CP

ASSIGNMENT 1

GROUP: **AYESHA ARSHAD, IQRA HAFEEZ**

ENROLLMENT: **01-131232-018, 01-131232-036**

SECTION: **BSE 1B**

DEPARTMENT OF SOFTWARE ENGINEERING

BAHRIA UNIVERSITY ISLAMABAD

Question 1: Shortest Path

1. Start by creating a list of all locations and initialize their distances from the source location as infinity, except for the source location itself, which is set to 0.
2. Also, maintain a list of visited locations, empty in the starting.
3. While there are unvisited locations:
    a. Select the unvisited location with the smallest distance from the source location.
    b. For the selected location, consider all its neighboring locations.
    c. Calculate the total distance of the neighbors from the source location through the current location.
    d. If this distance is less than the recorded distance for that neighbor, update the recorded distance.
4. Mark the current location as visited.
5. Repeat until you have visited all locations or until the destination location is marked as visited.
6. To find the shortest path, go back from the destination location to the source location using the recorded distances. This will give you the shortest path.

Question 2: Sort a list of numbers

1. Start by initializing an array.
2. Select a “pivot” element from the array.
3. This pivot is used to divide the array into two subarrays.
4. Divide the array into two subarrays, one with elements less than the pivot and another with elements greater than the pivot.
5. Recursively apply this function (Quicksort) to the subarrays until they are sorted.
6. Concatenate the sorted subarrays to obtain the final sorted array.

Quicksort is a good choice for sorting integers as it has a good average-case time complexity. However, other algorithms like merge sort and heap sort can be useful in other situations.

Question 3: Fibonacci Series

1. Start with creating an array fib to store the Fibonacci numbers.
2. Initialize it with a size of n+1 to handle the nth Fibonacci number.
3. Assign the first two Fibonacci numbers (0 and 1) to fib[0] and fib[1].
4. Use a loop to calculate the Fibonacci numbers from the 2nd to the nth.
5. In each iteration, calculate the Fibonacci number at index`` by summing the previous two Fibonacci numbers (fib[i-1] and fib[i-2]).
6. Stop at returning fib[n], which is the nth Fibonacci number.

Question 4: Inventory Management

1. Start by creating a database to store information about items in the inventory. Each item can store attributes such as name, description, price, and quantity.

2. Initialize with an empty inventory or existing data if available.

3. Adding items to the inventory:

Input: Item name, description, price, and quantity.

• Action:

- Check if the item already exists in the inventory.
- If it does, update the quantity by adding the new quantity to the existing one.
- If it doesn't, add a new entry with the provided information.

4. Removing items from the inventory:

Input: Item name.

Action:

- Check if the item exists in the inventory.
- If it does, remove the item's entry.
- If it doesn't, display an error message.

5. Updating item quantity:

Input: Item name, new quantity.

Action:

- Check if the item exists in the inventory.
- If it does, update the item's quantity to the new value.
- If it doesn't, display an error message.

6. Generating reports:

You can generate different types of reports based on your needs, such as:

- List all items in the inventory with their attributes (name, description, price, quantity).
- List items with low stock (where the quantity is below a specified threshold).
- Calculate the total value of the inventory (price* quantity for each item).
- Sort items based on name, price, or quantity.

7. Save changes: If you're using a database, make sure to save any changes to the inventory data.

8. Exit the system.