

Readme

I have implemented the features of the project by making a class for graph named myGraphClass.

GitHub Link: https://github.com/Divy2000/CSE464_2023_dplate164

This is the file structure of the project :-

```
src
|--- main
|   |--- application
|   |   |--- Main.java           - The file runs the command-line
|   |   |--- myGraphClass.java    interface and interact with myGraphClass.
|   |   |--- testGraph.dot        - This class generates and interact with the graph via
|   |   |--- testGraph.dot        Graphviz, this implements the methods for all the 4
|   |   |--- testGraph.dot        features.
|   |--- testGraph.dot          - This file is used in testing for importing the graph
|   |--- testGraph.dot          and do tests of all the 4 features.
|   |--- testGraph.dot          - This is a test graph to run the test code in GitHub
|   |--- testGraph.dot          workflow.
|
|--- test
|   |--- myGraphClass.java      - This file contains the tests for all 4 features, each
|   |--- myGraphClass.java      test-case tests one feature, I have used assert
|   |--- myGraphClass.java      statements to compare the outputs.
|
pom.xml                                         -This file contains the dependencies of the project.
```

Run the **src/main/application/Main.java** file to access the command-line interface.

To run the test cases run **test/myGraphClass.java**

Following are the commands I have used to run my code :-

“src/main/java/testGraph.dot” -> Parses the .dot file

“0” -> To print the graph details

“1” -> To enter the details of the graph to a file

“src/main/java/testGraph.txt” -> File-path for the output

“2” -> To add node to the graph

“node0” -> Label of the node to add

“3” -> To add multiple nodes to the graph

“node1,node2,node3” -> Three nodes to be added to the graph

“4” -> To add an edge to the graph

“white” -> First node of the edge

“node3” -> Second node of the edge

“5” -> To remove a node from the graph

“yellow” -> Label of the node to remove

“6” -> To remove multiple nodes from the graph

“node1,black” -> 2 nodes to remove

“7” -> To remove an edge from the graph

“blue” -> Label of the first node of the edge to remove

“cyan” -> Label of the second node of the edge to remove

“8” -> To output the graph to .dot file

“src/main/java/testGraph_o.dot” -> Path of the output file

“9” -> To output the graph to an image

“src/main/java/testGraph.png” -> Path of the output image file

“Q” -> To exit the program

In addition to this, now we have added feature for graph search :-

“10” -> Perform BFS search or DFS search in the graph

“bfs” -> For Breadth First Search

OR

“dfs” -> For Depth First Search

Running code screenshots and commit links :-

Feature 1

- Parsing a DOT graph file

The screenshot shows the IntelliJ IDEA interface with the project 'CSE464_2023_dplate164' open. The 'Run' tool window is active, showing the command: '/Library/Java/JavaVirtualMachines/adoptopenjdk-8.jdk/Contents/Home/bin/java ...'. Below it, the terminal output shows:

```
Enter the path to the dot file
src/main/testGraph.dot
SLF4J: Failed to load class "org.slf4j.impl.StaticLoggerBinder".
SLF4J: Defaulting to no-operation (NOP) logger implementation
SLF4J: See http://www.slf4j.org/codes.html#StaticLoggerBinder for further details.

Choose one of the following options
0 -> Print graph details
1 -> Output graph details to a text file
2 -> Add a node to the graph
3 -> Add multiple nodes to the graph
4 -> Add an edge to the graph
5 -> Remove a node from the graph
6 -> Remove multiple nodes from the graph
7 -> Remove an edge from the graph
8 -> Output graph to .dot file
9 -> Output graph to an image file
10 -> Perform BFS search or DFS search in the graph
Q -> Exit the software
Enter your choice
0

There are 8 nodes in the graph.
The label of the nodes are :-
red
pink
green
blue
white
yellow
black
cyan
```

- Print the details of the graph

The screenshot shows the IntelliJ IDEA interface with the project 'CSE464_2023_dplate164' open. The 'Run' tool window is active, showing the command: '/Library/Java/JavaVirtualMachines/adoptopenjdk-8.jdk/Contents/Home/bin/java ...'. Below it, the terminal output shows:

```
Enter your choice
0

There are 8 nodes in the graph.
The label of the nodes are :-
red
pink
green
blue
white
yellow
black
cyan

There are 12 edges in the graph.
The edges are :-
green--black
white--pink
white--cyan
cyan--green
yellow--red
white--yellow
pink--blue
yellow--green
pink--red
blue--black
red--black
cyan--blue
```

- Output the details to a file.

