Links:

Remove Node Function:
https://github.com/JulianGomez1604/CSE_464_2024_jjgomezm/commit/e2fbd562af65e410bd9942e8b5ea54a68579945b
Removes Nodes Function:
https://github.com/JulianGomez1604/CSE_464_2024_jjgomezm/commit/6c4e7fb409616e0e74fdb9a1eec3b9ab5b7dd354
Remove Edges Function:
https://github.com/JulianGomez1604/CSE_464_2024_jjgomezm/commit/ec6547fb890c9478ce53f92a5557e5b3ac4565a5
Test Cases for scenarios:
https://github.com/JulianGomez1604/CSE_464_2024_jjgomezm/commit/eaadb0153821f7b232e0021078c1377d68683e92
Continuous Integration:
https://github.com/JulianGomez1604/CSE_464_2024_jjgomezm/commit/c9a505ed622750ac7c7618c84474815f2271b747
Final Commit bfs Branch:
https://github.com/JulianGomez1604/CSE_464_2024_jjgomezm/commit/9f8324c23ea833cf7459d797217a28cc472ce672
Final Commit dfs branch:
https://github.com/JulianGomez1604/CSE_464_2024_jjgomezm/commit/b92951ec0beed35f44f326e343448d1bfdeede99

Merges:

Merge bfs and main:
https://github.com/JulianGomez1604/CSE_464_2024_jjgomezm/commit/6b0f6d4b9ca3b4a2a719f20b52016bda358ed404
Merge dfs and main(resolved conflicts):
https://github.com/JulianGomez1604/CSE_464_2024_jjgomezm/commit/bd95dadee9f0a49d1f45ae46b33bb28a4373c434

Running Test Cases:

For all test cases:

```
mvn -Dtest=testFirstFeature,testSecondFeature,testThirdFeature,testFourthFeature,testPart2 test
```

For only new cases:

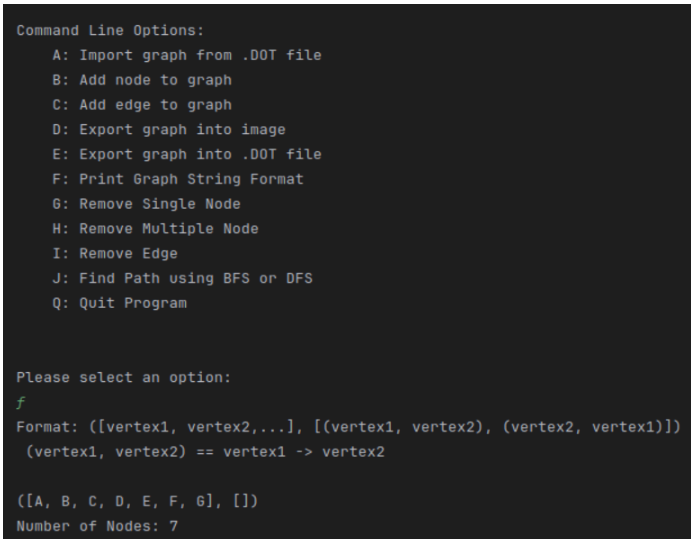
```
mvn -Dtest=testPart2 test
```

Remove Node Feature:

Inputs: (Input a single line at a time)

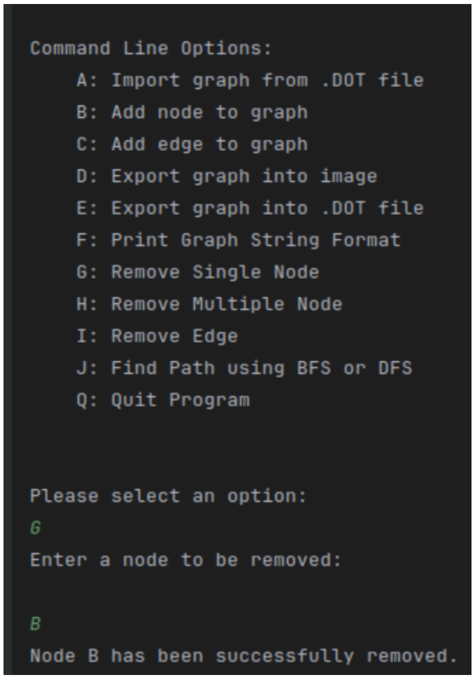
B
A, B, C, D, E, F, G
F
G
B
F

Output: (Not showing all inputs results but whats needed)
After inputting new nodes:

