# Grapher README

This Java application, **Grapher**, allows you to parse, manipulate, and visualize directed graphs. It utilizes the JGraphT library for graph manipulation and the JGraphX library for graph visualization.

## Features

1. **Parsing Graph from DOT File**
   - **Method:** `parseGraph(String filePath)`
   - **Description:** Parses a graph from a DOT file.
   - **Example:**

```
Grapher grapher = new Grapher();
Graph<String, DefaultEdge> graph = grapher.parseGraph("path/to/your/gra
```

2. **Adding Nodes**
   - **Methods:** `addNode(String label)` , `addNodes(String[] labels)`
   - **Description:** Adds nodes to the graph.
   - **Example:**

```
grapher.addNode("A");
grapher.addNodes(new String[]{"B", "C"});
```

3. **Adding Edges**
   - **Method:** `addEdge(String srcLabel, String dstLabel)`
   - **Description:** Adds directed edges between nodes.

- **Example:**

```
grapher.addEdge("A", "B");
```

4. **Exporting Graph to DOT Format**

   - **Method:** `outputDOTGraph(String filePath)`
   - **Description:** Exports the graph in DOT format to a file.
   - **Example:**

```
grapher.outputDOTGraph("path/to/save/graph.dot");
```

5. **Exporting Graph as Image**

   - **Method:** `outputGraphics(String filePath)`
   - **Description:** Exports the graph as an image file (PNG format).
   - **Example:**

```
grapher.outputGraphics("path/to/save/graph.png");
```

6. **Generating Graph Information**

   - **Method:** `toString()`
   - **Description:** Generates a string containing graph information, including nodes and edges.
   - **Example:**

```
String graphInfo = grapher.toString();
```

7. **Writing Graph Information to File**

   - **Method:** `writeToFile(String filePath)`
   - **Description:** Writes graph information to a text file.
   - **Example:**

```
grapher.writeToFile("path/to/save/graphInfo.txt");
```

8. **Removing a Node**

   - **Method:** `removeNode(String label)`
   - **Description:** Removes a node from the graph.
   - **Example:**

```
grapher.removeNode("B");
```

9. **Removing multiple Nodes**
    - **Method:** `removeNodes(String[] labels)`
    - **Description:** Removes multiple nodes from the graph
    - **Example:**

```
String[] nodesToRemove1 = {"A", "B"};
grapher.removeNodes(nodesToRemove1);
```

10 **Removing an Edge**

- **Method:** `removeEdge(String srcLabel, String dstLabel)`
- **Description:** Removes an edge from the graph.
- **Example:**

```
grapher.removeEdge("A", "B");
```

11. **Searching for a path in the graph**

- **Method:** `Path graphSearch(String src, String dst, Algorithm algo)`
- **Description:** Finds a path from a source node to destination node in BFS or DFS as specified.
- **Example:**

```
grapher.graphSearch("A", "E", Algorithm.BFS);
```

