UNIVERSIDAD AUTÓNOMA DE NUEVO LEÓN

Facultad de Ingeniería Mecánica y Eléctrica

Homework 3

Due date: Mon 19-Sep-2025 22:00

Course: Selected Topics on Optimization

Semester: Fall 2025

Student name: Adriana Concepción Aguirre Dorantes 2029675

The nearest neighbor heuristic:

Solution: [1, 5, 8, 7, 2, 3, 6, 4, 1] cost 235

When multiple tours had equal cost, I selected the canonical representative by rotating the cycle to start at the smallest node index; ties during the algorithm were broken by choosing the smallest node index.

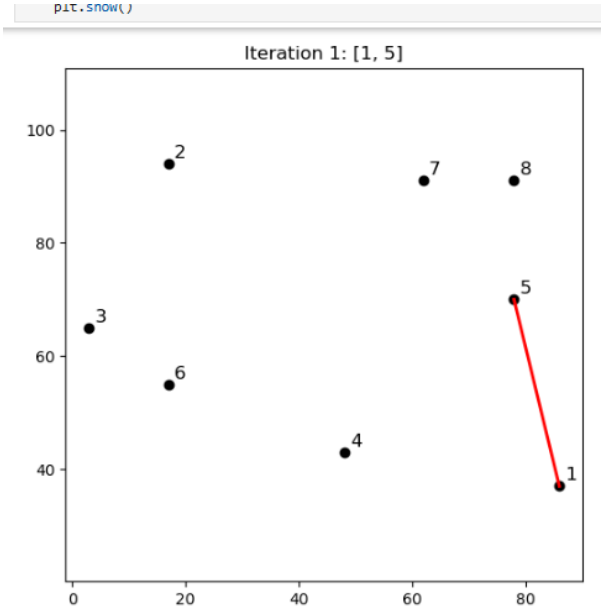
Start from a node, at each step go to the nearest unvisited node. Finally, return to the start. To improve: repeat starting from each node and select the best tour:

```
NearestNeighbor_TSP(D[1..n,1..n]):
    bestTour ← ∅
    bestCost ← ∞

    for start in {1, 2, ..., n}:
        tour ← [start]
        cost ← 0
        unvisited ← {all nodes} \ {start}
        current ← start

        while unvisited ≠ ∅:
            next ← argmin_{j ∈ unvisited} D[current, j]
            cost ← cost + D[current, next]
            tour.append(next)
            unvisited.remove(next)
            current ← next
        cost ← cost + D[current, start]
        tour.append(start)

        if cost < bestCost:
            bestCost ← cost
            bestTour ← tour

    return bestTour, bestCost
```

