Assignment 4 Report

Rostyslav Vyuha - 8648733
Sein Izumita - 8682570

The chosen data structure for this was an edge weighted directed graph. The graph includes 2 parameters; the number of vertices, the number of edges. There was also another class called 'DirectedEdge' which had the starting vertex, the destination vertex, and the weight of the edge. Using this graph and an adjacency matrix, we were able to read and create all of the connections from every station.

The algorithm used in part 2 was Dijkstra's shortest path. It uses a binary heap for the implementation. The algorithm takes 2 parameters; a edge weighted graph, and a source vertex. Runtime is O(ElogV), where E is the number of edges and V is the number of vertices. The algorithm will return the shortest path from a source station and a destination station and the time it will take to take that path.

Example outputs for question 2:
i)
```
java ParisMetro 20
20
129
311
7
290
136
68
283
146
247
357
278
183
```

ii)
```
java ParisMetro 20 79
20->129 48
129->311 49
311->314 90
314->343 98
343->32 54
32->303 41
```

```
303->298 46
298->225 75
225->177 88
177->79 41
Total time = 630.0s.
```

iii)

```
java ParisMetro 20 79 314
20->129 48
129->311 49
311->7 79
7->290 39
290->136 73
136->68 85
68->67 90
67->173 51
173->227 34
227->356 38
356->77 57
77->79 90
Total time = 733.0s
```

Case: If it is impossible to switch lines

```
java ParisMetro 1 5 235
1->235 99999 (Impossible!)
235->284 99999
284->285 90
285->305 61
305->229 43
229->312 64
312->313 90
313->140 47
140->122 103
122->125 59
125->124 90
124->14 46
14->13 90
13->5 99999 (Impossible!)
300780.0
```