PYTHON 101

Python Basics

```python
# Variables and Data Types
x = 10            # Integer
y = 3.14          # Float
name = "Alice"    # String
is_active = True  # Boolean

# Basic Operations
z = x + y  # Addition
print(z)   # Output: 13.14

# Input and Output
name = input("Enter your name: ")
print(f"Hello, {name}!")
```

Control Flow

```python
# If-Else Statement
age = 18
if age >= 18:
    print("Adult")
else:
    print("Not an Adult")

# For Loop
for i in range(5):
    print(i)  # Output: 0, 1, 2, 3, 4

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

Functions

```python
# Defining and Calling Functions
def greet(name):
    return f"Hello, {name}!"
print(greet("Alice"))  # Output: Hello, Alice!

# Lambda Functions
square = lambda x: x ** 2
print(square(5))  # Output: 25
```

Data Structures

```python
# List
fruits = ["apple", "banana", "cherry"]
fruits.append("orange")
print(fruits)  # Output: ['apple', 'banana', 'cherry', 'orange']

# Dictionary
person = {"name": "John", "age": 30}
print(person["name"])  # Output: John

# Set
numbers = {1, 2, 3, 2}
print(numbers)  # Output: {1, 2, 3}

# Tuple
point = (10, 20)
print(point[0])  # Output: 10

# Collection Module
from collections import namedtuple, deque, Counter

Point = namedtuple('Point', ['x', 'y'])
p = Point(1, 2)
print(p.x, p.y)  # Output: 1 2

d = deque([1, 2, 3])
d.appendleft(0)
print(d)  # Output: deque([0, 1, 2, 3])

counter = Counter(['a', 'b', 'a'])
print(counter)  # Output: Counter({'a': 2, 'b': 1})

# Iterators and Generators
def my_generator():
    yield 1
    yield 2
    yield 3

for value in my_generator():
    print(value)  # Output: 1, 2, 3
```

Strings and Regular Expressions

```python
# String Methods
s = "hello"
print(s.upper())  # Output: hello

# String Formatting
age = 25
print(f"I am {age} years old.")  # Output: I am 25 years old.

# Regular Expressions
import re

pattern = r"\d+"
text = "There are 3 apples"
match = re.search(pattern, text)
print(match.group())  # Output: 3
```

File Handling

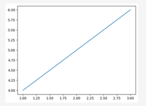
```python
# Reading Files
with open("file.txt", "r") as file:
    content = file.read()
    print(content)

# Writing to Files
with open("output.txt", "w") as file:
    file.write("Hello, World!")

# Exception Handling
try:
    with open("non_existent.txt", "r") as file:
        content = file.read()
except FileNotFoundError:
    print("File not found!")
```

Object-Oriented Programming (OOP)

```python
# Defining Classes
class Dog:
    def __init__(self, name):
        self.name = name

    def bark(self):
        return f"{self.name} is barking!"

dog = Dog("Buddy")
print(dog.bark())  # Output: Buddy is barking!

# Inheritance
class Labrador(Dog):
    def bark(self):
        return f"{self.name} is barking! I am a Labrador."

lab = Labrador("Max")
print(lab.bark())  # Output: Max is barking! I am a Labrador.
```

Working with Data

```python
# NumPy Arrays
import numpy as np

arr = np.array([1, 2, 3, 4])
print(arr.mean())  # Output: 2.5

# Pandas DataFrame
import pandas as pd

data = {"Name": ["John", "Alice"], "Age": [28, 24]}
df = pd.DataFrame(data)
print(df.head())  # Display the first few rows of the DataFrame

# Plotting with Matplotlib
import matplotlib.pyplot as plt

plt.plot([1, 2, 3], [4, 5, 6])
plt.show()
```



Web Development

```python
# Simple Web App with Flask
from flask import Flask

app = Flask(__name__)

@app.route("/")
def home():
    return "Hello, Flask!"

if __name__ == "__main__":
    app.run(debug=True)


# Handling HTTP Requests
@app.route("/submit", methods=["POST"])
def submit():
    data = request.form["data"]
    return f"Data received: {data}"
```

Testing

```python
# Testing with unittest
import unittest

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

class TestMath(unittest.TestCase):
    def test_add(self):
        self.assertEqual(add(1, 2), 3)

if __name__ == "__main__":
    unittest.main()

# Testing with pytest
def test_add():
    assert add(1, 2) == 3
```

V1