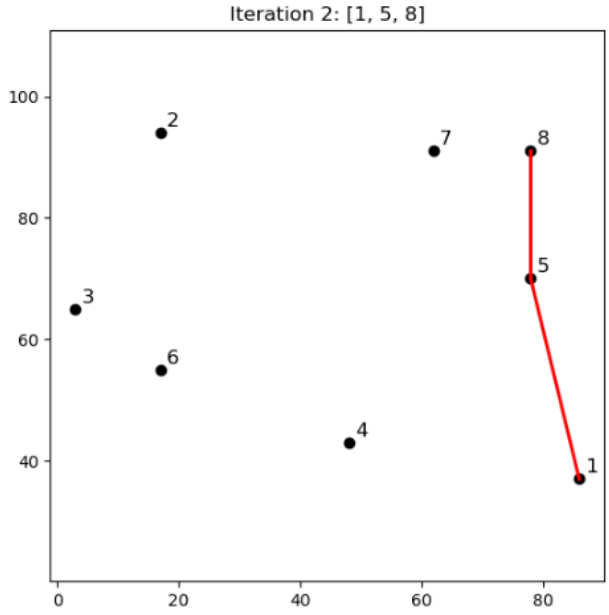
ITERATIONS

NN start 1: tour [1, 5, 8, 7, 2, 3, 6, 4, 1] cost 235

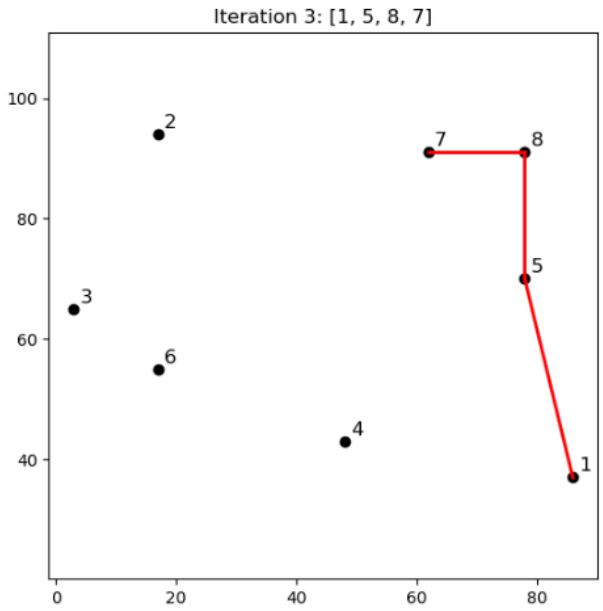
iter 1: from 1 -> 5 (d=33); cumulative after = 33



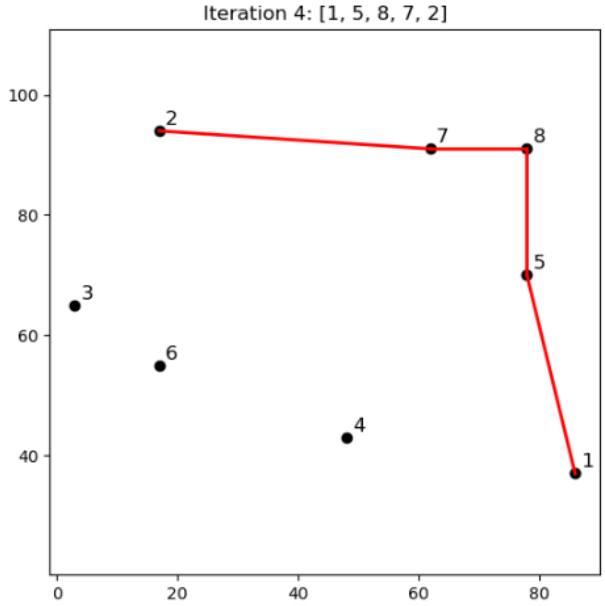
iter 2: from 5 -> 8 (d=21); cumulative after = 54



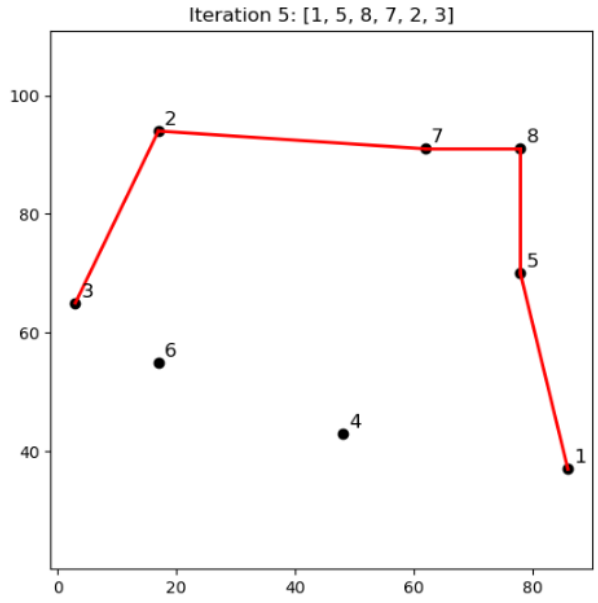
iter 3: from 8 -> 7 (d=16); cumulative after = 70



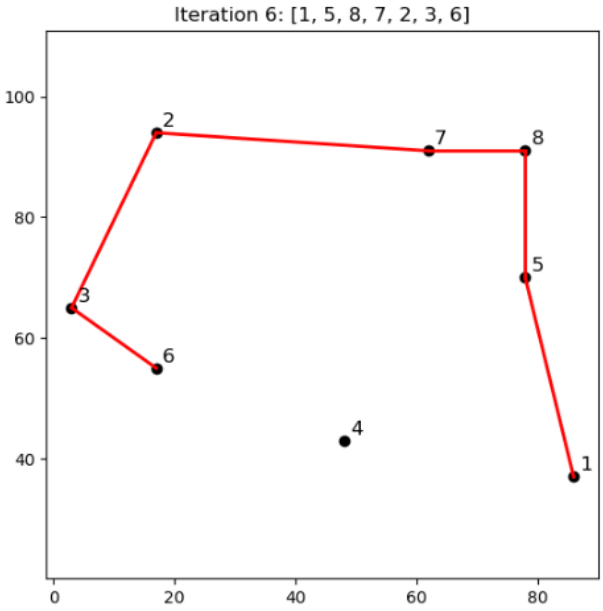
iter 4: from 7 -> 2 (d=45); cumulative after = 115



