Experiment - 8

8. Implementation of Hill Climbing Algorithm using LISP/PROLOG

```
Import random
def randomSolution(tsp):
cities=list (range(len(tsp)))
Solution= []
for I in range (len (tsp)):
randomcity =cities[random.randint(0,len(cities)-1)]
 solution.append(randomcity)
cities.remove(randomcity)
return solution
def routelength(tsp,solution):
routelenght = 0
for I in range (len(solution)):
routelength  += tsp[solution[i-1]][solution[i]]
return routelength
def getNeighbours(solution):
neighbours = []
for I in range(len(solution)):
for j in range(i+1,len(solution)):
neighbours =solution.copy()
neighbour[i]=solution[j]
neighbour[j]=solution[i]
return neighbours
def getbestNeighbour(tsp,neighbours):
while BestNeighbourRoutelength<currentRoutelength:
currentSolution=bestNeighbour
currentRouteLength=bestNeighbourRouteLength
neighbours=getNeighbours(currentSolution)bestNeighbour
BestNeighbourRouteLength=getbestNeighbour (tsp, neighbour)
return currentSolution, currentRouteLength
def main ():
tsp=[
      [0,400,500,300]
      [400, 0,300,500]
      [500, 300, 0, 400]
      [300, 500, 400, 0]
      ]
print (hillclimbing (tsp))
If_name_ == "_main_":
main ()
```

Output:

```
([0, 1, 2, 3], 400)
```