COMPUTER PROGRAMMING # 01

DEPARTMENT OF SOFTWARE ENGINEERING

BAHRIA UNIVERSITY ISLAMABAD CAMPUS

GROUP MEMBERS :

1. AIZAZ AHMED (01-131232-106)

2. M. AWAIS (01-131232-051)

CLASS: BSE (1-A)

TEACHER NAME: DR RAJA SULAIMAN

DATE OF SUBMISSION: 29-10-2023

GETHUB RESPIRATORY LINK:

https://github.com/aizazahmed01/AIZAZ-S-ASSIGNMENTS/blob/e512c163e3df18a196df7f455db773883c518a82/CP%20ASSIGNMENT%2001%20DONE.pdf

Assignment 1: Problem Solving (CLO-2)

Question 1: Finding the Shortest Path:

PROBLEM 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.

PLANNING THE SOLUTION:
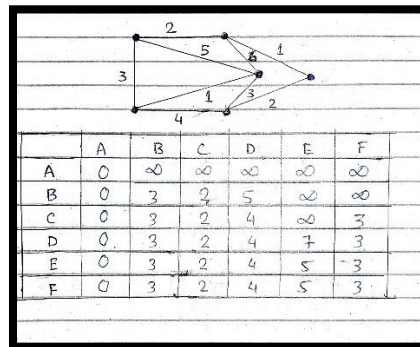
ALGORITHM:

STEP 1: Start

STEP 2: Let A be the current location B,C,D,E & F are the other unvisited locations.

STEP 3: Initialize A=0 & all the unvisited locations B=infinity, C=infinity, D=infinity, E=infinity & F=infinity.

STEP 4: Calculate the distance of all locations from current location.

STEP 5: Check if the current distance is shortest than the current location. Then update the distance for each neighboring location.

STEP 6: Make the current location as visited.

STEP 7: Check if the path is not shortest than move back to current location & find new path.

STEP 8: Do it for all the other unvisited locations.

**STEP 9:** So the total distance is the distance of destination from current location.

STEP 10: Stop.

Question 2: Sorting a List of Numbers

PROBLEM 2:

You are working on a project where you need to sort a list of numbers in ascending order. Design an algorithm to efficiently sort a list of integers. You should consider various sorting algorithms, evaluate their time complexity, and choose the most suitable one for the task.

PLANNING THE SOLUTION:
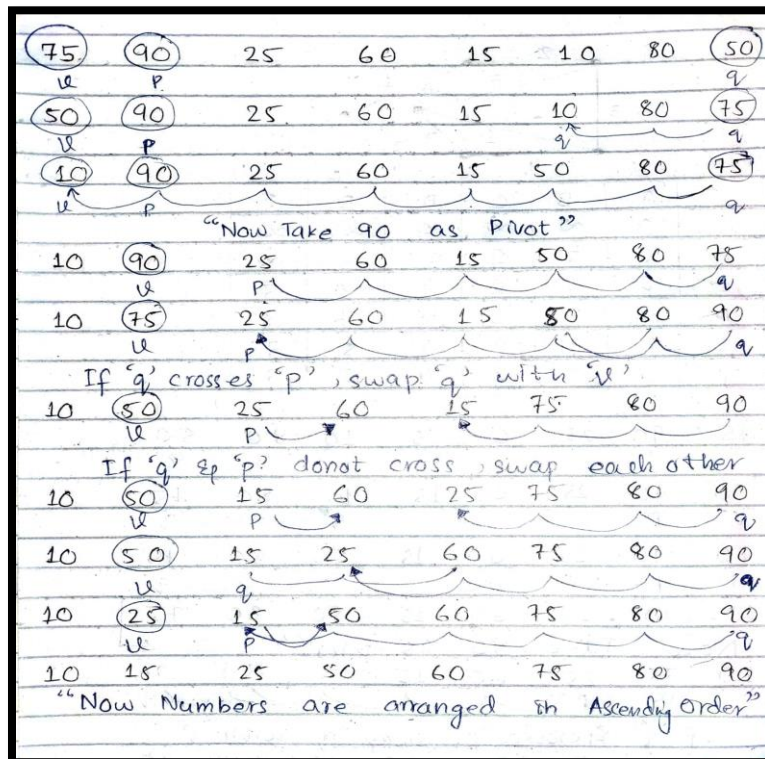
ALGORITHM:

STEP 1: Start

STEP 2: Choose a pivot number from the list of integers, choose the left most number as pivot number.

STEP 3: Read "p" , "q"

"p" will check the numbers from the list which are greater than pivot number & move to the right "q" will check the number from the list which are lesser than pivot number & move to the left.

STEP 4: Check if "p" is greater than pivot stop "p". At the same time, Check if "q" is lesser than pivot stop "q".

STEP 5: Swap the value of "p" & "q"

STEP 6: If "q" crosses p, swap "q" , swap "q" with the pivot.

STEP 7: Do this till all the values are arranged in ascending order.

STEP 8: Combine the list to get the final sorted list.

STEP 9: Stop

Question 3: Calculating Fibonacci Numbers

PROBLEM 2:

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.

ALGORITHM:

STEP 1: Start

STEP 2: Initialize two variables a=0, b=1.

STEP 3: Initialize 'num' to store the Fibonacci series.

STEP 4: If 'num' is equal to '0' return empty.

STEP 5: If 'num' is equal to '1' return a.

STEP 6: If 'num' is equal to '2' return (a,b).

STEP 7: If 'num' is greater than '3' loop (num2) times.

STEP 8: In each interaction, calculate the next number by adding 'a' & 'b' & then setting 'a' to 'b' & 'b' to new Fibonacci number.

STEP 9: Return the Fibonacci series containing nth terms.

STEP 10: Stop.

Question 4: Inventory Management

PROBLEM 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: Create an inventory data structure to store item information.

STEP 3: Initialize the inventory data structure.

STEP 4: Create a function to add items to inventory.

Input Stock Keeping Unit, item name, quantity & any additional item details. If Stock Keeping Unit already exists in the inventory then update the existing item list if it doesn't exist in the inventory list then create new item entry with the provided item information.

STEP 5: Create a function to remove items from inventory.

Input Stock Keeping Unit, and the quantity to be removed. If the quantity

STEP 6: Create a function to update item quantity.

Input Stock Keeping Unit and the new quantity. If the quantity already exists then update its quantity to new.

STEP 7: Create function to generate reports based on user requirments.

List all the items in inventory, items with low stock, items that need restocking and allow user to specify the criteria for reports.

STEP 8: Stop.