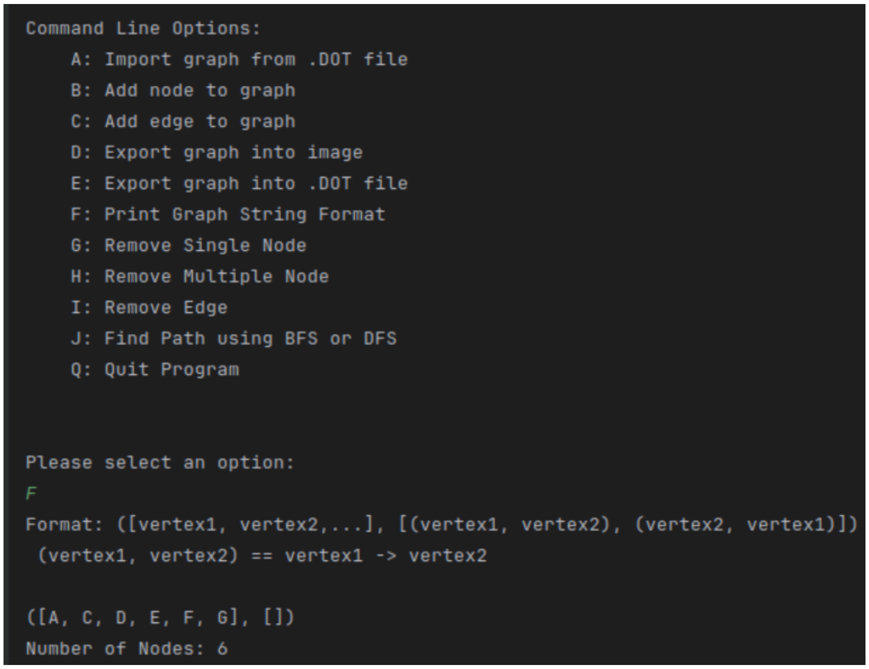


```
Command Line Options:
    A: Import graph from .DOT file
    B: Add node to graph
    C: Add edge to graph
    D: Export graph into image
    E: Export graph into .DOT file
    F: Print Graph String Format
    G: Remove Single Node
    H: Remove Multiple Node
    I: Remove Edge
    J: Find Path using BFS or DFS
    Q: Quit Program


Please select an option:
f
Format: ([vertex1, vertex2,...], [(vertex1, vertex2), (vertex2, vertex1)])
 (vertex1, vertex2) == vertex1 -> vertex2

([A, B, C, D, E, F, G], [])
Number of Nodes: 7
```

After removing node:



```
Command Line Options:
    A: Import graph from .DOT file
    B: Add node to graph
    C: Add edge to graph
    D: Export graph into image
    E: Export graph into .DOT file
    F: Print Graph String Format
    G: Remove Single Node
    H: Remove Multiple Node
    I: Remove Edge
    J: Find Path using BFS or DFS
    Q: Quit Program


Please select an option:
G
Enter a node to be removed:

B
Node B has been successfully removed.
```



