The Traveling Salesman Problem (TSP) is a Central Problem in Combinatorial Optimization which can be solved using various constructive and improvement heuristics.

In this assignment, you are to implement The Nearest Neighbour Heuristic (NNH) and Savings Heuristic (SH) which are constructive algorithms. Also, the k-opt iterative improvement algorithm is to be implemented by you (for k = 2).

The tasks:
In all datasets, the *N* points represent specific locations in some city or country. There are *n* pairs of (*x*, *y*) co-ordinates in the data file.

Name of datasets	No of locations	Optimal tour cost
pr 76	76	108159
berlin52	52	7542
st70	70	675

Task-1:
Run the greedy_simple version of the NNH and SH constructive algorithms with different locations or vertices as starting location. For example- for berlin52, we may start with any of the 52 locations. For this task, find out which location gives the shortest cost tour. Also,report on the costs of average, best, and worst cases. It is possible to have 76, 52, and 70 cases for these three datasets, respectively. Use a restricted number of k = 5 cases (you may use a larger k) and pick these cases (each with a different starting location) randomly. At the end of task-1, you should save the obtained best tour out of *k* cases.

The greedy_simple results						
	Avg. cases		Best Case		Worst Case	
	NNH	SH	NNH	SH	NNH	SH
pr76						
berlin52						
st70						

Task-2:
After finishing the task-1, we know which is the best starting location for NNH and SH constructive algorithms. For this starting location, construct TSP tours using the greedy_randomized version of the NNH and SH constructive algorithms. In the greedy randomized version, you are to choose any of the k-best neighbors, instead of choosing the best neighbor as was done for the greedy_simple version. For reporting the result, use a restricted number of k = 5 which means keep 5 best neighbours to choose for moving. Also, run n = 10 cases (each with the same starting location) and report the avg., best and worst cases. Since you are using a randomized algorithm, in each case the result would be different. At the end of task-2, you should save the best three tours out of *n* cases.

The greedy_randomized results						
	Avg. cases		Best Case		Worst Case	
	NNH	SH	NNH	SH	NNH	SH
pr76						
berlin52						
st70						

Task-3:

For each of NNH and SH constructive algorithms, do the following:
After finishing task-2, the best three tours were picked up. In addition, pick up the best tour found for task-1. Now run 2-opt iterative improvement algorithm for each of four tours and report the result for avg, best and worst cases out of these four cases.

You can implement for best improvement and for first improvement versions. For best improvement version, we select the best neighbor out of all neighbors for moving. On the other hand, we move to the first neighbor which shows improvement over the current solution, as soon as it is found in first_improvement version.

The 2-opt results for best improvement						
	Avg. cases		Best Case		Worst Case	
	NNH	SH	NNH	SH	NNH	SH
pr76						
berlin52						
st70						

The 2-opt results for first improvement						
	Avg. cases		Best Case		Worst Case	
	NNH	SH	NNH	SH	NNH	SH
pr76						
berlin52						
st70						

Now, provide a comparison of the obtained best tour (both best improvement and first improvement) with the optimal tour for the datasets. Assume the optimal tour cost to be 100%. For example, if the obtained best tour may be 1000 (200% of 500) and 750 (150% of 500), while the best tour is 500.

Performance Comparison						
	Optimal cost		Best Improvement		First Improvement	
	Actual cost	Percentage	Actual cost	Percentage	Actual cost	Percentage
			NNH	SH	NNH	SH
pr76	108159	100				
berlin52	7542	100				
st70	675	100				