

print(math.sqrt(16))  # Output: 4.0
```

File I/O

Reading and Writing Files: Python makes it easy to read from and write to files, enabling tasks like data storage and manipulation.

Example Scripts:

```
# Reading
with open('file.txt', 'r') as file:
    content = file.read()
    print(content)
# Writing
with open('output.txt', 'w') as file:
    file.write("Hello, World!")
```