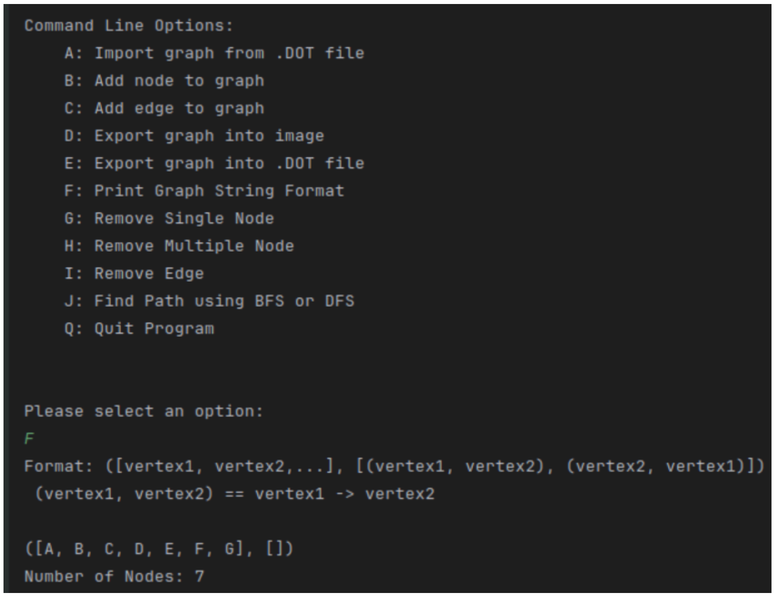
```
Command Line Options:
    A: Import graph from .DOT file
    B: Add node to graph
    C: Add edge to graph
    D: Export graph into image
    E: Export graph into .DOT file
    F: Print Graph String Format
    G: Remove Single Node
    H: Remove Multiple Node
    I: Remove Edge
    J: Find Path using BFS or DFS
    Q: Quit Program


Please select an option:
F
Format: ([vertex1, vertex2,...], [(vertex1, vertex2), (vertex2, vertex1)])
 (vertex1, vertex2) == vertex1 -> vertex2

([A, C, D, E, F, G], [])
Number of Nodes: 6
```

Remove Nodes Feature:

Inputs: (Input a single line at a time)

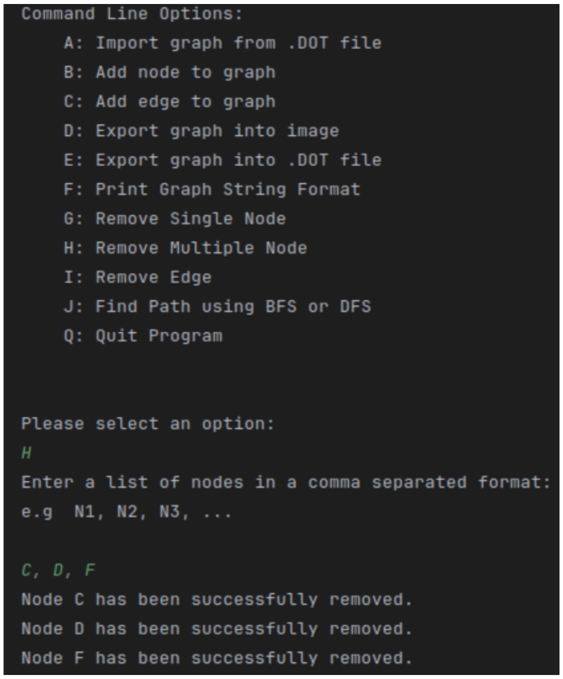
B
A, B, C, D, E, F, G
F
H
C, D, F
F

After Adding Nodes:

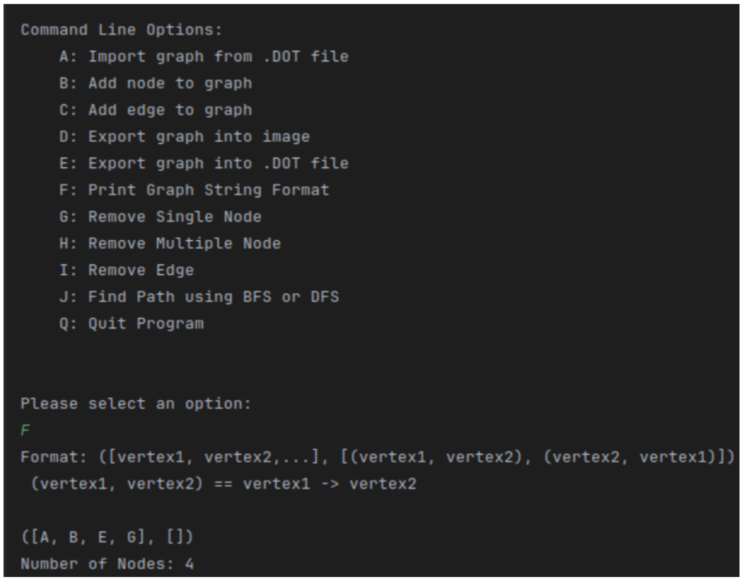


```
Command Line Options:
    A: Import graph from .DOT file
    B: Add node to graph
    C: Add edge to graph
    D: Export graph into image
    E: Export graph into .DOT file
    F: Print Graph String Format
    G: Remove Single Node
    H: Remove Multiple Node
    I: Remove Edge
    J: Find Path using BFS or DFS
    Q: Quit Program


Please select an option:
F
Format: ([vertex1, vertex2,...], [(vertex1, vertex2), (vertex2, vertex1)])
 (vertex1, vertex2) == vertex1 -> vertex2

([A, B, C, D, E, F, G], [])
Number of Nodes: 7
```

After removing nodes:



```
Command Line Options:
    A: Import graph from .DOT file
    B: Add node to graph
    C: Add edge to graph
    D: Export graph into image
    E: Export graph into .DOT file
    F: Print Graph String Format
    G: Remove Single Node
    H: Remove Multiple Node
    I: Remove Edge
    J: Find Path using BFS or DFS
    Q: Quit Program


Please select an option:
H
Enter a list of nodes in a comma separated format:
e.g  N1, N2, N3, ...

C, D, F
Node C has been successfully removed.
Node D has been successfully removed.
Node F has been successfully removed.
```



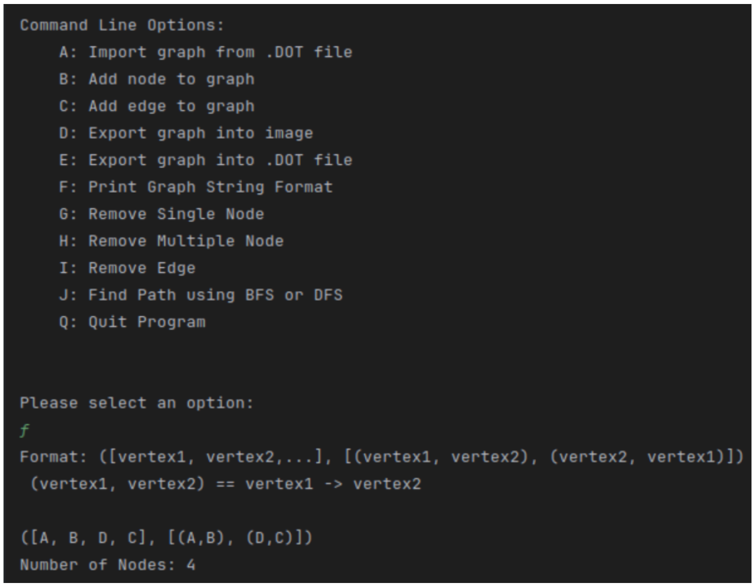
```
Command Line Options:
    A: Import graph from .DOT file
    B: Add node to graph
    C: Add edge to graph
    D: Export graph into image
    E: Export graph into .DOT file
    F: Print Graph String Format
    G: Remove Single Node
    H: Remove Multiple Node
    I: Remove Edge
    J: Find Path using BFS or DFS
    Q: Quit Program


Please select an option:
F
Format: ([vertex1, vertex2,...], [(vertex1, vertex2), (vertex2, vertex1)])
 (vertex1, vertex2) == vertex1 -> vertex2

([A, B, E, G], [])
Number of Nodes: 4
```

