Kristopher J Sewell CE2336.002 Pseudocode Project 5
kjs170430

Model

Graph-Weighted. Construct w/
-planet list of vertices
-edge list of weightededges
Pilot-LinkedList. Construct w/ default. Populate w/ (use java base LinkedList)
-Pilot pilot = new Pilot()
-list.add(pilot)
-pilot.getVertices().add(int)
Pilot- members{double length, bool valid, list<int> vertices, String name, bool sortAlpha}
functions{ String toString(),
int compareTo(Pilot p),
double getLength(int start, int end),
void calculateLength(),
getters,setters}

View

InputFile – for a given file name, open, and return a stream of Strings, from file lines.
-verifies that filepath is valid
-can be opened and read from.
OutputFile – for a given file name, open, and write(append) data to the file.
-First open creates fresh file on init of class.
-controller can write to file at anypoint.

Controller – Single Class linear logic

-Create graph, pilot-LinkedList, inputfile, outputfile.
-Open first input file “galaxy.txt” to populate graph.
Generates a stream.forEach(Controller::populateEdges); //“str → populateEdges(str)”
void populateEdges(String toSplit)
-for each string.split(“ “). split[0] is vertex name
-each split after is an edge- vertName,weight
create: getVert(split[0]).addedge(split[0],split[1,3,5],split[2,4,6]);
i % 2 == 0 && i !=0 , i % 2 == 1 && i != 0 while i < split.length()
-edges can be added without vertex being created initially.
-Check for edges with no vertice references. (shouldn’t be needed. As file is “valid”)

-Open second input file “pilot_routes.txt” to populate pilot-list.
-pilot add using existing name logic from prior project.
-change tuple to simple integer to represent vertices.
-Area logic is no longer needed will not be reused.
-for each pilot try to calculate shortest path & length.
double getLength(int start, int end)
-get ShortestPathTree spt for the first vertice(start).
-Add the length for spt.getCost(end)
if it isnt infinite to length return double.
If it is infinite path is not valid setValid(false) & return double.infinite.

void calculateLength()
while(valid && pilot.vertices.hasNext())
length += getLength(current, current.getNext());
-Repeat cost calculate for each vertice in pilot. Summing the cost with length each time.

-store both validity and length in pilot object

-stable sort (mergesort,block,cube,insetion) alphabeticly, then by length.
This will give a full list by length then alpha sorted.
Default java list calls for a comparator but has a stable mergesort by default.
-for each in pilot list. pilot.toString into outputfile if pilot.isValidPath()
-for each in pilot list. Pilot.toString into outputfile if not pilot.isValidPath()

Main will contain creation of controller, filenames, cleanup(if any) and exception handling.