EXPERIMENT 6

Python programming for hill climbing search

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):

routeLength = 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)):

neighbour = solution.copy()

neighbour[i] = solution[j]

neighbour[j] = solution[i]

neighbours.append(neighbour)

return neighbours

def getBestNeighbour(tsp, neighbours):

bestRouteLength = routeLength(tsp, neighbours[0])

bestNeighbour = neighbours[0]

for neighbour in neighbours:

currentRouteLength = routeLength(tsp, neighbour)

if currentRouteLength < bestRouteLength:

bestRouteLength = currentRouteLength

bestNeighbour = neighbour

return bestNeighbour, bestRouteLength

def hillClimbing(tsp):

currentSolution = randomSolution(tsp)

currentRouteLength = routeLength(tsp, currentSolution)

neighbours = getNeighbours(currentSolution)

bestNeighbour, bestNeighbourRouteLength = getBestNeighbour(tsp, neighbours)

while bestNeighbourRouteLength < currentRouteLength:

currentSolution = bestNeighbour

currentRouteLength = bestNeighbourRouteLength

neighbours = getNeighbours(currentSolution)

bestNeighbour, bestNeighbourRouteLength = getBestNeighbour(tsp, neighbours)

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

