

lab 23 Graphs via Adjacency Lists

Instructions: In this lab implement a Graph with an adjacency list.

Implement the following class:

```
1 #ifndef GRAPHAL_H
2 #define GRAPHAL_H
3
4 /* This class represents a weighted directed graph via an adjacency list.
5  * Vertices are given an index, starting from 0 and ascending
6  * Class W : W represent the weight that can be associated with an edge.
7  * We will not weight the vertices.
8  */
9
10 template<class W>
11 class GraphAL {
12     private:
13         /* You fill out. */
14     public:
15         /* Initialize an empty graph. */
16         GraphAL();
17
18         /* Initialize the Graph with a fixed number of vertices. */
19         GraphAL(const int vertices);
20
21         /* Destructor shall free up memory */
22         ~GraphAL();
23
24         /* Adds amt vertices to the graph. */
25         void addVertices(int amt);
26
27         /* Removes a vertex.
28          * return wheter sucessful or not
29          */
30         bool removeVertex(int idx);
31
32         /* Adds an edge with weight W to the graph. */
33         bool addEdge(const int start, const int end, const W &weight);
34
35         /*
36          * Remove edge from graph.
37          */
38         bool removeEdge(const int start, const int end);
39
40         void depthFirstTraversal(void (*visit)(const int node));
41         void breadthFirstTraversal(void (*visit)(const int node));
```

```

42
43     /*
44     * Return adjacent weight from start to end (or -1 if they are
45     * not adjacent.
46     */
47     W adjacent(const int start, const int end);
48
49     /* Returns the TOTAL weight of the minimum spanning tree with the
50     * given starting node.
51     * You must use Prim's MST.
52     */
53     W prims(const int start);
54
55     /* Print out the Graph */
56     void print() const;
57
58 };
59
60 #include "graphal.cpp"
61
62 #endif

```

Write some test cases:

Create some test cases, using cxxtestgen, that you believe would cover all aspects of your code.

Memory Management:

Now that are using new, we must ensure that there is a corresponding delete to free the memory. Ensure there are no memory leaks in your code! Please run Valgrind on your tests to ensure no memory leaks!

How to turn in:

Turn in via GitHub. Ensure the file(s) are in your directory and then:

- \$ git add <files>
- \$ git commit
- \$ git push

Due Date: April 17, 2019 2359

Teamwork: No teamwork, your work must be your own.