**Python**

Python is a powerful, high-level, and easy-to-learn programming language used for web development, data science, automation, and more. Let’s dive into its **fundamental concepts** with examples.

**1. Installing Python**

Download and install Python from [Python’s official website](https://www.python.org/downloads/).  
To check if Python is installed, open a terminal or command prompt and run:

python --version

**2. Python Syntax and Indentation**

Python uses indentation (spaces or tabs) to define blocks of code instead of braces {}.

**Example:**

if True:

print("Python uses indentation!")

(Incorrect indentation will cause an IndentationError.)

**3. Variables and Data Types**

Python variables **do not require explicit declaration**. The type is inferred automatically.

# Integer

x = 10

print(type(x)) # Output: <class 'int'>

# Float

y = 20.5

print(type(y)) # Output: <class 'float'>

# String

name = "Python"

print(type(name)) # Output: <class 'str'>

# Boolean

is\_python\_easy = True

print(type(is\_python\_easy)) # Output: <class 'bool'>

**4. Taking User Input**

name = input("Enter your name: ")

print("Hello, " + name + "!")

(By default, input() reads as a string.)

**5. Type Conversion (Casting)**

a = int("10") # Converts string "10" to integer

b = float("10.5") # Converts string "10.5" to float

c = str(100) # Converts integer 100 to string

**6. Operators in Python**

Python supports arithmetic, comparison, logical, and bitwise operators.

**Arithmetic Operators:**

a, b = 5, 2

print(a + b) # Addition: 7

print(a - b) # Subtraction: 3

print(a \* b) # Multiplication: 10

print(a / b) # Division: 2.5

print(a // b) # Floor Division: 2

print(a % b) # Modulus: 1

print(a \*\* b) # Exponentiation: 25

**7. Conditional Statements**

Python uses if, elif, and else for decision-making.

age = int(input("Enter your age: "))

if age >= 18:

print("You are an adult.")

elif age >= 13:

print("You are a teenager.")

else:

print("You are a child.")

**8. Loops in Python**

**(a) For Loop**

for i in range(5): # 0 to 4

print(i)

**(b) While Loop**

count = 0

while count < 5:

print(count)

count += 1

**(c) Loop Control Statements**

for i in range(5):

if i == 3:

break # Stops loop when i == 3

print(i)

**9. Functions in Python**

Functions help **reuse** code.

def greet(name):

print("Hello, " + name + "!")

greet("Alice")

**Returning Values**

def add(x, y):

return x + y

result = add(3, 5)

print(result) # Output: 8

**10. Lists in Python (Arrays)**

Lists store multiple values in a single variable.

fruits = ["Apple", "Banana", "Cherry"]

print(fruits[0]) # Output: Apple

fruits.append("Orange") # Add item

fruits.remove("Banana") # Remove item

print(fruits) # ['Apple', 'Cherry', 'Orange']

**11. Tuples (Immutable Lists)**

Tuples **cannot be changed** after creation.

colors = ("Red", "Green", "Blue")

print(colors[1]) # Output: Green

**12. Dictionaries (Key-Value Pairs)**

Dictionaries store **data in key-value pairs**.

person = {"name": "Alice", "age": 25}

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

person["city"] = "New York" # Add a new key-value pair

**13. Sets (Unique Values)**

Sets store **unique, unordered elements**.

numbers = {1, 2, 3, 3, 4}

print(numbers) # Output: {1, 2, 3, 4}

**14. Exception Handling (Try-Except)**

try:

x = int(input("Enter a number: "))

print(10 / x)

except ZeroDivisionError:

print("You can't divide by zero!")

except ValueError:

print("Invalid input! Please enter a number.")

**15. File Handling**

Python allows reading and writing files easily.

**Writing to a File**

file = open("example.txt", "w")

file.write("Hello, Python!")

file.close()

**Reading from a File**

file = open("example.txt", "r")

print(file.read())

file.close()

**16. Object-Oriented Programming (OOP)**

Python supports **OOP principles** like **classes and objects**.

class Car:

def \_\_init\_\_(self, brand, model):

self.brand = brand

self.model = model

def details(self):

return f"{self.brand} {self.model}"

car1 = Car("Toyota", "Corolla")

print(car1.details()) # Output: Toyota Corolla

**17. Modules and Libraries**

Python has **built-in and external modules**.

**Using a Built-in Module (math)**

import math

print(math.sqrt(25)) # Output: 5.0

**Installing External Modules**

pip install requests

import requests

response = requests.get("https://www.python.org")

print(response.status\_code)

**18. List Comprehension**

numbers = [x for x in range(10) if x % 2 == 0]

print(numbers) # Output: [0, 2, 4, 6, 8]

**19. Lambda Functions**

square = lambda x: x \*\* 2

print(square(4)) # Output: 16

**20. Decorators (Advanced Topic)**

def decorator(func):

def wrapper():

print("Function is being called")

func()

return wrapper

@decorator

def hello():

print("Hello, World!")

hello()

**Conclusion**

These basics will help you **build a strong foundation** in Python. After mastering these, you can explore: **Data Science** (Pandas, NumPy, Matplotlib)  
**Web Development** (Django, Flask)  
**Machine Learning** (Scikit-learn, TensorFlow)  
**Automation & Scripting**