**ASSIGNEMENT-3(2a)**

**1. Lists**

**Python Official Documentation:**

* **Key Points**:
  + Lists are ordered collections of items, mutable (modifiable), and can contain elements of different types.
  + Accessed by indexing and slicing (list[start:end]), and support various operations like appending, extending, and inserting elements.
  + Methods include append(), extend(), insert(), pop(), remove(), and sort() for manipulation.
* **Code Example**:

python

Copy code

# Creating a list

my\_list = [1, 2, 3, 4, 5]

# Accessing elements

print(my\_list[0]) # Output: 1

# Slicing

print(my\_list[1:4]) # Output: [2, 3, 4]

# Appending and extending

my\_list.append(6)

print(my\_list) # Output: [1, 2, 3, 4, 5, 6]

# Removing an element

my\_list.remove(3)

print(my\_list) # Output: [1, 2, 4, 5, 6]

**GeeksforGeeks:**

* **Key Points**:
  + Provides additional insights into list comprehensions for concise creation of lists based on existing lists.
  + Explains how to use built-in functions like map(), filter(), and reduce() with lists for functional programming paradigms.

**2. Functions**

**Python Official Documentation:**

* **Key Points**:
  + Functions are defined using def keyword, can accept parameters, and can return values using return.
  + Scope of variables (local and global), function annotations (def func(a: int, b: str) -> float:), and lambda functions (lambda arguments: expression).
  + Recursion, decorators (@decorator), and nested functions are supported.
* **Code Example**:

python

Copy code

# Defining a function

def greet(name):

return f"Hello, {name}!"

# Calling the function

print(greet("Alice")) # Output: Hello, Alice!

# Using lambda function

square = lambda x: x \*\* 2

print(square(5)) # Output: 25

**Real Python:**

* **Key Points**:
  + Focuses on practical usage of functions in Python, emphasizing modular and reusable code.
  + Discusses best practices for defining functions, including docstrings, parameter naming conventions, and handling return values effectively.

**3. Python Modules**

**Python Official Documentation:**

* **Key Points**:
  + Modules allow organizing Python code into files containing functions, classes, and variables.
  + Importing modules using import module\_name or from module\_name import function\_name.
  + \_\_name\_\_ attribute, used to determine if a module is run as a script or imported.
* **Code Example**:

python

Copy code

# Creating a module

# File: mymodule.py

def greet(name):

return f"Hello, {name}!"

if \_\_name\_\_ == "\_\_main\_\_":

print(greet("Alice"))

# Importing and using the module

import mymodule

print(mymodule.greet("Bob")) # Output: Hello, Bob!

**W3Schools:**

* **Key Points**:
  + Provides interactive examples for learning modules, demonstrating how to import and use them in Python scripts.
  + Emphasizes the role of modules in organizing code, enhancing reusability, and facilitating collaboration in larger projects.