## How to Run

1. Clone the repository - Link
2. **Compile the Code:**

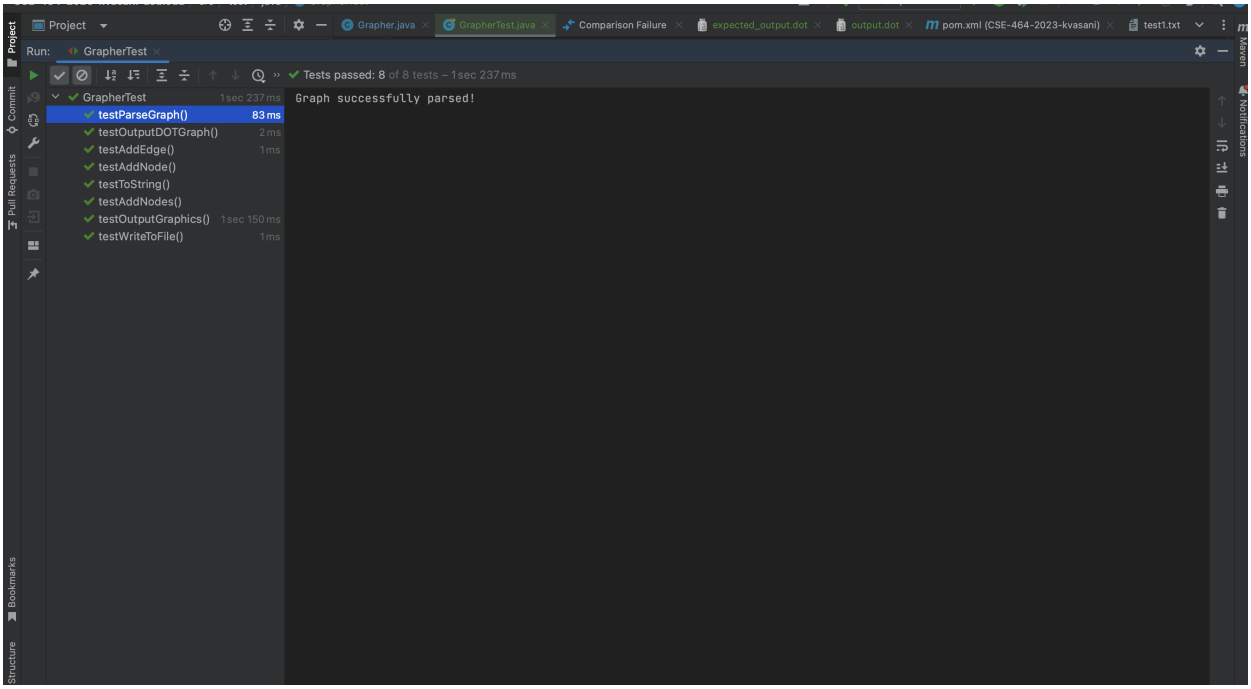
```
mvn package
```

3. **Run Tests:**