The screenshot shows the IntelliJ IDEA interface with the project 'CSE464_2023_dpage164' open. The code editor displays the Main.java file, which contains a main method that reads a graph from a file named 'testGraph.dot'. The terminal window below shows the execution of the program, starting with a menu of options:

```

Choose one of the following options
0 -> Print graph details
1 -> Output graph details to a text file
2 -> Add a node to the graph
3 -> Add multiple nodes to the graph
4 -> Add an edge to the graph
5 -> Remove a node from the graph
6 -> Remove multiple nodes from the graph
7 -> Remove an edge from the graph
8 -> Output graph to .dot file
9 -> Output graph to an image file
10 -> Perform BFS search or DFS search in the graph
Q -> Exit the software
Enter your choice
1
Enter the filepath of the output file:
output.txt
Details about graph are successfully written to the file
  
```

The status bar at the bottom indicates that 'src/main/testGraph.dot' has been copied.

Commit link: https://github.com/Divy2000/CSE464_2023_dpage164/commit/992acecb2f21363762b58f52b81f04d088277959

Feature 2

- Add a node to the graph

The screenshot shows the IntelliJ IDEA interface with the project 'CSE464_2023_dpage164' open. The code editor displays the Main.java file, which contains a main method that reads a graph from a file named 'testGraph.dot'. The terminal window below shows the execution of the program, starting with a menu of options. The user selects option 2 to add a node to the graph:

```

Choose one of the following options
0 -> Print graph details
1 -> Output graph details to a text file
2 -> Add a node to the graph
3 -> Add multiple nodes to the graph
4 -> Add an edge to the graph
5 -> Remove a node from the graph
6 -> Remove multiple nodes from the graph
7 -> Remove an edge from the graph
8 -> Output graph to .dot file
9 -> Output graph to an image file
10 -> Perform BFS search or DFS search in the graph
Q -> Exit the software
Enter your choice
2
Enter the label of the node you want to add:
newNode1
  
```

A tooltip in the bottom right corner says: "Externally added files can be added to Git View Files Always Add Don't Ask Again".

- Add list of nodes to the graph

```

CSE464_2023_ddate164 / .gitignore
Project Commit Pull Requests Bookmarks Structure
Run TODO Problems Terminal Services Build Dependencies
Git pom.xml (CSE464-GraphManager) Maven Notifications

Run: application.Main

Choose one of the following options
0 -> Print graph details
1 -> Output graph details to a text file
2 -> Add a node to the graph
3 -> Add multiple nodes to the graph
4 -> Add an edge to the graph
5 -> Remove a node from the graph
6 -> Remove multiple nodes from the graph
7 -> Remove an edge from the graph
8 -> Output graph to .dot file
9 -> Output graph to an image file
10 -> Perform BFS search or DFS search in the graph
Q -> Exit the software
Enter your choice
3

Enter the labels of node you want to add separated by comma ('a,b,c'):
newNode2, node1beRemoved

```

Externally added files can be added to Git
View Files Always Add Don't Ask Again

Externally added files can be added to Git // View Files // Always Add // Don't Ask Again (5 minutes ago)

- Remove a node from the graph

```

CSE464_2023_ddate164 / .gitignore
Project Commit Pull Requests Bookmarks Structure
Run TODO Problems Terminal Services Build Dependencies
Git pom.xml (CSE464-GraphManager) Maven Notifications

Run: application.Main

Choose one of the following options
0 -> Print graph details
1 -> Output graph details to a text file
2 -> Add a node to the graph
3 -> Add multiple nodes to the graph
4 -> Add an edge to the graph
5 -> Remove a node from the graph
6 -> Remove multiple nodes from the graph
7 -> Remove an edge from the graph
8 -> Output graph to .dot file
9 -> Output graph to an image file
10 -> Perform BFS search or DFS search in the graph
Q -> Exit the software
Enter your choice
5

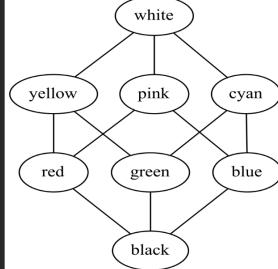
Enter the label of the node you want to remove:
node1beRemoved

