Approach

Hill Climbing

My main goal is to find the shortest path to visit all locations

• First, I take input for the number of locations and the coordinates of the locations.
• I call a function and pass the locations, warehouse, and the location number
• I use the Euclidean function for calculating heuristic values.
• I write a function named **total_distance_for_A_Route**, which takes a route as input and returns the total cost of all locations according to the visit.

• Then, I create a new random route as a list, which is an initial route. I compare another route with this route. Then I add the warehouse at the end of the route because I have to return warehouse after delivery in all locations
• Find the distance or cost for the initial route
• Then I create a new route from the initial route swapping by two locations after that I find out cost for this new route.

a) If the cost of new route is less than older route then I keep this new route and cost and so on
b) If the cots of new route is greater than older route then I breake the algorithm and return the stored route and cost of this

OUTPUT

Path: 1-> 2-> 5-> 3-> 4->1

Cost for this path is: 17.28

Simulated Annealing

My main goal is to find the shortest path to visit all locations

• First, I take input for the number of locations and the coordinates of the locations.

a) Assume initial Temperature is 100
b) Cooling rate is 95%

• I call a function and pass the locations, warehouse, location number, initial temperature, cooling rate
• I use the Euclidean function for calculating heuristic values.
• I write a function named **total_distance_for_A_Route**, which takes a route as input and returns the total cost of all locations according to the visit.

• Then, I create a new random route as a list, which is an initial route. For I compare another route with this route. Then I add the warehouse at the end of the route because I have to return warehouse after delivery in all locations
• Find the distance or cost for the initial route
• Then I create a new route from the initial route, swapping by two locations after that I find out cost for this new route.

a) If the cost of the new route is greater than the older route, then
    I. Then I find probability and random generated number
    II. If a random generated number is less than or equal of probability then I keep this **bad** route
b) If the cost of the new route is less than older route then I store or keep this route and cost of this
c) If temperature is 0 then I breake the algorithm and return the route or path and cost

OUTPUT
Path: 1-> 2-> 4-> 5-> 3->1
Cost for this path is: 14.67

Discuss

Inputs:

Enter the Coordinate for location-> 1: **0 0**

Enter the Coordinate for location-> 2: **2 3**

Enter the Coordinate for location-> 3: **4 0**

Enter the Coordinate for location-> 4: **4 3**

Enter the Coordinate for location-> 5: **6 1**

For Hill climbing

Path: 1-> 2-> 5-> 3-> 4->1

Cost for this path is: 17.28

For Simulated Annealing

Path: 1-> 2-> 4-> 5-> 3->1

Cost for this path is: 14.67

Here, simulated annealing is given a better solution because it's able to reduce local minima by accepting a bad neighbor based on probability.
But the Hill climbing is stuck when he gets a bad neighbor. That's why simulated annealing give the better solution