VAULTOFCODES

TASK-2

❖ Functions and Modules

- What are Python Functions
- Function Declaration
- Types of Functions
- Create & Call a Function
- Advantages of Functions
- Disadvantages of Functions

- What are Python Modules
- Standard Library Modules
- Create a Module
- Import a Module
- Advantages of Modules
- Disadvantages of Modules

WHAT ARE PYTHON FUNCTIONS?

- A function is a block of code which only runs when it is called.
- You can pass data, known as parameters, into a function.
- A function can return data as a result.

FUNCTION DECLARATION:

The syntax to declare a function is:

```
def function_name(parameters):
        #function body
         return expression
```

- **function_name**: The name of the function.
- **parameters**: The input parameters that are passed to the function.
- **Function body**: The statements written within a function.

TYPES OF FUNCTIONS:

- The following are the different types of Python Functions:

1. **Built-in Functions**: Python's standard library includes number of built-in functions. Some of Python's built-in functions are print(), int(), type(), sum(),char() etc.
2. **Recursion Functions:** A recursive function is a function defined in terms of itself via **self-referential** expressions.
3. **Lambda Functions**: They are called as **anonymous function** that are defined without a name. While normal functions are defined using the **def** keyword in Python, anonymous functions are defined using the **lambda** keyword.
4. **User-defined Functions**: Functions that we define ourselves to do certain specific task are referred as user-defined functions.

CREATE AND CALL A FUNCTION:

- In Python, you create a function by using the **def** keyword. Let's look at an example of this.

Example:

```
In [1]: #To create a function
        def my_function():
            print("This is a function")
```

- The **def** keyword only creates and defines a function. To call the function, use the function name, followed by parentheses.

Example:

```
In [2]: #To call a function
        def my_function():
            print("This is a function")

        my_function()

        This is a function
```

ADVANTAGES OF FUNCTIONS:

- Enables **reusability** and reduces redundancy.
- Makes a code **modular**.
- Provides **abstraction** functionality.
- The program becomes **easy to understand** and manage.
- **Breaks** an extensive program into smaller and simpler pieces.

DISADVANTAGES OF FUNCTIONS:

- Programmers have less control over how they work and **less flexibility** to customize their behaviour.
- **Complexity:** Using too many functions can make the code harder to understand.
- **Maintenance:** Maintaining a large number of functions can be challenging.

WHAT ARE PYTHON MODULES?

- A module is simply a Python file with a **.py** extension that can be imported inside another Python program.
- The name of the Python file becomes the module name.
- The module contains — 1) **Definitions and Classes**

  2) **Variables**

  3) **Functions**

STANDARD LIBRARY MODULES

- The Python Standard Library is a collection of script modules accessible to a Python program to simplify the programming process and removing the need to rewrite commonly used commands.
- They can be used by 'calling/importing' them at the beginning of a script.
- The following are among the most important:

- time
- sys
- os
- math
- random
- pickle
- urllib
- re
- cgi
- socket

CREATE A MODULE

- Let's create a simple calc.py in which we define two functions, one **add** and another **subtract**.

```
# A simple module, calc.py

def add(x, y):
        return (x+y)

def subtract(x, y):
        return (x-y)
```

IMPORT A MODULE

- We can import the functions, and classes defined in a module to another module using the **import statement** in some other Python source file.
- **Syntax of Python Import :** `import module`

```
# importing module calc.py

import calc

print(calc.add(10, 2))
```

Output : 12

ADVANTAGES OF MODULES

- **Reusability** : Working with modules makes the code reusable.
- **Simplicity:** Module focuses on a small proportion of the problem, rather than focusing on the entire problem.
- **Scoping:** A separate namespace is defined by a module that helps to avoid collisions between identifiers.

DISADVANTAGES OF MODULES

- **Name Collisions :** The variables, functions or classes should not be with the same name.
- **Global state** : The number of modules increases significantly, which making it harder to manage and navigate through the project.
- **Complexity :** Modules introduce global state which can be problematic in larger codebases.

❖ DATA MANIPULATION IN PYTHON

- What is Data Manipulation in Python
- Data Manipulation Techniques in Python
- Most commonly used Python Libraries in Data Manipulation

✓ NumPy

✓ Pandas

✓ Matplotlib and Seaborn

- Advantages of Data Manipulation
- Disadvantages of Data Manipulation

WHAT IS DATA MANIPULATION IN PYTHON?

- Data manipulation is the process of organizing or arranging data in order to make it easier to interpret.
- Data manipulation in Python involves performing various operations on data to extract, transform, clean, and analyze it.
- The key feature of data manipulation is enabling faster business operations and also emphasize optimization in the process.

DATA MANIPULATION TECHNIQUES IN PYTHON

- Pandas
- NumPy
- Regular Expressions
- String Manipulation
- List Comprehensions
- Datetime Manipulation
- Grouping and Aggregation

PYTHON LIBRARIES IN DATA MANIPULATION

✓NUMPY

- NumPy is a fundamental library for numerical computations in Python.
- It provides support for multidimensional arrays and a wide range of mathematical functions, making it essential for data manipulation and scientific computing.

Example of using NumPy:

```
import numpy as np

# Create a NumPy array

data = np.array([1, 2, 3, 4, 5])

# Perform operations on the array

mean = np.mean(data)
```

✓ PANDAS

- Pandas is a powerful library for data manipulation and analysis.
- It provides data structures like DataFrames and Series, which make it easy to work with tabular data, perform data cleaning, filtering, aggregation, and more.

Example of using Pandas:

```
import pandas as pd

# Create a DataFrame

data = pd.DataFrame({'A': [1, 2, 3], 'B': [4, 5, 6]})

# Perform operations on the DataFrame

mean_A = data['A'].mean()
```

✓ MATPLOTLIB AND SEABORN

- These libraries are used for data visualization in Python, allowing you to create various types of plots and charts to explore and present your data visually.

Example of using Matplotlib:

```
import matplotlib.pyplot as plt
# Create a simple line plot
x = [1, 2, 3, 4, 5]
y = [10, 12, 5, 8, 9]
plt.plot(x, y)
plt.xlabel('X-axis')
plt.ylabel('Y-axis')
plt.title('Simple Line Plot')
plt.show()
```

ADVANTAGES OF DATA MANIPULATION:

- Easy to Use
- Simple Data Merging
- Scalability
- Open-source
- Versatility
- Cross Platform Compatibility
- Data is Flexible

DISADVANTAGES OF DATA MANIPULATION

- Slow Execution Speed
- Large Memory Consumption
- Dependency Management
- Not suitable for Mobile and Game Development
- Less suitable for real time Application

THANK YOU