**Task 1: Modifying Immutable Types Objective: Familiarize yourself with the immutable nature of certain data types.**

**Instructions:**

' Create a string variable with the value "Hello". ' Write a function to attempt to change the first character of the string to "J". ' Observe and note the error message you receive.

**Task 2: Mutable List Modifications**

**Objective: Understand the mutable nature of lists.**

**Instructions:**

' Create a list of integers. ' Write a function that accepts this list as an argument, appends a new integer to it, and returns the modified list. ' After calling the function, print the original list to observe the changes. Discuss why the original list was modified outside of the function.

**Task 3: Tuple with Mutable Elements**

**Objective: Grasp the concept that while tuples are immutable, their elements can be**

**mutable.**

**Instructions:**

• Create a tuple containing a list. ' Attempt to modify the list inside the tuple (e.g., append a new element). ' Print the tuple to see the changes. Discuss your findings.

**Task 4: Deep Copy vs Shallow Copy**

**Objective: Understand the differences between a deep copy and a shallow copy with mutable types.**

**Instructions:**

' Create a list containing sub-lists. • Make a shallow copy of this list. ' Modify the sub-list in the original list and observe the changes in both the original and the copied list. ' Now, make a deep copy of the list and repeat the modification. Observe the changes in the deep-copied list.

**Task 5: Mutable Default Arguments in Functions**

**Objective: Recognize the pitfalls of using mutable default arguments in functions.**

**Instructions:**

' Create a function that takes a list as an argument with a default value of an empty list. ' In the function, append a new item to the list every time the function is called. ' Call the function multiple times without providing a list argument and print the list after each call. Observe and discuss the behavior.

**Task 6: Create a python application of “Book Management System” to implement**

**mutability and immutability in real world application and paste screenshot of the**

**output:**

Features:

1. Add Book: Users can add a book with details like title, author, and publication year.

2. List Books: Users can view the list of all books.

3. Borrow Book: Users can marke book as borrowed.

4. Return Book: Users can mark a borrowed book as retuned

5. Search Book: Users can search for a book by its title or author.

Data Structures:

a. “books\_list”:A mutable list to store books. Each book is a dictionary with details like "id",

"title", “author”, "year", and "borrowed".

b. “porrowed\_books\_archive: An immutable tuple to archive books that have been

borrowed and returned.