Bahria University, Islamabad

Department of Software Engineering

Computer Programming

Teacher: Dr. Raja Suleman

Assignment 1 (Algorithms)

Made By:

- **Mudassir Riaz** **01-131232-046**
- **Muhammad Awais** **01-131232-052**

Date:29/October/2023

QUESTION # 1

Imagine you are developing a GPS navigation system. You are given a map with various locations and the road 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:

1. **START.**
2. First, we will Initialize graph as a dictionary where keys represent the locations or (nodes) and values represent dictionaries containing neighboring locations and their associated distances (edges). Then we Initialize the starting location and the end location as the destination location (Input)**.**
3. Create a dictionary of the distances to store the shortest distance from the starting to every other location. We will Initialize it with infinity for all locations except for the start location, which should have distance of 0. Then we will create an empty set of visited to keep the track of visited locations and a dictionary to store the previous location for each location. (Initializing)
4. While there are some unvisited locations:
   - Select the unvisited location with the smallest distance from the source.
   - Mark the selected location as visited by adding them to the visited set.
   - For the selected location, consider all its unvisited neighbors and calculate their tentative distance from the starting location.
   - Apply condition that if shortest path is found we will update the distances in the distance dictionary and set the previous location in the previous dictionary.
   - At last, we would have all the visited locations, we can easily track back the shortest path to any location just by checking the start and end location and also by starting from the end point and follow the shortest path until it reaches the source.
5. Display the shortest path or route and total distance of that shortest path as output.
6. **END.**

QUESTION # 2

Suppose you are working on a project where you need to sort the 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.

1. **START.**
2. Consider a number series of three numbers.
3. Declare Variables:
   - num1= first number.
   - num2= second number.
   - num3= third numbers.
4. Take input from the user of three numbers.
5. Compare num1 with num2 and apply a If condition that if num1 is greater than num2, swap the values of each other.
6. Compare num2 and num3 and apply a If condition that if num2 is greater than num3, swap the values.
7. Compare num1 and num2 if only they were swapped before and apply the same If condition that if num1 is greater than num2, swap them.
8. Display Output as ascending order : num1,num2,num3.
9. **END.**

QUESTION # 3

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

Algorithm:

1. **START.**
2. **Declare variables:**
   - **n=** number of terms in the Fibonacci series.
   - **i=** counter variable for the loop.
   - **t1=** first term of series.
   - **t2=** second term of series.
   - **next_term=** next term of the series or the nth value of the Fibonacci series.
3. Take input of how many terms the user wants in the series (take input of "n" integer).
4. Display Fibonacci Series as a heading on the console screen.
5. Start a **For Loop** with counter variable "i" , condition (i<=n) and an increment.
6. **Body of For Loop:**
   - Use a If- Condition (If "i" is equal to 1, the loop will print the 1st term(t1)).
   - Then after this condition, a continue statement will force the loop to the starting point and will increment the value of i.
   - Use another If- Condition (If "i" is equal to 2, the loop will print the 2st term(t1)).
   - Then after this condition, a continue statement will force the loop to the starting point and will increment the value of i.
   - Calculate the next term by adding term 1 and term 2 (next_term=t1+t2).
   - Assign value of t2 to t1. (t1=t2)
   - Assign a new value to t2. (t2=next_term)
   - Print next term and a comma.
   - The loop will keep running until the condition becomes false.
7. When the loop will end, the program will also end with displaying Fibonacci series of nth term on the screen.
8. **END.**

QUESTION # 4

Create an Algorithm for a store's inventory management system. Your algorithm should be able to add or 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:

1. **START.**
2. Declare a character variable (choice).
3. Display the menu output like:

Store Management

i. Press '1' to add items.
ii. Press '2' to remove items.
iii. Press '3' to update existing items.
iv. Press '4' to generate report of the quantities.
v. Press '5' or 'Q' to exit.

4. Display a message of "Enter your option".
5. Take input of user choice.
6. Use a switch statement with expression (choice):
   - Case '1': Add items.
   - Case '2': Remove items.
   - Case '3': Update existing items.
   - Case '4': Generate report of quantities.
   - Case ('5'|| 'Q'): Exit and display message of thank you.
7. There is also a default option if the user inputs incorrect data type.
8. End of switch statement.
9. **END.**

❖ **Github Repositry Link(Mudassir Riaz):**

https://github.com/mudassirriaz64/Computer-Programming-Assignment-1.git

❖ **Github Repositry Link(Awais Ishaq):**

https://github.com/AwaisIshaq7/Computer-Programming-Assigns.git