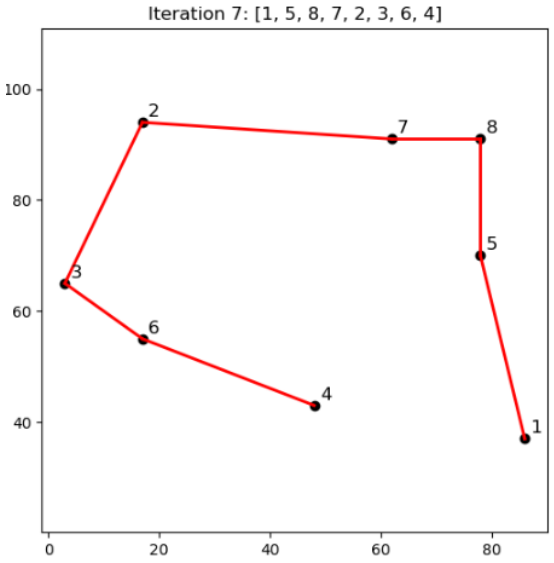
iter 5: from 2 -> 3 (d=32); cumulative after = 147



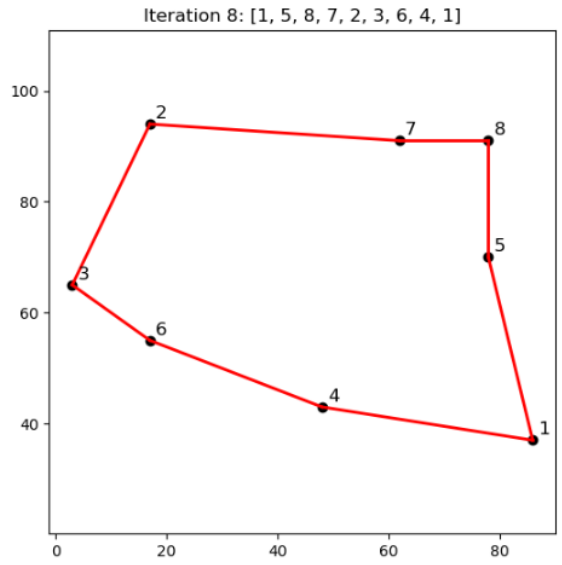
iter 6: from 3 -> 6 (d=17); cumulative after = 164



