garaph

0.1.0

Generated by Doxygen 1.8.17

Chapter 1

Class Index

1.1 Class List

Here are the classes, structs, unions and interfaces with brief descriptions:

Chapter 2

File Index

2.1 File List

Here is a list of all files with brief descriptions:

Chapter 3

Class Documentation

3.1 graph Class Reference

Public Member Functions

- graph (int input)
- void addEdge (int O, int E)
- void removeEdge (int O, int E)
- bool hasEdge (int O, int j)
- void displayM ()
- void outEdges (int O, vector< int > &edges)
- void inEdges (int O, vector< int > &edges)
- int nVertices ()
- void dfs2 (graph &g, int r)
- void bfs (graph &g, int r)

Public Attributes

- int B
- int TheMatrix [7][7] = {0}

3.1.1 Detailed Description

Definition at line 13 of file main.cpp.

3.1.2 Constructor & Destructor Documentation

3.1.2.1 graph()

```
graph::graph (
              int input )  [inline]
```

Definition at line 18 of file main.cpp.

```
18                    {
19        B = input;
20    }
```

3.1.3 Member Function Documentation

3.1.3.1 addEdge()

```
void graph::addEdge (
              int O,
              int E )  [inline]
```

Definition at line 22 of file main.cpp.

```
22                           {
23    TheMatrix[O][E] = true;
24 }
```

3.1.3.2 bfs()

```
void graph::bfs (
              graph & g,
              int r )  [inline]
```

Definition at line 73 of file main.cpp.

```
73                          {
74    bool *seen = new bool[g.nVertices()];
75    vector<int> q;
76    q.push_back(r);
77    seen[r] = true;
78    while (q.size() > 0) {
79        int O = q.back();
80        cout « endl « O « " > " « "This is BFS" « endl;
81        q.pop_back();
82        vector<int> edges;
83        g.outEdges(O, edges);
84        for (int k = 0; k < edges.size(); k++) {
85            int E = edges[k];
86            if (!seen[E]) {
87                q.push_back(E);
88                seen[E] = true;
89    }
90 }
91 }
92 delete[] seen;
93 }
```

3.1.3.3 dfs2()

```
void graph::dfs2 (
               graph & g,
               int r )  [inline]
```

Definition at line 53 of file main.cpp.

```
53                               {
54     bool *f = new bool[g.nVertices()];
55     vector<int> s;
56     s.push_back(r);
57     while (s.size() > 0) {
58         int O = s.back();
59         cout « endl « " > " « "This is DFS" « endl;
60         s.pop_back();
61         if (f[O] == *f) {
62             f[O] = f;
63             vector<int> edges;
64             g.outEdges(O, edges);
65             for (int k = 0; k < edges.size(); k++)
66                 s.push_back(edges[k]);
67     }
68     }
69     delete[] f;
70 }
```

3.1.3.4 displayM()

```
void graph::displayM ( )  [inline]
```

Definition at line 31 of file main.cpp.

```
31                  {
32     for(int O = 0; O < B; O++) {
33         for(int E = 0; E < B; E++) {
34             cout « TheMatrix[O][E];
35         }
36         cout « endl;
37     }
38 }
```

3.1.3.5 hasEdge()

```
bool graph::hasEdge (
               int O,
               int j )  [inline]
```

Definition at line 28 of file main.cpp.

```
28                               {
29 return TheMatrix[O][j];
30 }
```

3.1.3.6 inEdges()

```
void graph::inEdges (
               int O,
               vector< int > & edges )  [inline]
```

Definition at line 44 of file main.cpp.

```
44                                            {
45     for (int E = 0; E < B; E++)
46         if (TheMatrix[O][E]) edges.push_back(E);
47 }
```

3.1.3.7 nVertices()

```
int graph::nVertices ( )  [inline]
```

Definition at line 48 of file main.cpp.

```
48                {
49    return B * B;
50 }
```

3.1.3.8 outEdges()

```
void graph::outEdges (
             int O,
             vector< int > & edges )  [inline]
```

Definition at line 40 of file main.cpp.

```
40                                          {
41    for (int E = 0; E < B; E++)
42        if (TheMatrix[E][O]) edges.push_back(E);
43 }
```

3.1.3.9 removeEdge()

```
void graph::removeEdge (
             int O,
             int E )  [inline]
```

Definition at line 25 of file main.cpp.

```
25                                {
26    TheMatrix[O][E] = false;
27 }
```

3.1.4 Member Data Documentation

3.1.4.1 B

```
int graph::B
```

Definition at line 15 of file main.cpp.

3.1.4.2 TheMatrix

```
int graph::TheMatrix[7][7] = {0}
```

Definition at line 16 of file main.cpp.

The documentation for this class was generated from the following file:

- /home/bona/CPTR227/Graphs/src/main.cpp

Chapter 4

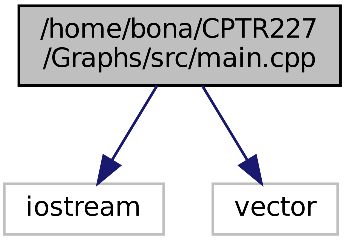
File Documentation

4.1 /home/bona/CPTR227/Graphs/src/main.cpp File Reference

This is a test of CMake, doxygen, and GitHub.

```
#include <iostream>
#include <vector>
```

Include dependency graph for main.cpp:



Classes

- class graph

Functions

- int main (int, char ∗∗)

4.1.1 Detailed Description

This is a test of CMake, doxygen, and GitHub.

This is the long brief at the top of main.cpp.

Author

Bona Tufa

Date

4/7/2021

4.1.2 Function Documentation

4.1.2.1 main()

```
int main (
             int ,
             char **  )
```

Definition at line 95 of file main.cpp.

```
95                      {
96    graph value(7);
97    value.addEdge(3,0);
98    value.addEdge(6,3);
99    value.displayM();
100    value.bfs(value, 1);
101    value.removeEdge(4,4);
102    value.displayM();
103    value.addEdge(1,2);
104    value.addEdge(2,4);
105    value.hasEdge(1,2);
106    value.dfs2(value, 3);
107    value.displayM();
108   //value.dfs2(value, 3);
109
110 }
```

Index