Data Structures and Algorithm Final Assignment

Rivin Pathirage Curtin ID: - 21198889

❖ Design of the program

We will discuss the overall structure of the program (How it all comes together) ADT's used (Abstract Data Structures) description of each class.

The program consists of 7 class files 4 of which has inner classes implemented which is illustrated in the below UML diagram. Including all the inner classes (There are 6 inner classes), the program consists of 13 classes in the program.

The program is brought together in a single main method which also tests the entire functionality, namely "TestHarness." Some of the functionality of the classes or some methods are not directly tested in the TestHarness even though without their proper functioning the program would crash. For an example the "DSAQueue" class that I implemented in for the practical 3 is used for the queue necessary for the breadth first search which is tested in the TestHarness(The breadth first search is tested directly in the TestHarness, but not tested directly in the TestHarness) wouldn't function without the proper functioning of the DSAQueue class.

The DSAGraph class consist of DSAGraphNode and DSAGraphEdge classes. The DSAGraphNode class implements the graph node (Vertices) and DSAGraphEdge class implements the edge of the graph ADT (Abstract Data Type) and finally the DSAGraph class utilizes the DSAGraphNode and DSAGraphEdge classes and implements the graph.

The Graph nodes and the edges of the graph is connected as a Linked List with the help of the DSALinkedList class (Implemented in the Practical 4). As mentioned above the breadth first search method utilizes DSAQueue class for the required queue. Similarly, Depth First Search utilizes the DSAStack class for the required stack ADT.

Data Structures and Algorithm Final Assignment

Rivin Pathirage Curtin ID: - 21198889

The DSALinkedList class utilizes the DSALinkedListNode to implement the nodes of the Linked List and uses the DSALinkedListIterator class to iterate though the Linked List when necessary.

The DSAHashTable class is used to store and quickly retrieve (Implemented for the practicle 7) the details of the vertices (Area and it's attributes such as temperature, humidity etc). DSAHashTable has the DSAHashEntry inner class to deal with the inserting of data to the hash node.

The DSAHeap is used for sorting areas (vertices) in the risk high to low order (max heap, DSAHeap class implements a max heap) and also retrieve that data. DSAHeap class has an inner class namely DSAHeapEntry for getting the data into the heap tree.

Finally, The TestHarness as mentioned above tests the program and reads the input text files (location.txt and UAVData.txt). For different functionalities of the program an interactive menu has been implemented.

❖ Testing Methodology and Results

The testing for this program is done simply using the main method (in the TestHarness class). All the functionalities have been and can be tested in this main method. For an example the adding of edges and simply tested and executed by using the "addVertex" and "addEdge" methods of DSAGraph class.

Result of the testing shows all implemented functions and ADT's are working successfully. Some improvement and polishing can be done for some methods, which we will be discussing after the UML diagram.

