Bahria University, Islamabad

Department of Software Engineering

Computer Programming

(Fall-2023)

Teacher: Dr. Raja Suleman

Group Assignment: Sharafat Ullah &
Shoaib Akhter

Enrollment No: 01-131232-081 &
01-131232-067

Lab Journal:01

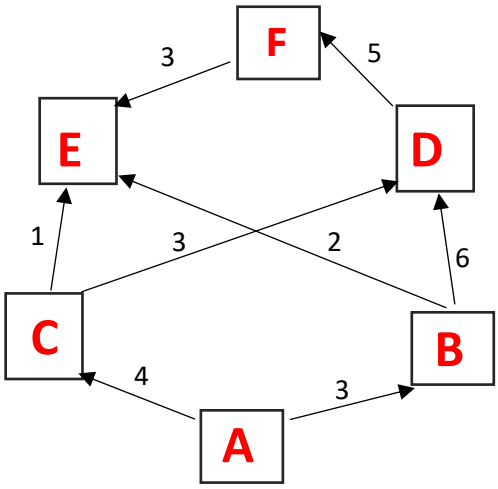
Date: 10/23/2023

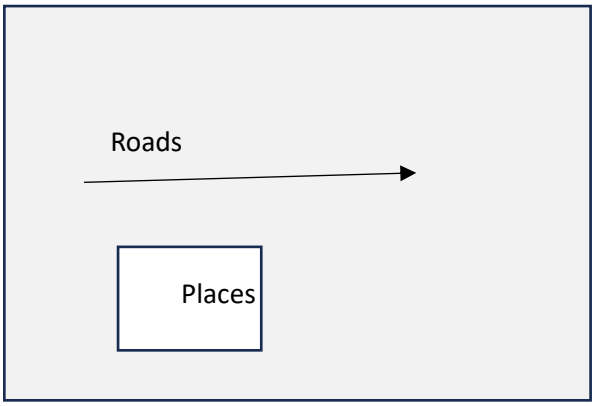
Comments:

Signature

TASK 1:

Imagine you are developing a GPS navigation system. You are given a map with various locations and the roads connecting them. Your task is to write an algorithm to find the shortest path from one location to another. You can assume that you have a list of locations and the distance between each pair of locations. Your algorithm should output the shortest path and the total distance?





- **Algorithm**

Step 1: *Start*

Step 1: Input your Given Locations (A to F) *{INPUT}*

Step 3: Use Dijkstra Algorithm to find the least distance. ***{FUNCTION}***
Step 4: Print the least distance one. Which. is A to B, E & F Or A to C, E & F. because in both routes the distance is 8. *{OUTPUT}*

Step 5: End

TASK 2:

Sorting a List of Numbers

Algorithm:

Let Suppose our List is (76,4,23,12,2,45,3,1,24,84)

Step 1: Start

Step 2: Enter the List of Numbers. (76,4,23,12,2,45,3,1,24,84) {Input}

*Step 3: Finding Ascending Order through **Quick sorting method.** {Function}*

Step 4: Print the Array. {Output}

Step 5: End.

Explanation:

Different Sorting method.

Quick Sorting:

- *Sort Quick is the best choice for this list.it has an average case time complexity of O (n log n) and usually perform well for various types of data.*
- *Quick Sort is particularly efficient when the list is relatively large, and its worst-case scenario is ess likely to occur on random input list.*

Merge Sorting:

- *Merge Sort is also a good option it consistently has a time complexity of O(n log n), making it efficient for sorting lists, regardless of the initial order.*
- *Merge Sort is stable, meaning it preserves the relative order of equal elements, wich can be a benefit in some scenarios.*

Bubble Sort:

- *Bubble Sort is not recommended for this list, especially if the list is large. It has a worst-case time complexity of O(n^2) and is generally less efficient than Quick Sort or Merge Sort.*

Counting Sorting:

- *Counting sort is not suitable for this list because its designed for lists wit small integer values and a limited range.*

*So, the best choice for Given List is **Quick Sorting**. As it will be more* efficient *and versatile for various input scenarios.*

TASK 3:

The Fibonacci sequence is a series of numbers where each number is the sum of the two preceding ones (e.g., 0, 1, 1, 2, 3, 5, 8, 13, ...). Write an algorithm to calculate the nth Fibonacci number. Your algorithm should be efficient and capable of handling large values of n.

Step 1: Start

Step 2: User Input For desirable Fibonacci Number.

Step 3:

Initialize two variables, a and b, both initially set to 0 and 1, respectively. Initialize "a" variable count to 2. This variable will keep track of the current Fibonacci number.

Step 4:

Check if n is less than 2. If it is, return n as the nth Fibonacci Number.

***Step 5**:*

if less than or equal to n, do the following:

***Step 6**:*

Calculate the next Fibonacci number by adding a and b. This will be the (count + 1)th Fibonacci number.

Step 7:

Update a with the value of b, Update b with the value of the newly calculated Fibonacci number. Increment count by 1.

***Step 8**:*

Once the loop is complete, b will contain the nt Fibonacci number.

***Step 9:** End.*

TASK 4:

You are tasked with creating an algorithm for a store's inventory management system. Your algorithm should be able to add and remove items from the inventory, update the quantity of existing items, and generate reports of the items and their quantities. Design an algorithm that efficiently manages the store's inventory based on these requirements.

Algorithm:

Step 1: Start

Step 2: *Initialize an empty data structure to store the inventory items.*

Step 3: *Create Functions for Add item, Remove item, Update Quantity and Report Card.*

Step 4*: Make Condition if someone is trying to remove or update an item that does not exist.*

Step 5: *End.*