Boston University

METCS 526: Final Project

By: Aidan Duffy

Section I, Pseudocode:

**Algorithm One (n, G):**

Input: some starting node *n* and an undirected, weighted, connected graph G

Output: the node *v* that, among all neighboring nodes of *n*, has the smallest return value of *dd(v)*, or *direct distance* to the destination node Z*.*

Minimum\_distance = Integer.MAX\_VALUE

Best\_node = null

**For** each node *v* which shares an edge with node *n* **do**:

**if** *dd(v)* < minimum\_distance **then:**

minimum\_distance = *dd(v)*

best\_node = v

return best\_node

**Algorithm Two(n, G):**

Input: some starting node *n* and an undirected, weighted, connected graph G

Output: the node *v* that, among all neighboring nodes of *n*, has the smallest return value of *dd(v)* + *w(n,v)*, or *direct distance* to the destination node Zplus the weight of the edge shared by node *n* and node *v.*

Minimum\_distance = Integer.MAX\_VALUE

Best\_node = null

**For** each node *v* which shares an edge with node *n* **do**:

**if** *w(n,v) + dd(v)* < minimum\_distance **then:**

minimum\_distance = *dd(v) + w(n,v)*

best\_node = v

return best\_node

Both of these algorithms will run iteratively until the current node *n* is Z.

Section II, Data Structures Used:

I created two new classes, a Node and a Graph.

For the Node, it contained:

1. a character for its letter value (‘A’, ‘B’,…’Z’) called identifier
2. a HashMap where the key is the Node that neighbors this Node and the value is the weight of their shared edge, called neighbors
3. an integer called direct\_distance which is this Node’s direct distance to the destination Node Z.

For the Graph, it contained:

1. an ArrayList of all of the Nodes in this graph called nodes.
2. a Node called position which is the current position of our graph traversal