ECE 250 Project 2: Design Document

Akbar Zafar: 20832368

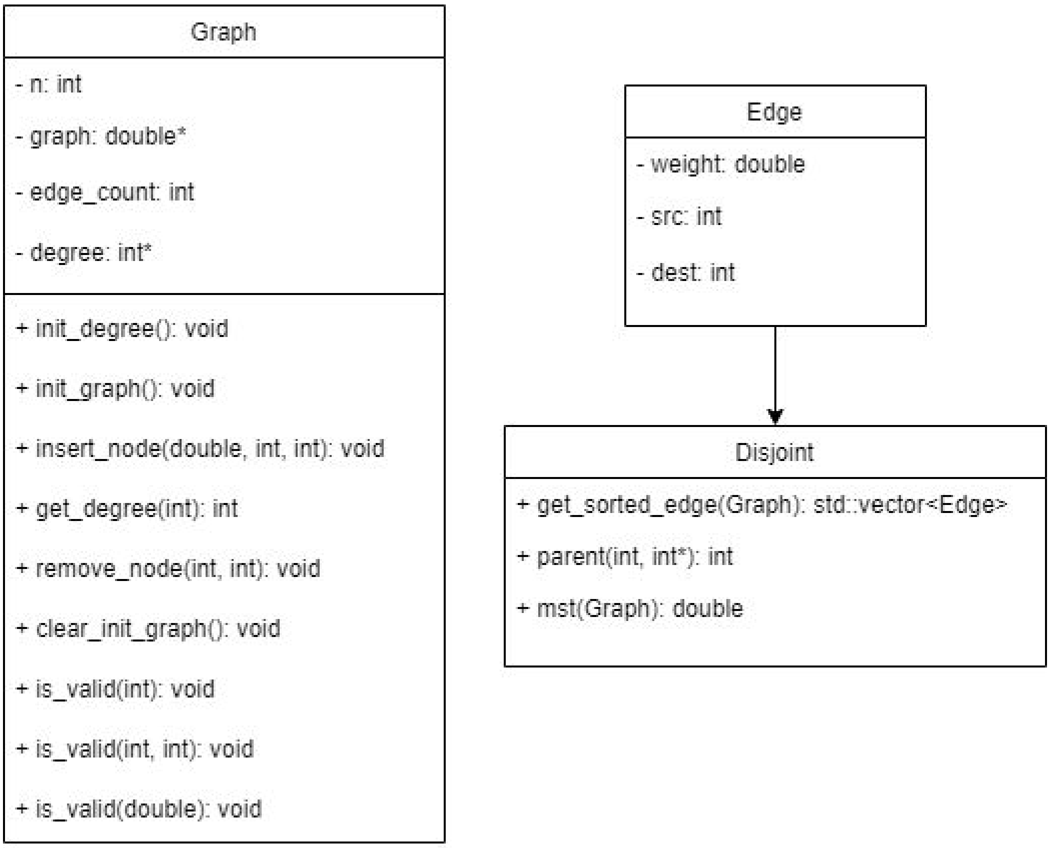
Overview of Classes

The classes used were a Graph class, an Edge class, and a Disjoint class.

The Graph class represents a general graph created by inserted edges. This class also allows manipulation on the graph such as deletion, insertion, and clearing graph. This class also stores data for edge count and degree of a node.

The Disjoint class handles all data related to calculating the MST.

UML Class Diagram



Details on Design Decisions

The constructor for the Graph class takes the size of the graph. With this, a graph represented by an adjacency matrix using an array of size n^2 is declared and initialized to zeros to indicate a graph node with no connection. Also, an array of size n is declared to keep track of the degree of each node.

The destructor for the Graph class deallocates any memory in the graph and the vertex array.

The QuadTree destructor calls a recursive function to deallocate all nodes and set to nullptr. In addition, it sets the root to nullptr and resets the count for the size of the quadtree to 0.

Test Cases

Aside from the test cases provided, I came up with several test cases trying cover any edge cases such as creating a graph with a negative value. Also, I tried multiple ways to break my code.

Performance Evaluation

All calculations are based off assuming we have a balanced tree

Graph:

Insert: The time to insert is O(1).

Delete: The time to delete an edge between nodes is O(1).

Degree: the time to find the degree of a vertex is (1).

edge_count: The time to get the number of edges in a graph is O(1).

Clear: The time to clear all nodes in a graph is $O(n^2)$ since we have to go through each element in the array.

MST: The time required to calculate the mst is $O(n^2)$ since the mst first needs to build a vector using the graph of size n^2 and that is the longest operation.