iter 7: from 6 -> 4 (d=33); cumulative after = 197



iter 8: from 4 -> 1 (d=38); cumulative after = 235



NN start 2: tour [2, 3, 6, 4, 1, 5, 8, 7, 2] cost 235

iter 1: from 2 -> 3 (d=32); cumulative after = 32

iter 2: from 3 -> 6 (d=17); cumulative after = 49

iter 3: from 6 -> 4 (d=33); cumulative after = 82

iter 4: from 4 -> 1 (d=38); cumulative after = 120

iter 5: from 1 -> 5 (d=33); cumulative after = 153

iter 6: from 5 -> 8 (d=21); cumulative after = 174

iter 7: from 8 -> 7 (d=16); cumulative after = 190

iter 8: from 7 -> 2 (d=45); cumulative after = 235

NN start 3: tour [3, 6, 4, 1, 5, 8, 7, 2, 3] cost 235

iter 1: from 3 -> 6 (d=17); cumulative after = 17

iter 2: from 6 -> 4 (d=33); cumulative after = 50

iter 3: from 4 -> 1 (d=38); cumulative after = 88

iter 4: from 1 -> 5 (d=33); cumulative after = 121

iter 5: from 5 -> 8 (d=21); cumulative after = 142

iter 6: from 8 -> 7 (d=16); cumulative after = 158

iter 7: from 7 -> 2 (d=45); cumulative after = 203

iter 8: from 2 -> 3 (d=32); cumulative after = 235

NN start 4: tour [4, 6, 3, 2, 7, 8, 5, 1, 4] cost 235

iter 1: from 4 -> 6 (d=33); cumulative after = 33

iter 2: from 6 -> 3 (d=17); cumulative after = 50

iter 3: from 3 -> 2 (d=32); cumulative after = 82

iter 4: from 2 -> 7 (d=45); cumulative after = 127

iter 5: from 7 -> 8 (d=16); cumulative after = 143

iter 6: from 8 -> 5 (d=21); cumulative after = 164

iter 7: from 5 -> 1 (d=33); cumulative after = 197

iter 8: from 1 -> 4 (d=38); cumulative after = 235

NN start 5: tour [5, 8, 7, 2, 3, 6, 4, 1, 5] cost 235

iter 1: from 5 -> 8 (d=21); cumulative after = 21

iter 2: from 8 -> 7 (d=16); cumulative after = 37

iter 3: from 7 -> 2 (d=45); cumulative after = 82

iter 4: from 2 -> 3 (d=32); cumulative after = 114

iter 5: from 3 -> 6 (d=17); cumulative after = 131

iter 6: from 6 -> 4 (d=33); cumulative after = 164

iter 7: from 4 -> 1 (d=38); cumulative after = 202

iter 8: from 1 -> 5 (d=33); cumulative after = 235

NN start 6: tour [6, 3, 2, 7, 8, 5, 1, 4, 6] cost 235

iter 1: from 6 -> 3 (d=17); cumulative after = 17

iter 2: from 3 -> 2 (d=32); cumulative after = 49

iter 3: from 2 -> 7 (d=45); cumulative after = 94

iter 4: from 7 -> 8 (d=16); cumulative after = 110

iter 5: from 8 -> 5 (d=21); cumulative after = 131

iter 6: from 5 -> 1 (d=33); cumulative after = 164

iter 7: from 1 -> 4 (d=38); cumulative after = 202

iter 8: from 4 -> 6 (d=33); cumulative after = 235

NN start 7: tour [7, 8, 5, 1, 4, 6, 3, 2, 7] cost 235

iter 1: from 7 -> 8 (d=16); cumulative after = 16

iter 2: from 8 -> 5 (d=21); cumulative after = 37

iter 3: from 5 -> 1 (d=33); cumulative after = 70

iter 4: from 1 -> 4 (d=38); cumulative after = 108

iter 5: from 4 -> 6 (d=33); cumulative after = 141

iter 6: from 6 -> 3 (d=17); cumulative after = 158

iter 7: from 3 -> 2 (d=32); cumulative after = 190

iter 8: from 2 -> 7 (d=45); cumulative after = 235

NN start 8: tour [8, 7, 5, 1, 4, 6, 3, 2, 8] cost 256

iter 1: from 8 -> 7 (d=16); cumulative after = 16

iter 2: from 7 -> 5 (d=26); cumulative after = 42

iter 3: from 5 -> 1 (d=33); cumulative after = 75

iter 4: from 1 -> 4 (d=38); cumulative after = 113

iter 5: from 4 -> 6 (d=33); cumulative after = 146

iter 6: from 6 -> 3 (d=17); cumulative after = 163

iter 7: from 3 -> 2 (d=32); cumulative after = 195

iter 8: from 2 -> 8 (d=61); cumulative after = 256

The nearest insertion heuristic:

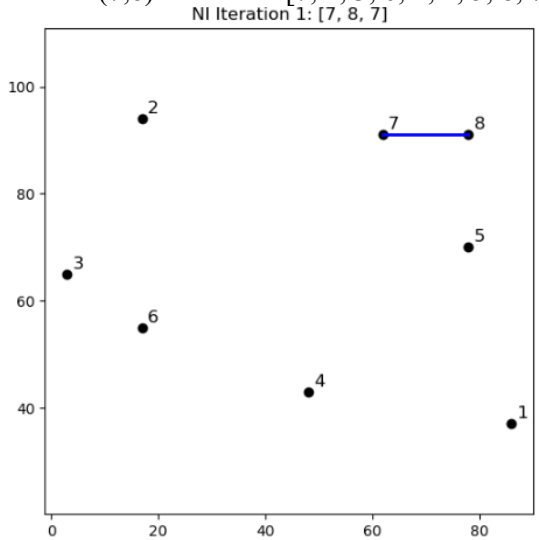
Solution: final tour [7, 2, 3, 6, 4, 1, 5, 8, 7] cost 235

