Bozga Rareș Ionuț

Lab Assignment 1 – Graph Documentation

In order to use the Graph utility, please import the following:

#include “graph.h”

Main functionality

You will probably be interested in creating a graph and then inserting data into it manually, or reading the data from a file.

Here are 2 examples of code which do the former and the latter.

Example 1:

```
#include “graph.h”
#include <vector.h>

using namespace std;

int main() {
	Graph g;
	for (int i = 0; i < 10; i++) {
		g.addVertex(i); // add 10 vertices
	}

	g.addEdge(1, 2, 10); // add edge 1 -> 2 with cost 10
	g.addEdge(2, 3, 15); // add edge 2 -> 3 with cost 15
	g.addEdge(4, 4, 20); // add edge 4 -> 4 with cost 20

	g.print(); // print graph

	return 0;
}
```

Example 2:

```
#include “graph.h”
#include <vector.h>

using namespace std;

int main() {
	Graph g;
	g.fromFile(“some_graph.txt”); // read from file

	g.print(); // print graph

	return 0;
}
```

- **addVertex(int v)**
  Adds a vertex to the graph.
  Returns true if the vertex was added, and false if it already exists.

- **removeVertex(int v)**
  Removes a vertex from the graph, and destroys edges linked to the vertex.
  Returns true if the vertex was removed, and false if it was not part of the graph.

- **isVertex(int v)**
  Returns true if v is a vertex in the graph, and false otherwise.

- **addEdge(int from, int to, int cost)**
  Adds an edge with a cost to the graph.
  Returns true if the vertex was added, and false if it already exists.

- **removeEdge(int from, int to)**
  Removes an edge from the graph.
  Returns true if the vertex was removed, and false if it was not part of the graph.

- **isEdge(int from, int to)**
  Returns true if this is an edge in the graph, and false otherwise.

- **getCost(int from, int to)**
  Returns the cost of an edge in the graph, or 0 if the edge doesn't exist.

- **getVerticesOut(int from)**
  Returns a vector containing all vertices which come out of the provided vertex.

- **getVerticesIn(int to)**
  Returns a vector containing all vertices which go into the provided vertex.

- **fromFile(string input)**
  Loads a graph from textual file content.
  Provide on the first line the number of vertices and edges, then on the following lines, each edge including the cost.
  Returns true if the file was loaded, and false otherwise.

- **toFile(string input)**
  Saves a graph to a textual file. Will overwrite old contents.
  Returns true if the file was saved, and false otherwise.

- **print()**
  Prints the graph's contents to the standard console output stream.

Iterating the graph.

To iterate the graph, use the GraphIterator provided in the Graph class.
The iterator works like most other implementations, so it should be easy to use.

See the following example, which prints the in-bound and out-bound nodes of all the vertices in a graph.

```
GraphIterator iter = graph.iterator();
iter.first();

while (iter.valid()) {
    int vertex = iter.getCurrent();

    std::cout << "IN: ";
    for (int in : graph.getVerticesIn(vertex)) {
       cout << in << " ";
    } cout << endl;

    std::cout << "OUT: ";
    for (int out : graph.getVerticesOut(vertex)) {
      cout << out << " ";
    } cout << endl;

      iter.next();
}
```

- **iterator()**
  Returns the GraphIterator for a Graph.

- **first()**
  Sets the iterator's current element to the first one.

- **valid()**
  Returns true if the current element is valid, and false otherwise.

- **next()**
  Moves to the next element in the iterator, if the current element is valid.

- **getCurrent()**
  Returns the current element in the iterator.