```

Externally added files can be added to Git
View Files Always Add Don't Ask Again

Externally added files can be added to Git // View Files // Always Add // Don't Ask Again (6 minutes ago)

- Remove multiple node from the graph



The screenshot shows an IDE interface with a project named "CSE464_2023_dpage164". In the center is a graph visualization window displaying a network of nodes. The nodes are labeled with colors: white, yellow, pink, cyan, red, green, blue, and black. The graph structure is as follows: white is connected to yellow, pink, and cyan; yellow is connected to white, pink, and red; pink is connected to white, yellow, and green; cyan is connected to white, pink, and blue; red is connected to yellow, pink, and green; green is connected to pink, cyan, blue, and black; blue is connected to cyan and green; and black is connected to green. A terminal window below the graph shows the following interaction:

```

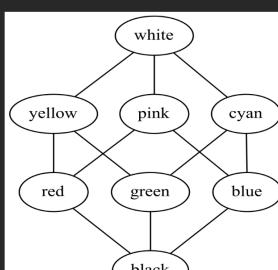
Choose one of the following options
0 -> Print graph details
1 -> Output graph details to a text file
2 -> Add a node to the graph
3 -> Add multiple nodes to the graph
4 -> Add an edge to the graph
5 -> Remove a node from the graph
6 -> Remove multiple nodes from the graph
7 -> Remove an edge from the graph
8 -> Output graph to .dot file
9 -> Output graph to an image file
10 -> Perform BFS search or DFS search in the graph
Q -> Exit the software
Enter your choice
4
Enter the first/source node of the edge
newNode2, white
Choose one of the following options
  
```

A tooltip in the bottom right corner says: "Externally added files can be added to Git View Files Always Add Don't Ask Again".

Commit Link: https://github.com/Divy2000/CSE464_2023_dpage164/commit/c2e5e0723c60eed8c52ad7ac50c373458055bff