1. Start with the smallest edge (i,j) → initial subtour [i, j, i].
2. At each step, find the nearest unvisited node to the current tour.
3. Insert it in the position that increases the tour length the least.
4. Repeat until all nodes are included.

```
NearestInsertion_TSP(D[1..n,1..n]):
   (i, j) ← argmin_{i≠j} D[i,j]
   tour ← [i, j, i]
   unvisited ← {all nodes} \ {i, j}

   while unvisited ≠ ∅:
      v ← argmin_{k ∈ unvisited} min_{u ∈ tour} D[k,u]
      bestPos ← -1
      bestIncrease ← ∞
      for p = 1 to length(tour)-1:
         u ← tour[p]
         w ← tour[p+1]
         increase ← D[u,v] + D[v,w] - D[u,w]
         if increase < bestIncrease:
            bestIncrease ← increase
            bestPos ← p

      tour.insert(bestPos+1, v)
      unvisited.remove(v)
   cost ← 0
   for p = 1 to length(tour)-1:
      cost ← cost + D[tour[p], tour[p+1]]
   return tour, cost
```

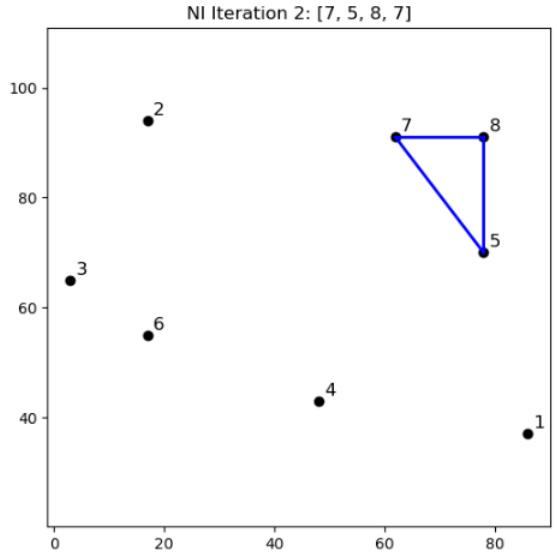
ITERATIONS

NI start (7,8): final tour [7, 2, 3, 6, 4, 1, 5, 8, 7] cost 235

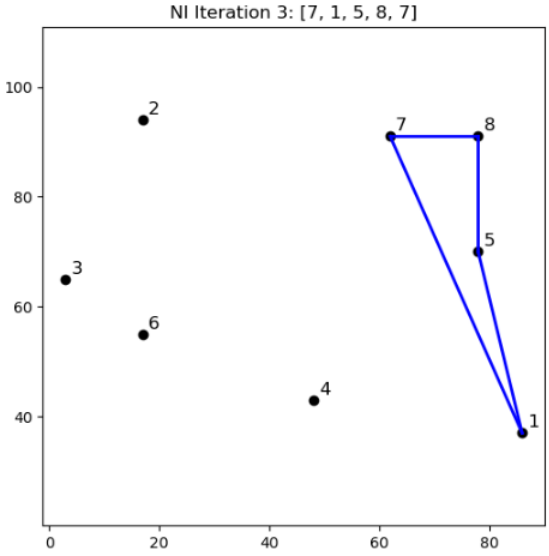


Step 1: chosen node 5 (dist to tour 21)

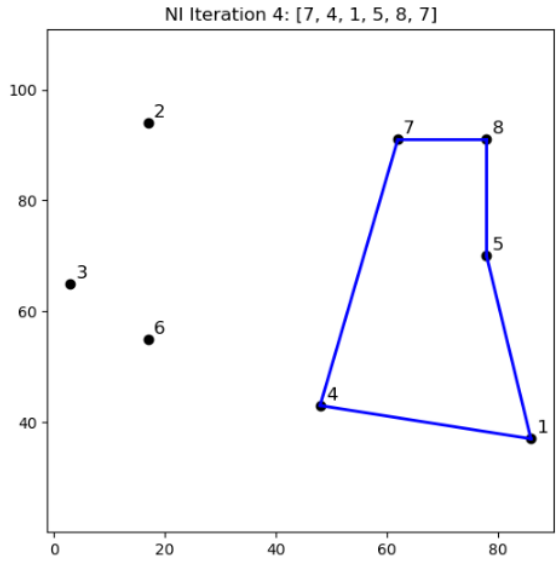
deltas for insertion positions: (pos=0, i=7, j=8, delta=31), (pos=1, i=8, j=7, delta=31)
chosen insertion between (7, 8), delta=31
tour after: [7, 5, 8, 7], cost after: 63