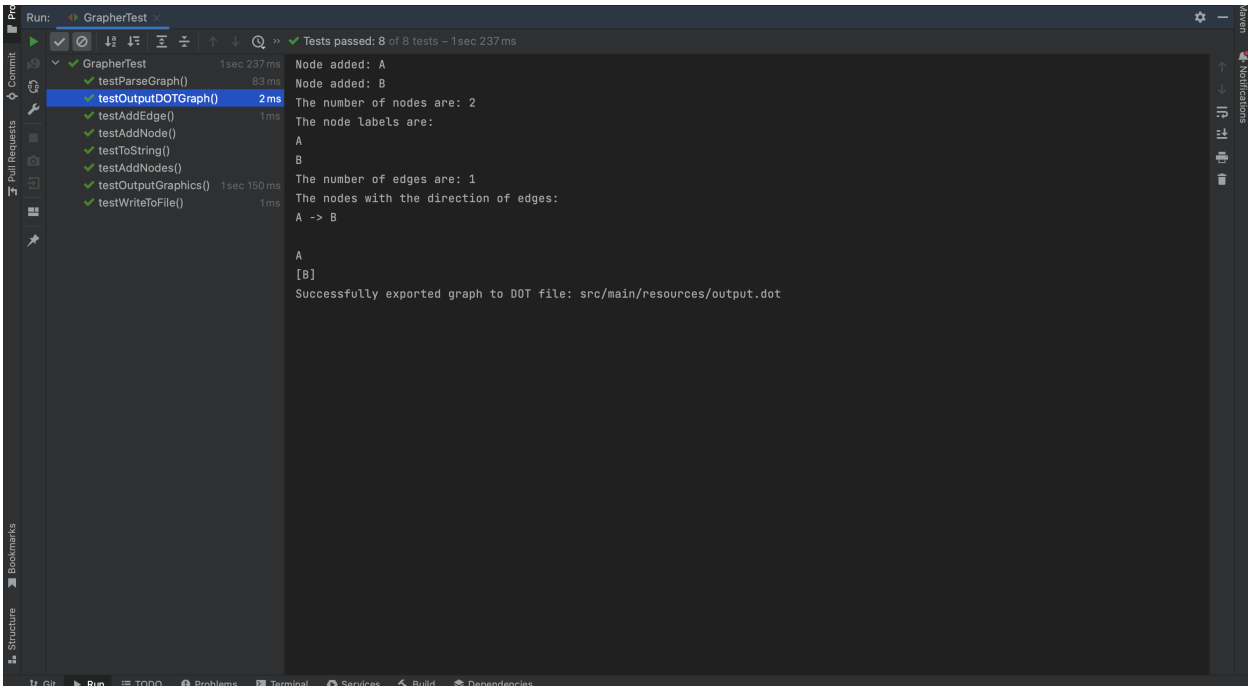
```
mvn test
```

## Screenshots

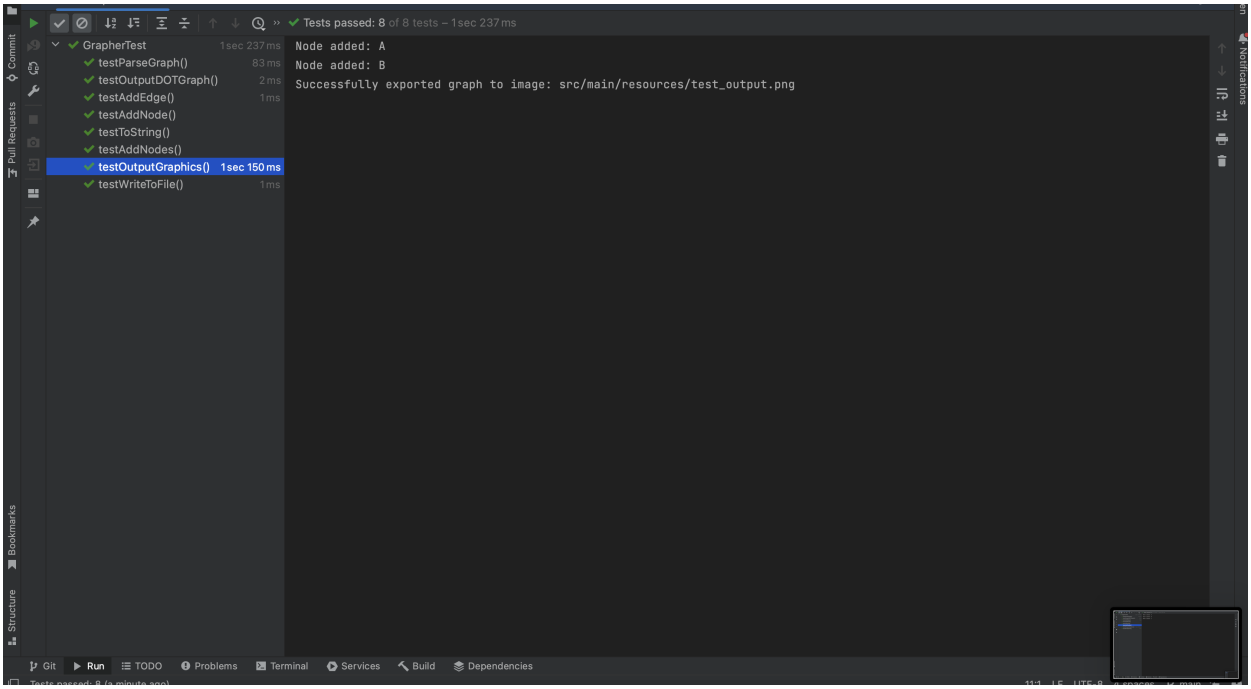
- Parsed Graph Information:




- Exported Graph (DOT Format):

