Methodology

Section 1: getInput()

Section 2: initStructures()

Section 3: readCSVFile()

Section 4: sortList(in)

Section 5: printMPV()

Section 6: printLPV()

Section 7: matrixToString()

Section 8: listToString()

Section 9: getPath()

Section 10: writeCSVFile(out, filePath)

Worst Case Cost

Section 0: main(args)

14 units

Section 1: getInput()

10 units

Section 2: initStructures()

4n2 + 7n + 6 units

Section 3: readCSVFile()

17n + 10 units

Section 4: sortList(in)

6n2 + 7n + 4 units

Section 5: printMPV()

7n2 + 17n + 7 units

Section 6: printLPV()

7n2 + 17n + 8 units

Section 7: matrixToString()

6n2 + 15n + 10 units

Section 8: listToString()

8n2 + 9n + 6 units

Section 9: getPath()

8n + 7 units

Section 10: writeCSVFile(out, filePath)

7 units

Total:

38n2 + 97n + 89 units, this program is O(n2) for C = 184 and N0 = 1 because 38n2 + 97n + 89 is always less than or equal to 184n2

Output

Number of Vertices: 256

Input CSV File (full path including extensions, use '/' for separator): C:/Users/Inderpreet/Desktop/CPSC331/GraphEdges\_256x3000.csv

Number of neighbors for MPV: 35

MPV, Neighbors

21,11,16,22,24,38,49,56,60,79,81,83,88,95,98,121,125,128,135,136,143,146,149,155,162,167,177,184,186,192,202,211,215,219,232,253

Number of neighbors for LPV: 12

LPV, Neighbors

29,15,67,68,71,88,116,144,163,173,193,206,240

118,1,18,20,72,82,100,117,132,173,177,232,235

Analysis Questions

1. After running the program the AdjacencyMatrix.csv file is 129 kb while the AdjacencyList.csv is 24 kb, this is because the list only contains the ‘name’ of the connected vertices but doesn’t contain any extra data like the matrix file does.
2. In my implementation the list likely takes up more space than the matrix does because the matrix only takes n2 bytes (it is a 2d byte array) while the list is an array of size n holding n LinkedLists which each need memory for the data and links.