Feature 3

- Add an edge to the graph



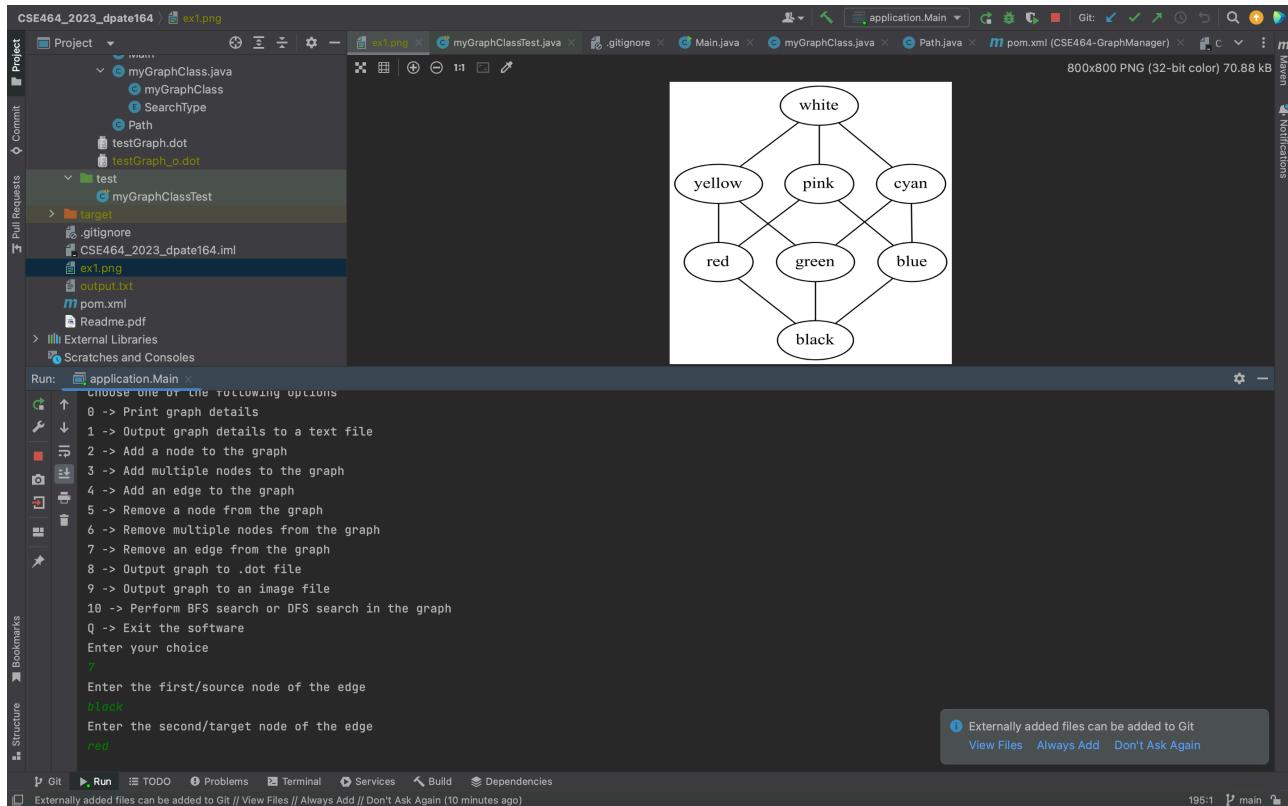
The screenshot shows the same IDE interface and graph as the previous one. The terminal window now displays:

```

Choose one of the following options
0 -> Print graph details
1 -> Output graph details to a text file
2 -> Add a node to the graph
3 -> Add multiple nodes to the graph
4 -> Add an edge to the graph
5 -> Remove a node from the graph
6 -> Remove multiple nodes from the graph
7 -> Remove an edge from the graph
8 -> Output graph to .dot file
9 -> Output graph to an image file
10 -> Perform BFS search or DFS search in the graph
Q -> Exit the software
Enter your choice
4
Enter the first/source node of the edge
newNode1
Enter the second/target node of the edge
black
  
```

A tooltip in the bottom right corner says: "Externally added files can be added to Git View Files Always Add Don't Ask Again".