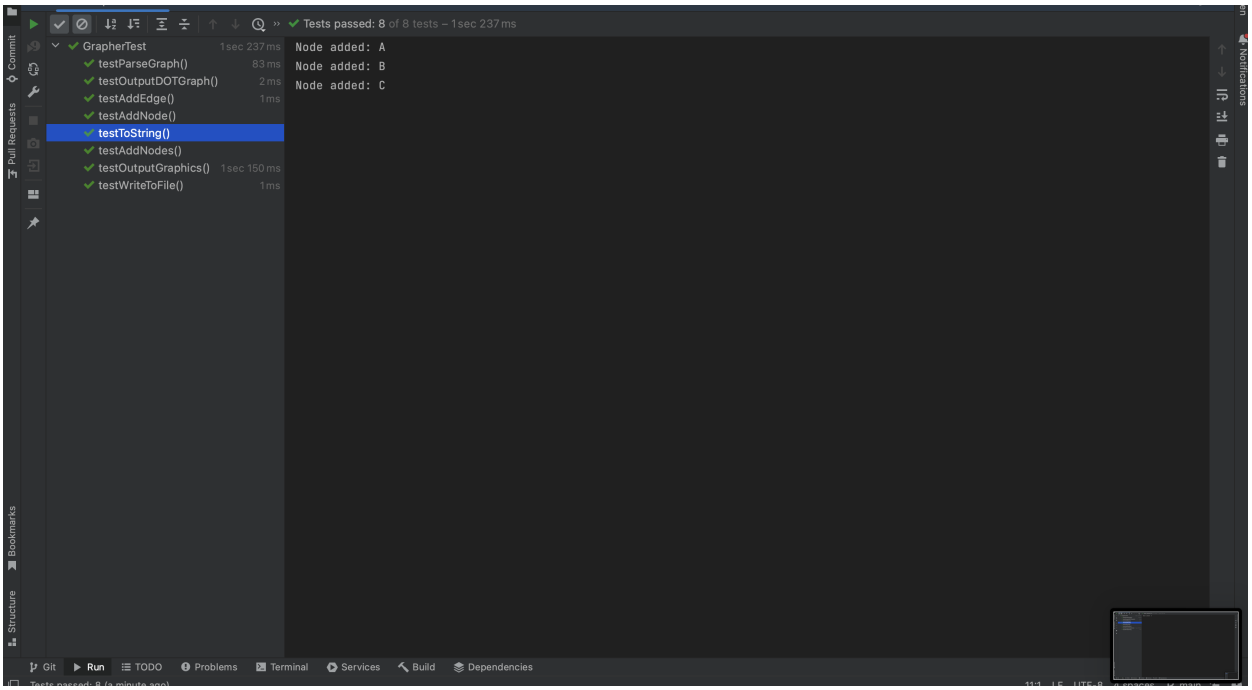


- Exported Graph (Image):

