In Traveling Salesperson Problem (TSP), the task is to find the shortest distance tour passing through each node of the graph exactly once.

As input, you will be given the (x,y) coordinates of n cities, from which it is possible to compute the intercity distances as eucildean distance between two cartesian coordinates. We assume a complete graph which means from any one vertex, it is possible to visit all the other (n-1) vertices using a single edge.

Two types of heuristic ways to find the solution.

(1) Construction heuristics: builds a solution from scratch (starting with nothing). Often called "greedy heuristics".

(2) Improvement heuristics (neighborhood search): starts with a solution, and then tries to improve the solution, usually by making small changes in the current solution.

For this task, you need to implement two construction heuristics and two improvement heuristics.

Construction heuristics:

(A) Nearest Neighbor:

Step 1. Start with any node as the beginning node
Step 2. Find the unvisited node closest to the last node added to the path. Add this node to the path.
Step 3. Repeat Step 2 until all nodes are contained in the path. Then join the first and last nodes.

(B)

In insertion heuristic methods, we use a heuristic in which we add one city at a time, but the next city can be added anywhere in the tour (not just the beginning or the end.)

An insertion procedure takes a sub-tour on k nodes at iteration k and determines which of the remaining n-k nodes shall be inserted to the sub-tour next (the selection step) and where (between which two nodes) it should be inserted (the insertion step).

(B1) Nearest Insertion

Step 1. Start with a sub-graph consisting of node i only.
Step 2. Find node r such that Cir is minimal and form sub-tour i-r-i.
Step 3. (Selection step) Given a sub-tour, find node r not in the sub-tour closest to any node j in the sub-tour; i.e. with minimal Crj
Step 4. (Insertion step) Find the arc (i, j) in the sub-tour which minimizes Cir + Crj - Cij . Insert r between i and j.
Step 5. If all the nodes are added to the tour, stop. Else go to step 3

(B2) Cheapest Insertion

Step 1. Start with a sub-graph consisting of node i only.
Step 2. Find node r such that Cir is minimal and form sub-tour i-r-i.
Step 3. Find (i, j) in sub-tour and node r, not in the sub-tour, such that Cir + Crj - Cij is minimal. Insert r between i and j.

In cheapset insertion, the city is added to the subtour whose addition has the least cost increment.

(C)
Improvement heuristics:

Improvement heuristics start with a feasible solution and look for an improved solution that can be found by making a very small number of changes.

(C1) 2-opt heuristic for 2-opt neighborhood search:
Two TSP tours are called 2-adjacent if one can be obtained from the other by deleting two edges and adding two edges.
A TSP tour T is called 2-optimal, if there is no 2-adjacent tour to T with lower cost than T.

Improvement heuristics are based on searching a "neighborhood." Let N(T) be the neighborhood of tour T. In this case, the N(T) consists of all tours that can be obtained from T deleting two edges and inserting two edges.

Look for an improvement obtained by deleting two edges and adding two edges. Look for a 2-adjacent tour with lower cost than the current tour which is an improved tour. In first choice hill-climbing, if one improved tour is found, then it replaces the current tour. In steepest ascent hill-climbing, this neighbourhood search continues until there is a 2-optimal tour. You may also experiment with stochastic hill-climbing which chooses at random from among the uphill (improvment) moves.

(C2) 3-opt heuristic for 2-opt neighborhood search: It is similar to 2-opt heuristic, but here the neighbrhood consists of all tours that can be obtained by deleting three edges and inserting three edges.

You can find the pseudocode of 2-opt and 3-opt heuristic from their wikipedia pages.