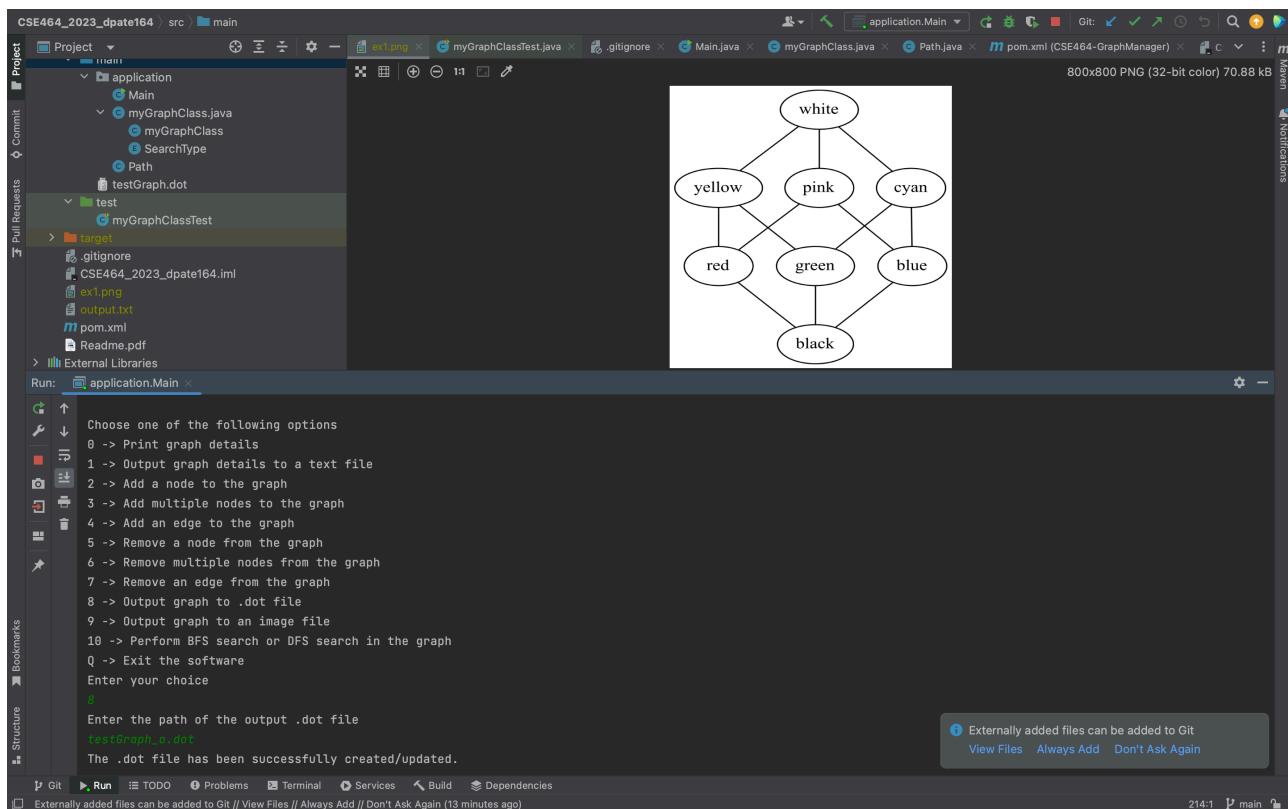
- Remove an edge from the graph



Commit Link: https://github.com/Divy2000/CSE464_2023_ddate164/commit/4fcf172714d5db0b8b19bc522782d1f3f1fa26b0

Feature 4

- Output the DOT graph file



- Output the image file

The screenshot shows the IntelliJ IDEA interface with the project 'CSE464_2023_ddate164' open. The code editor displays `Main.java` which contains logic for handling user input and performing graph operations. Below the code editor is a terminal window showing a menu of options for graph manipulation, such as printing details, adding nodes or edges, and outputting graphs to various formats. The bottom of the screen shows the IntelliJ navigation bar with icons for Git, Run, TODO, Problems, Terminal, Services, Build, and Dependencies.

```

case "6":
    System.out.println("Enter the labels of node you want to remove separated by comma ('a,b,c'):");
    nodeLabels = inputScanner.nextLine();
    nodeLabelList = nodeLabels.split( regex: "," );
    g.removeNodes(nodeLabelList);
    break;
case "7":
    System.out.println("Enter the first/source node of the edge");
    node1 = inputScanner.nextLine();
    System.out.println("Enter the second/target node of the edge");
    node2 = inputScanner.nextLine();
    g.removeEdge(node1, node2);
    break;
case "8":
    System.out.println("Enter the path of the output .dot file");
    filepath = inputScanner.nextLine();

```

Commit Link: https://github.com/Divy2000/CSE464_2023_ddate164/commit/f36ef90b4ae229af14958bf57da1d9c8179503c5

Final commit of finishing the testing portion: https://github.com/Divy2000/CSE464_2023_ddate164/commit/82b8581aa5068c9804c18beb58abe7cc01e21766

Final commit for adding maven support: https://github.com/Divy2000/CSE464_2023_ddate164/commit/8c7363bda391c40d0865358f37e1ee66b59063dc

Final commit for completing the continuous integration: https://github.com/Divy2000/CSE464_2023_ddate164/commit/669e2d62a7a94fbc0a038a561ab0c1f0aa705246

BFS Branch: https://github.com/Divy2000/CSE464_2023_ddate164/tree/bfs

DFS Branch: https://github.com/Divy2000/CSE464_2023_ddate164/tree/dfs

Adding BFS implementation: https://github.com/Divy2000/CSE464_2023_ddate164/commit/c65d58291562086c2838ff501920aae9e405d5bf

Adding DFS implementation: https://github.com/Divy2000/CSE464_2023_ddate164/commit/b6c9a76f85e7a4d1f695e29629fe0a0943fe39b5

Merge branch BFS to main: https://github.com/Divy2000/CSE464_2023_ddate164/commit/119d64372419fbe526ec6756e19c5ca274fc16e6

Merge branch DFS to main: https://github.com/Divy2000/CSE464_2023_ddate164/commit/0192df91cf231a6e8ce38f5ef6508d49a49af2bc

