

Graph

0.3.0

Generated by Doxygen 1.8.17



<b>1 Class Index</b>	<b>1</b>
1.1 Class List	1
<b>2 File Index</b>	<b>3</b>
2.1 File List	3
<b>3 Class Documentation</b>	<b>5</b>
3.1 Graph Class Reference	5
3.1.1 Detailed Description	5
3.1.2 Constructor & Destructor Documentation	6
3.1.2.1 Graph()	6
3.1.3 Member Function Documentation	6
3.1.3.1 addEdge()	6
3.1.3.2 bfs()	6
3.1.3.3 dfs2()	7
3.1.3.4 displayMatrix()	7
3.1.3.5 hasEdge()	7
3.1.3.6 inEdges()	7
3.1.3.7 nVertices()	8
3.1.3.8 outEdges()	8
3.1.3.9 removeEdge()	8
3.1.4 Member Data Documentation	8
3.1.4.1 Matrix	8
<b>4 File Documentation</b>	<b>9</b>
4.1 /home/addis/graph/src/main.cpp File Reference	9
4.1.1 Detailed Description	10
4.1.2 Function Documentation	10
4.1.2.1 main()	10
<b>Index</b>	<b>11</b>



# Chapter 1

## Class Index

### 1.1 Class List

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

<a href="#">Graph</a> . . . . .	5
---------------------------------	---



## Chapter 2

# File Index

### 2.1 File List

Here is a list of all files with brief descriptions:

<code>/home/addis/graph/src/main.cpp</code>	
This is a graph project . . . . .	9





# Chapter 3

## Class Documentation

### 3.1 Graph Class Reference

#### Public Member Functions

- `Graph` (int input)
- void `addEdge` (int i, int j)
- void `removeEdge` (int i, int j)
- bool `hasEdge` (int i, int j)
- void `displayMatrix` ()
- void `outEdges` (int i, vector< int > &edges)
- void `inEdges` (int i, vector< int > &edges)
- int `nVertices` ()
- void `bfs` (`Graph` &g, int r)
- void `dfs2` (`Graph` &g, int r)

#### Public Attributes

- int `Matrix` [10][10] = { 0 }

#### 3.1.1 Detailed Description

Add two integers (brief)

Adds a and b, two integers (long description)

##### Parameters

<i>a</i>	integer
<i>b</i>	integer

##### Returns

integer sum of a and b

Definition at line 22 of file main.cpp.

### 3.1.2 Constructor & Destructor Documentation

#### 3.1.2.1 Graph()

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

Definition at line 28 of file main.cpp.

```
28     {
29         n = input;
30     }
```

### 3.1.3 Member Function Documentation

#### 3.1.3.1 addEdge()

```
void Graph::addEdge (
    int i,
    int j ) [inline]
```

Definition at line 32 of file main.cpp.

```
32     {
33         Matrix[i][j] = true;
34     }
```

#### 3.1.3.2 bfs()

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

Definition at line 64 of file main.cpp.

```
64     {
65         bool *seen = new bool[g.nVertices()];
66         vector<int> q;
67         q.push_back(r);
68         seen[r] = true;
69         while (q.size() > 0) {
70             int i = q.back();
71             cout << endl << i << " > " << "This is BFS" << endl;
72             q.pop_back();
73             vector<int> edges;
74             g.outEdges(i, edges);
75             for (int k = 0; k < edges.size(); k++) {
76                 int j = edges[k];
77                 if (!seen[j]) {
78                     q.push_back(j);
79                     seen[j] = true;
80                 }
81             }
82         }
83         delete[] seen;
84     }
```

### 3.1.3.3 dfs2()

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

Definition at line 88 of file main.cpp.

```
88     {
89         bool *c = new bool[g.nVertices()];
90         vector<int> s;
91         s.push_back(r);
92         while (s.size() > 0) {
93             int i = s.back();
94             cout << endl << i << " > " << "This is DFS" << endl;
95             s.pop_back();
96             if (c[i] == *c) {
97                 c[i] = c;
98                 vector<int> edges;
99                 g.outEdges(i, edges);
100                 for (int k = 0; k < edges.size(); k++)
101                     s.push_back(edges[k]);
102             }
103         }
104         delete[] c;
105     }
```