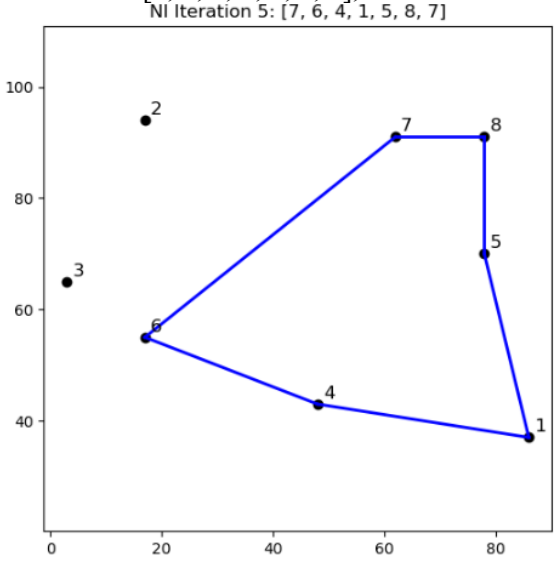
Step 2: chosen node 1 (dist to tour 33)
deltas for insertion positions: (pos=0, i=7, j=5, delta=66), (pos=1, i=5, j=8, delta=66), (pos=2, i=8, j=7, delta=97)
chosen insertion between (7, 5), delta=66
tour after: [7, 1, 5, 8, 7], cost after: 129



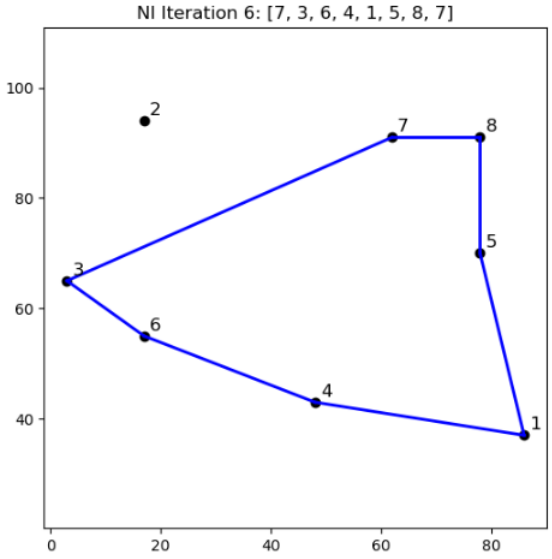
Step 3: chosen node 4 (dist to tour 38)
deltas for insertion positions: (pos=0, i=7, j=1, delta=29), (pos=1, i=1, j=5, delta=45), (pos=2, i=5, j=8, delta=75), (pos=3, i=8, j=7, delta=90)
chosen insertion between (7, 1), delta=29
tour after: [7, 4, 1, 5, 8, 7], cost after: 158



Step 4: chosen node 6 (dist to tour 33)
deltas for insertion positions: (pos=0, i=7, j=4, delta=40), (pos=1, i=4, j=1, delta=66), (pos=2, i=1, j=5, delta=100), (pos=3, i=5, j=8, delta=111), (pos=4, i=8, j=7, delta=111)
chosen insertion between (7, 4), delta=40
tour after: [7, 6, 4, 1, 5, 8, 7], cost after: 198



Step 5: chosen node 3 (dist to tour 17)
deltas for insertion positions: (pos=0, i=7, j=6, delta=24), (pos=1, i=6, j=4, delta=34), (pos=2, i=4, j=1, delta=99), (pos=3, i=1, j=5, delta=129), (pos=4, i=5, j=8, delta=133), (pos=5, i=8, j=7, delta=127)
chosen insertion between (7, 6), delta=24
tour after: [7, 3, 6, 4, 1, 5, 8, 7], cost after: 222



Step 6: chosen node 2 (dist to tour 32)
  deltas for insertion positions: (pos=0, i=7, j=3, delta=13), (pos=1, i=3, j=6, delta=54), (pos=2, i=6, j=4, delta=65), (pos=3, i=4, j=1, delta=110), (pos=4, i=1, j=5, delta=121), (pos=5, i=5, j=8, delta=105), (pos=6, i=8, j=7, delta=90)
  chosen insertion between (7, 3), delta=13
  tour after: [7, 2, 3, 6, 4, 1, 5, 8, 7], cost after: 235