TOPS TECHNOLOGY

Python – Collections, functions and Modules

Presented By:
Nandni Vala

Modules

1.Introduction to Python modules and importing modules.

- A **module** in Python is a file containing Python definitions and statements. It allows you to organize your code into reusable components. A module can contain functions, classes, variables, and runnable code.
- Example of a Simple Module :
- Create a file named mymodule.py:
- # mymodule.py
- def greet(name):
-     return f"Hello, {name}!"
- Importing Modules :
- To use the functionality from a module, you can **import** it using the import keyword.

- Types of Imports
- **Import the entire module**
- You can import the entire module and then use its functions with the module name.
- Example:
- import mymodule
- result = mymodule.greet("Alice")
- print(result) # Output: Hello, Alice!
- **Import specific functions or variables from a module**
- You can import specific items (e.g., functions, variables) from the module.
- Example :
- from mymodule import greet
- result = greet("Bob")
- print(result) # Output: Hello, Bob!

- Standard Library Modules
- Python comes with a **standard library** of modules that provide many useful functions, such as math, os, and random.
- Example:
- import math
- print(math.sqrt(16)) # Output: 4.0

2.Standard library modules: math, random.

- **math Module :**
- The math module provides mathematical functions and constants.

- Commonly Used Functions:
- **math.sqrt(x)**: Returns the square root of x.
- **math.pow(x, y)**: Returns x raised to the power of y.
- **math.factorial(x)**: Returns the factorial of x.
- **math.pi**: The constant π (pi), approximately 3.14159.
- **math.e**: The constant e, approximately 2.71828.
- Example:
- import math
- print(math.sqrt(16))   # Output: 4.0
- print(math.pow(2, 3))   # Output: 8.0
- print(math.factorial(5))   # Output: 120
- print(math.pi)   # Output: 3.141592653589793
- print(math.e)   # Output: 2.718281828459045

- **random Module**
- The random module implements pseudo-random number generators and provides functions to generate random numbers and select random elements.
- Commonly Used Functions:
- **random.randint(a, b)**: Returns a random integer between a and b (inclusive).
- **random.random()**: Returns a random floating-point number between 0.0 and 1.0.
- **random.choice(sequence)**: Returns a random element from a non-empty sequence.
- **random.shuffle(sequence)**: Shuffles the elements of the sequence in place.
- **random.sample(population, k)**: Returns a list of k unique elements randomly chosen from the population.
- Example:
- import random
- print(random.randint(1, 10)) # Output: Random integer between 1 and 10

3.Creating custom modules.

- To create a custom module in Python:
- **Create a .py file** (e.g., mymodule.py) containing functions, classes, or variables.
- **Import the module** in other Python files using import.
- Example:
- mymodule.py:
- def greet(name):
-     return f"Hello, {name}!"
- main.py:
- import mymodule
- print(mymodule.greet("Alice")) # Output: Hello, Alice!