### 3.1.3.4 displayMatrix()

```
void Graph::displayMatrix ( ) [inline]
```

Definition at line 42 of file main.cpp.

```
42     {
43         for(int i = 0; i < n; i++) {
44             for(int j = 0; j < n; j++) {
45                 cout << Matrix[i][j];
46             }
47             cout << endl;
48         }
49     }
```

### 3.1.3.5 hasEdge()

```
bool Graph::hasEdge (
    int i,
    int j ) [inline]
```

Definition at line 38 of file main.cpp.

```
38     {
39         return Matrix[i][j];
40     }
```

### 3.1.3.6 inEdges()

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

Definition at line 55 of file main.cpp.

```
55     {
56         for (int j = 0; j < n; j++)
57             if (Matrix[j][i]) edges.push_back(j);
58     }
```

### 3.1.3.7 nVertices()

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

Definition at line 60 of file main.cpp.

```
60     {
61     return n * n;
62 }
```

### 3.1.3.8 outEdges()

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

Definition at line 51 of file main.cpp.

```
51     {
52     for (int j = 0; j < n; j++)
53         if (Matrix[i][j]) edges.push_back(j);
54 }
```

### 3.1.3.9 removeEdge()

```
void Graph::removeEdge (
    int i,
    int j ) [inline]
```

Definition at line 35 of file main.cpp.

```
35     {
36     Matrix[i][j] = false;
37 }
```

## 3.1.4 Member Data Documentation

### 3.1.4.1 Matrix

```
int Graph::Matrix[10][10] = { 0 }
```

Definition at line 26 of file main.cpp.

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

- [/home/addis/graph/src/main.cpp](#)

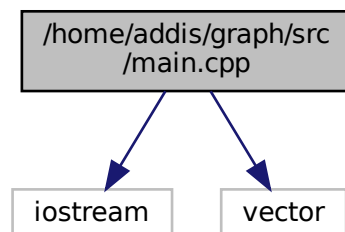
## Chapter 4

# File Documentation

### 4.1 /home/addis/graph/src/main.cpp File Reference

This is a graph project.

```
#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 graph project.

This is the long brief at the top of [main.cpp](#).

#### Author

Addis Bogale

#### Date

4/2/2021

### 4.1.2 Function Documentation

#### 4.1.2.1 main()

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

Definition at line 110 of file main.cpp.

```
110     {
111         Graph value(10);
112         value.addEdge(0,1);
113         value.addEdge(3,3);
114         value.displayMatrix();
115         value.bfs(value, 0);
116         value.removeEdge(3,3);
117         value.displayMatrix();
118
119         value.addEdge(2,3);
120         value.addEdge(1,3);
121
122         value.hasEdge(0,1);
123         value.dfs2(value, 2);
124         value.displayMatrix();
125
126
127     }
```

# Index

/home/addis/graph/src/main.cpp, [9](#)

addEdge  
    Graph, [6](#)

bfs  
    Graph, [6](#)

dfs2  
    Graph, [6](#)

displayMatrix  
    Graph, [7](#)

Graph, [5](#)  
    addEdge, [6](#)  
    bfs, [6](#)  
    dfs2, [6](#)  
    displayMatrix, [7](#)  
    Graph, [6](#)  
    hasEdge, [7](#)  
    inEdges, [7](#)  
    Matrix, [8](#)  
    nVertices, [7](#)  
    outEdges, [8](#)  
    removeEdge, [8](#)

hasEdge  
    Graph, [7](#)

inEdges  
    Graph, [7](#)

main  
    main.cpp, [10](#)

main.cpp  
    main, [10](#)

Matrix  
    Graph, [8](#)

nVertices  
    Graph, [7](#)

outEdges  
    Graph, [8](#)

removeEdge  
    Graph, [8](#)