# Lab09 Report

Partners: Keerthi & Grayson

*****Special Instructions for running main.cpp*****

**To run it as an executable, in the terminal type build/runMain to start the program. Moves.txt is already in the build folder so you can access then in the UI of the program and don't need extra commands to run the exe.**

**Initialize graph via file:**

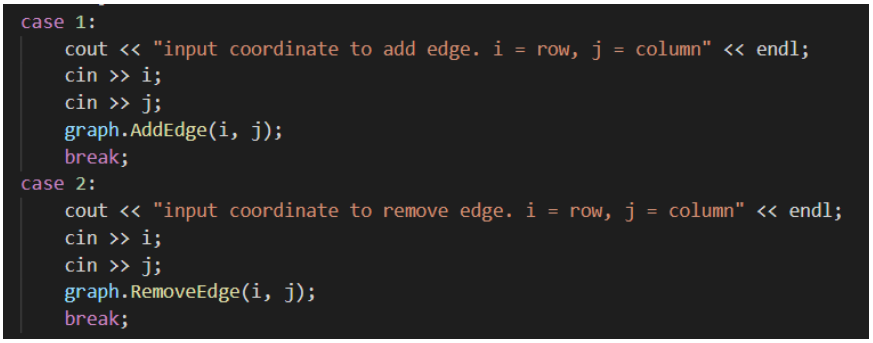
build/runMain
N:
3
M:
3
Do you have a file for the moves you would like to place? Y or N
Y
<graph will be initialized>

To run and make your own graph **(no file)** just type build/runMain and begin.

For main.cpp, no matter what (file or no file) you need to input the graph size (2d vector size) as the first step. If you are using the moves.txt in the zip file, type 3 for N and 3 for M to initialize the size of the graph for the 2d vector in the file.

For using AddEdge and RemoveEdge, for the row and column number just type the number and hit enter. For example, if you want i=1 j = 2, type in the terminal:

1
2
Since the input for those functions is set up like this:

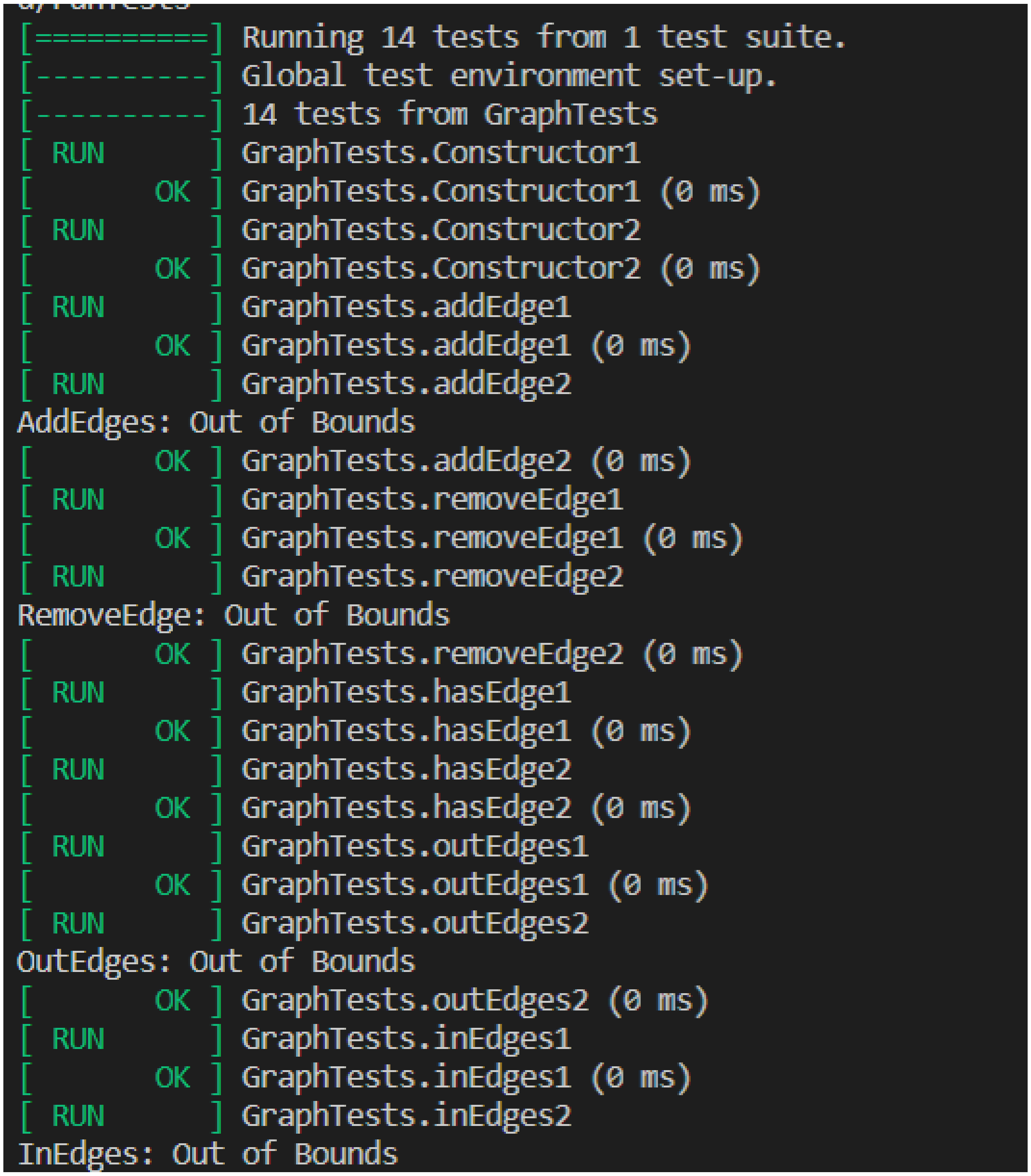


```
case 1:
    cout << "input coordinate to add edge. i = row, j = column" << endl;
    cin >> i;
    cin >> j;
    graph.AddEdge(i, j);
    break;
case 2:
    cout << "input coordinate to remove edge. i = row, j = column" << endl;
    cin >> i;
    cin >> j;
    graph.RemoveEdge(i, j);
    break;
```

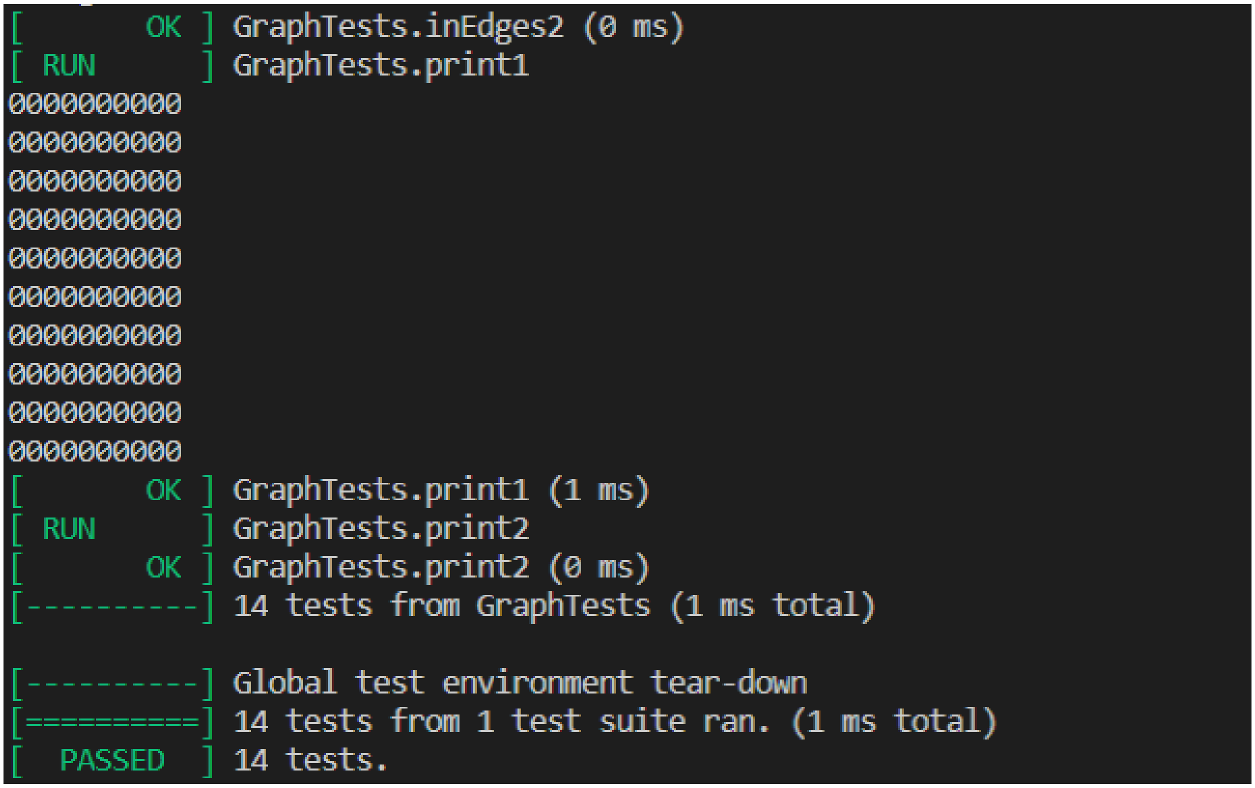
### Design for Internal Structure of Graph