- BFS search example

```

CSE464_2023_dpage164 src main application Main
Project .gitignore ext.png myGraphClassTest.java gitignore Main.java myGraphClass.java Path.java pom.xml (CSE464-GraphManager) Maven Notifications
src workflows maven.yml
main application Main
    myGraphClass.java
    myGraphClass
    SearchType
    Path
    testGraph.dot
    test
        myGraphClassTest
    target
        .gitignore
        CSE464_2023_dpage164.iml
Run: application.Main
case "6":
    System.out.println("Enter the labels of node you want to remove separated by comma ('a,b,c')");
    nodeLabels = inputScanner.nextLine();
    nodeLabelList = nodeLabels.split( regex: "[,]" );
    g.removeNodes(nodeLabelList);
    break;
case "7":
    System.out.println("Enter the first/source node of the edge");
    node1 = inputScanner.nextLine();
    System.out.println("Enter the second/target node of the edge");
    node2 = inputScanner.nextLine();
    g.addEdge(node1, node2);
    break;
case "8":
    System.out.println("Enter the path of the output .dot file");
    filepath = inputScanner.nextLine();
Run: application.Main
↑ 2 -> Add a node to the graph
↓ 3 -> Add multiple nodes to the graph
→ 4 -> Add an edge to the graph
← 5 -> Remove a node from the graph
↶ 6 -> Remove multiple nodes from the graph
↶ 7 -> Remove an edge from the graph
↶ 8 -> Output graph to .dot file
↶ 9 -> Output graph to an image file
↶ 10 -> Perform BFS search or DFS search in the graph
Q -> Exit the software
Enter your choice
10
Enter the first/source node
white
Enter the second/target node
newNode1
Enter the search type 'bfs' or 'dfs'
dfs
white -> cyan -> blue -> black -> newNode1
Externally added files can be added to Git
View Files Always Add Don't Ask Again
Git Run TODO Problems Terminal Services Build Dependencies
Externally added files can be added to Git // View Files // Always Add // Don't Ask Again (22 minutes ago)
257:1 LF UTF-8 4 spaces main

```

- DFS search example

```

CSE464_2023_dpage164 src main application Main
Project .gitignore ext.png myGraphClassTest.java gitignore Main.java myGraphClass.java Path.java pom.xml (CSE464-GraphManager) Maven Notifications
src workflows maven.yml
main application Main
    myGraphClass.java
    myGraphClass
    SearchType
    Path
    testGraph.dot
    test
        myGraphClassTest
    target
        .gitignore
        CSE464_2023_dpage164.iml
Run: application.Main
case "6":
    System.out.println("Enter the labels of node you want to remove separated by comma ('a,b,c')");
    nodeLabels = inputScanner.nextLine();
    nodeLabelList = nodeLabels.split( regex: "[,]" );
    g.removeNodes(nodeLabelList);
    break;
case "7":
    System.out.println("Enter the first/source node of the edge");
    node1 = inputScanner.nextLine();
    System.out.println("Enter the second/target node of the edge");
    node2 = inputScanner.nextLine();
    g.addEdge(node1, node2);
    break;
case "8":
    System.out.println("Enter the path of the output .dot file");
    filepath = inputScanner.nextLine();
Run: application.Main
↑ 2 -> Add a node to the graph
↓ 3 -> Add multiple nodes to the graph
→ 4 -> Add an edge to the graph
← 5 -> Remove a node from the graph
↶ 6 -> Remove multiple nodes from the graph
↶ 7 -> Remove an edge from the graph
↶ 8 -> Output graph to .dot file
↶ 9 -> Output graph to an image file
↶ 10 -> Perform BFS search or DFS search in the graph
Q -> Exit the software
Enter your choice
10
Enter the first/source node
white
Enter the second/target node
newNode1
Enter the search type 'bfs' or 'dfs'
dfs
white -> pink -> blue -> black -> newNode1
Externally added files can be added to Git
View Files Always Add Don't Ask Again
Git Run TODO Problems Terminal Services Build Dependencies
Externally added files can be added to Git // View Files // Always Add // Don't Ask Again (23 minutes ago)
280:1 LF UTF-8 4 spaces main

```