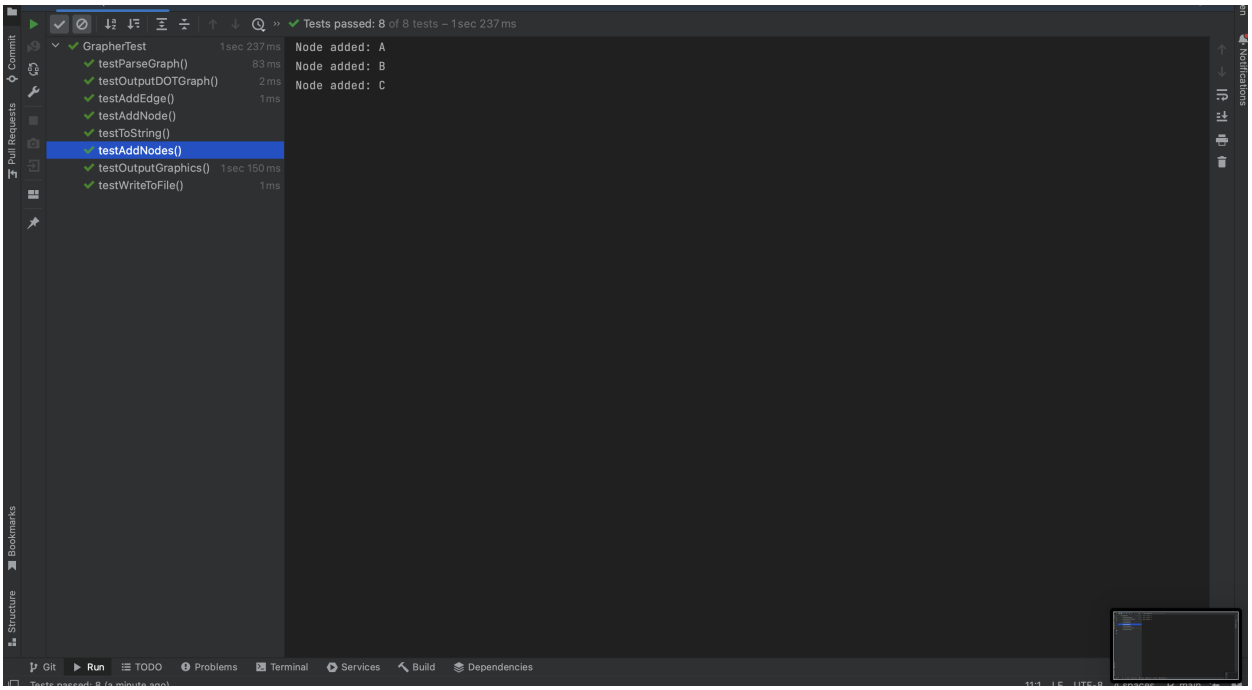



- Exported Graph (Image):

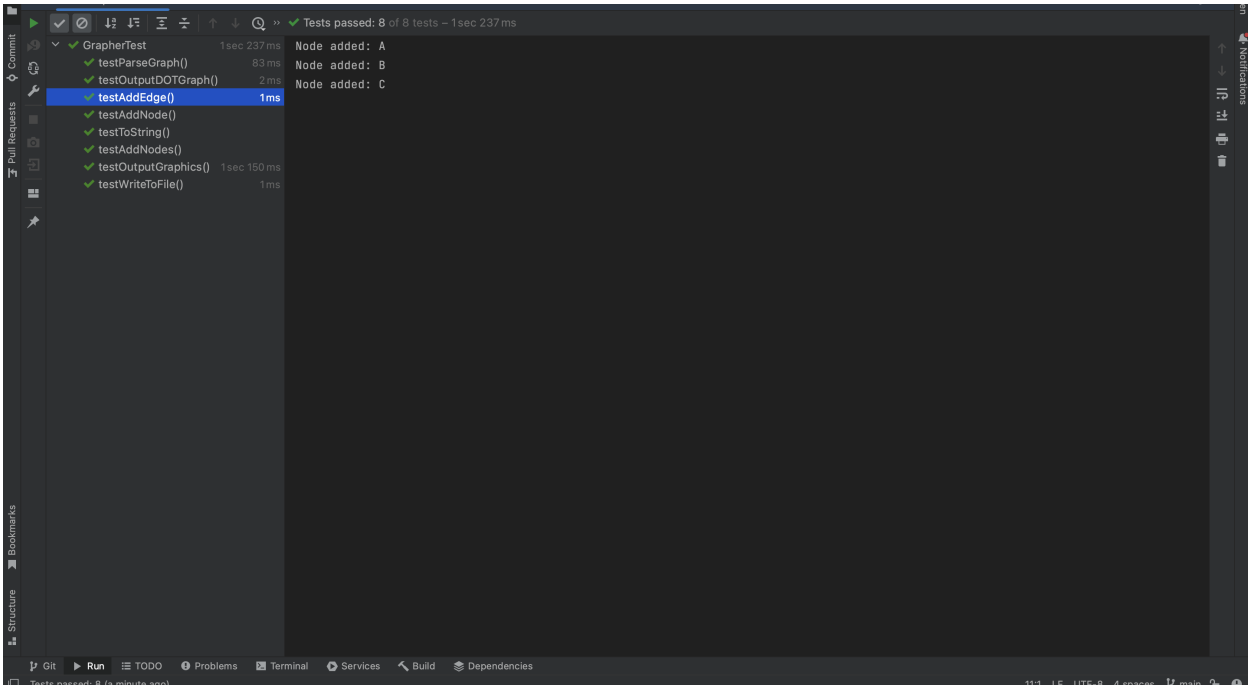

- Output to String:

- 

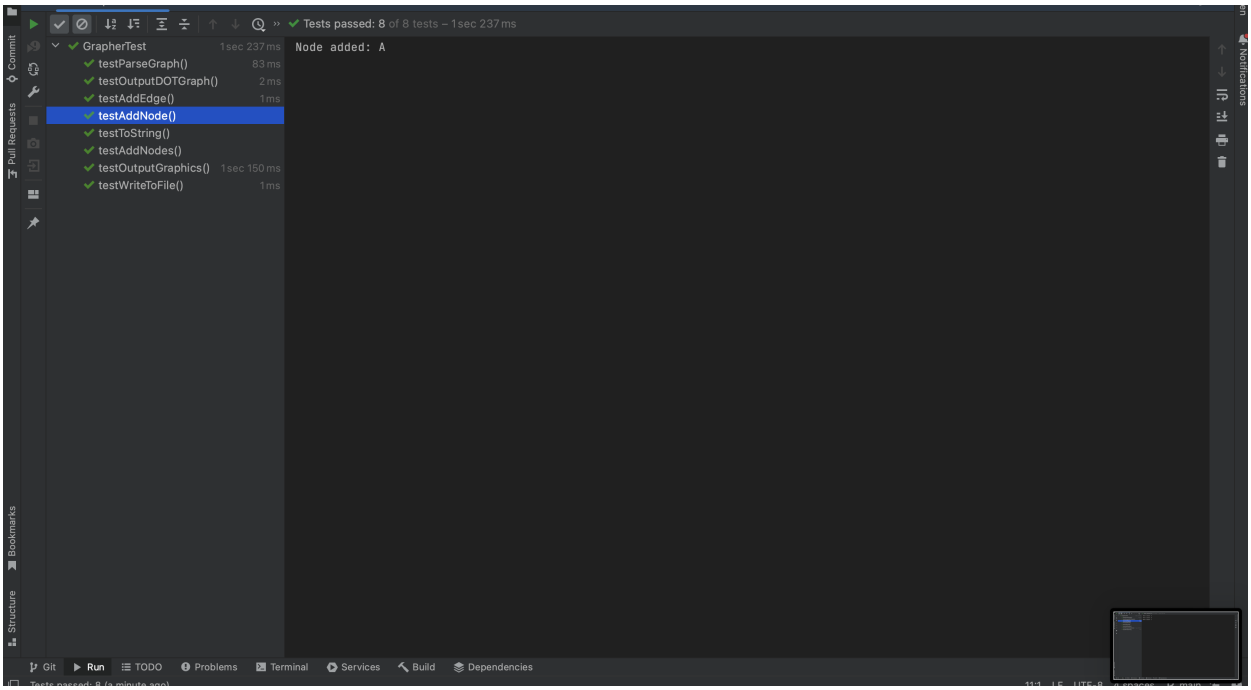

- Adding list of nodes:

- 

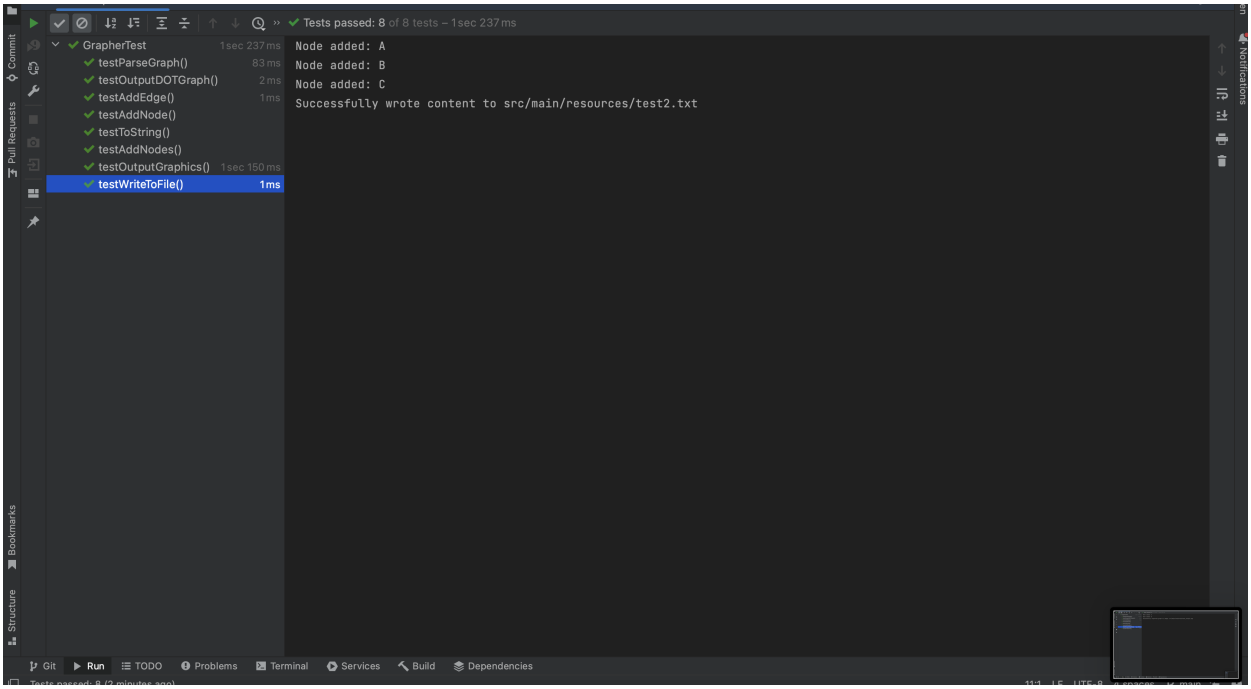
- Added Edges:




- Added Node:



- Write Graph to text file:




- Remove a Node from the Graph:

```
Node added: F
The number of nodes are: 5
The node labels are:
A
B
C
D
F
The number of edges are: 5
The nodes with the direction of edges:
A -> B
B -> C
C -> A
A -> D
D -> F

Node present in graph
The number of nodes are: 4
The node labels are:
A
B
D
F
The number of edges are: 3
The nodes with the direction of edges:
A -> B
A -> D
D -> F
```

- Remove multiple Nodes from the Graph:

```
Node added: D
Node added: E
The number of nodes are: 5
The node labels are:
A
B
C
D
E
The number of edges are: 4
The nodes with the direction of edges:
A -> B
B -> C
C -> A
A -> D

The number of nodes are: 5
The node labels are:
A
B
C
D
E
The number of edges are: 4
The nodes with the direction of edges:
A -> B
B -> C
C -> A
A -> D

Node present in graph
Node present in graph
```

```
Node not present in graph
Node not present in graph
The number of nodes are: 3
The node labels are:
C
D
E
The number of edges are: 0
The nodes with the direction of edges:

Node not present in graph
Node not present in graph

Process finished with exit code 0
```

- Created a BFS branch:

```
Graph successfully parsed!
Node added: D
Node added: E
BFS path traversed : A -> D -> E
```

- Created a DFS branch:

```
Node added: A
Node added: B
Node added: C
DFS path traversed : A -> B -> C
```

- Searching for a path from source node to destination node (merged conflicts):

```
/Library/Java/JavaVirtualMachines/jdk-21.jdk/Contents/Home/bin/java ...
Graph successfully parsed!
Node added: D
Node added: E
BFS Traversed : A -> D -> E
Node added: A
Node added: B
Node added: C
DFS Traversed: A -> B -> C
Node already exists: D

Process finished with exit code 0
```

## Commits

- Initial commit
- Built Maven. Also added feature 1
- Finished feature 1. This commit outputs the graph and writes it to a text file.
- Finished feature 2. Node and list of nodes can now be added. The result is reflected in the output of the graph.
- Finished Feature 3. The Edges are added to the graph and it is reflected when the graph is outputed.
- Finished Feature 4. The graph is visible in the dot file and a png image is also formed to visualize the graph.
- Added all the tests and the finishing touches
- Added APIs for removing a node, removing multiple nodes and removing an edge
- Created maven.yml
- Uncommented previous test cases
- Merged the maven into main
- Added bfs branch
- Added dfs branch
- Merged conflicts between dfs and bfs branches