In the adjacency matrix we created 1s were used as edges and 0s were the default values without any edges.  Adding an edge took coordinate values and added a 1 to the associated position within the matrix. Removing an edge function essentially the same except that it removed an edge at the associated position and replaced it with a zero.  The function to check for an edge took a set of coordinates and checked the associated position within the matrix for a one and returned true if there was, otherwise false. The outEdges function returned the columns within a row that contained edges and the inEdges function returned the rows within a column that had edges.

### Proof of Positive and Negative Test cases passing



```
[==========] Running 14 tests from 1 test suite.
[----------] Global test environment set-up.
[----------] 14 tests from GraphTests
[ RUN      ] GraphTests.Constructor1
[       OK ] GraphTests.Constructor1 (0 ms)
[ RUN      ] GraphTests.Constructor2
[       OK ] GraphTests.Constructor2 (0 ms)
[ RUN      ] GraphTests.addEdge1
[       OK ] GraphTests.addEdge1 (0 ms)
[ RUN      ] GraphTests.addEdge2
AddEdges: Out of Bounds
[       OK ] GraphTests.addEdge2 (0 ms)
[ RUN      ] GraphTests.removeEdge1
[       OK ] GraphTests.removeEdge1 (0 ms)
[ RUN      ] GraphTests.removeEdge2
RemoveEdge: Out of Bounds
[       OK ] GraphTests.removeEdge2 (0 ms)
[ RUN      ] GraphTests.hasEdge1
[       OK ] GraphTests.hasEdge1 (0 ms)
[ RUN      ] GraphTests.hasEdge2
[       OK ] GraphTests.hasEdge2 (0 ms)
[ RUN      ] GraphTests.outEdges1
[       OK ] GraphTests.outEdges1 (0 ms)
[ RUN      ] GraphTests.outEdges2
OutEdges: Out of Bounds
[       OK ] GraphTests.outEdges2 (0 ms)
[ RUN      ] GraphTests.inEdges1
[       OK ] GraphTests.inEdges1 (0 ms)
[ RUN      ] GraphTests.inEdges2
InEdges: Out of Bounds
```



```
[       OK ] GraphTests.inEdges2 (0 ms)
[ RUN      ] GraphTests.print1
0000000000
0000000000
0000000000
0000000000
0000000000
0000000000
0000000000
0000000000
0000000000
0000000000
[       OK ] GraphTests.print1 (1 ms)
[ RUN      ] GraphTests.print2
[       OK ] GraphTests.print2 (0 ms)
[----------] 14 tests from GraphTests (1 ms total)

[----------] Global test environment tear-down
[==========] 14 tests from 1 test suite ran. (1 ms total)
[  PASSED  ] 14 tests.
```