Assignment 1: Problem Solving

Algorithms

Question 1: Finding the Shortest Path

- Note the coordinates (longitude and latitude) of the origin/ starting point of the person.
- List all the routes/ paths available to the destination.
- Each route must have a series of waypoints with their respective coordinates.
- Using the haversine formula:

  $a = \sin^2(\Delta latDifference/2) + \cos(lat1).\cos(lt2).\sin^2(\Delta lonDifference/2)$

  calculate the distance between the starting point and each waypoint along the route. Then sum up these distances to find the total distance of the route.

- Calculate the distances of all the possible routes.
- Compare the distances of the routes to find the shortest/ quickest route.
- Provide the user with the shortest route and the total distance.

Question 2: Calculating Fibonacci Numbers

- Take an integer as input (should not be less than or equal to 0)
- If the integer is less than or equal to zero, the input is invalid.
- If the integer = 1, it's result is 0, and if the integer = 2, then result = 1.
- If the integer is greater than 2, then add the (integer-1) and (integer-2) Fibonacci numbers.
- Display the output.

Question 3: Inventory Management

- Give every item a unique serial number and store its relative information (price, quantity) along with it in the inventory.
- To add a new item on the list, check whether the item already exits or not, if it does, then just update its quantity, if it doesn't, then create a new entry with its own unique identifier and details.
- To remove an item from the inventory, find the item by its serial number and delete the entry, if the item doesn't exist, display an error message.
- To update an item, search up the item and change its quantity, price, or category.
- To generate its inventory report, display list of items with their complete details (serial number, product name, price, quantity, category, expiry etc.).

Question 4: Inventory Management

- Get a series of numbers as input from the user and store it in a list named 'list_A'.
- First find the smallest number in the list by comparing it to all the entries and store it in 'list_B'.
- After the number has been stored in a separate list, then omit that number from 'list_A'.
- Then find the smallest number from the 'list_A' and store it in 'list_B' again in such an order that it comes after the previous number.
- Keep on repeating it until 'list_A' runs out of entries.

Github link: https://github.com/ninsaga/CP