Data Structures and Algorithm Final Assignment

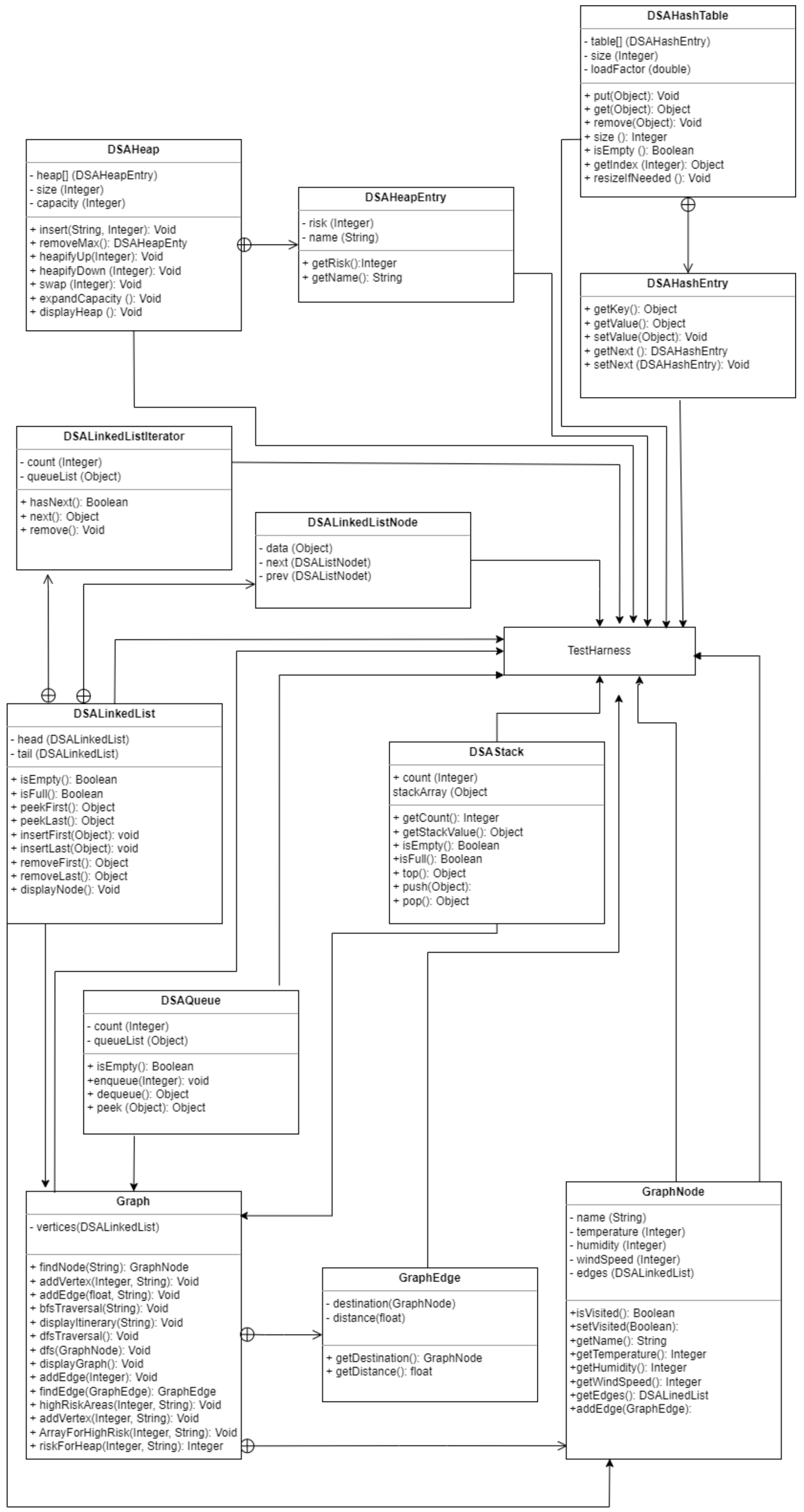
Rivin Pathirage Curtin ID: - 21198889

Feature / Function	Requirements	Design	Test
Program features			
Display graph	Displaying the graph	displayGraph() method is tested in the TestHarness.java class	Success
Add a node to the graph	Adding a Vertex to the graph	addVertex() method is tested in the TestHarness.java class	Success
Searching for a vertex	Searching a vertex of the graph	findNode() method is tested in addEdge(), bfsTraversal(), displayItinerary() methods in Graph.java class	Success
Breath First Search	Traversing the graph for the Breadth first Search order	bfsTraversal() method is tested in the TestHarness.java class	Success
Depth First Search	Traversing the graph for the depth first Search order	dfs() and it's wrapper dfsTraversal() method is tested in the TestHarness.java class	Success
Providing the itinerary	Finding and displaying the shortest path between two vertices	displayItinarary() method is tested in the TestHarness.java class	Success
Hashing the data	Putting the details of vertices in the hash table for quick retrieval	DSAHashTable.java class is tested in the TestHarness.java class	Success
Using a Heap tree	Putting the details of vertices to a max Heap Tree	DSAHeap.java class is tested in the TestHarness.java class	Success
File I/O functions			
Read File	Data must be read in by location.txt file and UAVData.txt file	Each feature when called reads the files and populates the objects and variables in TestHarness.java class	Success

Data Structures and Algorithm Final Assignment

Rivin Pathirage Curtin ID: - 21198889

❖ UML Diagram



Data Structures and Algorithm Final Assignment

Rivin Pathirage Curtin ID: - 21198889

❖ Possible Improvements

- For the Itinerary methods input logic can be improved
- User inputting bfsTraversal method kept getting error. If overcame that would be an improvement
- User Input for getting a hash value from a key could be improved

These issue could've been improved if not to lack of time due to other two final assignments, the overlapping of final assignment time durations, the practical and vivas. I hope to do better next time.