Remove Edges Feature:

Inputs: (Input a single line at a time)

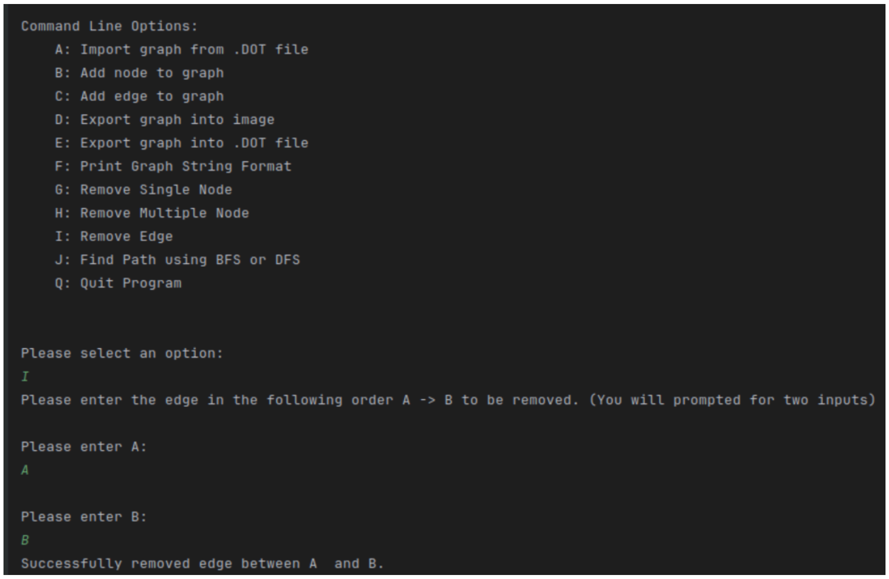
C
A
B
C
D
C
F
I
A
B
F

After adding edges:

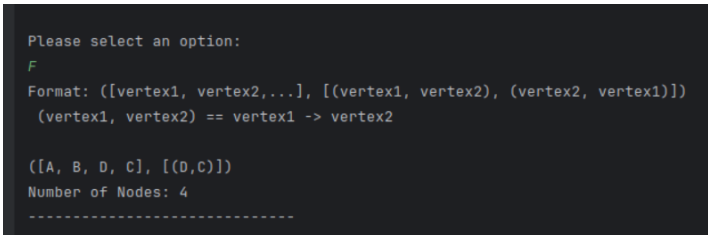


```
Command Line Options:
    A: Import graph from .DOT file
    B: Add node to graph
    C: Add edge to graph
    D: Export graph into image
    E: Export graph into .DOT file
    F: Print Graph String Format
    G: Remove Single Node
    H: Remove Multiple Node
    I: Remove Edge
    J: Find Path using BFS or DFS
    Q: Quit Program


Please select an option:
f
Format: ([vertex1, vertex2,...], [(vertex1, vertex2), (vertex2, vertex1)])
 (vertex1, vertex2) == vertex1 -> vertex2

([A, B, D, C], [(A,B), (D,C)])
Number of Nodes: 4
```

After removing edges:



```
Command Line Options:
    A: Import graph from .DOT file
    B: Add node to graph
    C: Add edge to graph
    D: Export graph into image
    E: Export graph into .DOT file
    F: Print Graph String Format
    G: Remove Single Node
    H: Remove Multiple Node
    I: Remove Edge
    J: Find Path using BFS or DFS
    Q: Quit Program


Please select an option:
I
Please enter the edge in the following order A -> B to be removed. (You will prompted for two inputs)

Please enter A:
A

Please enter B:
B
Successfully removed edge between A  and B.
```



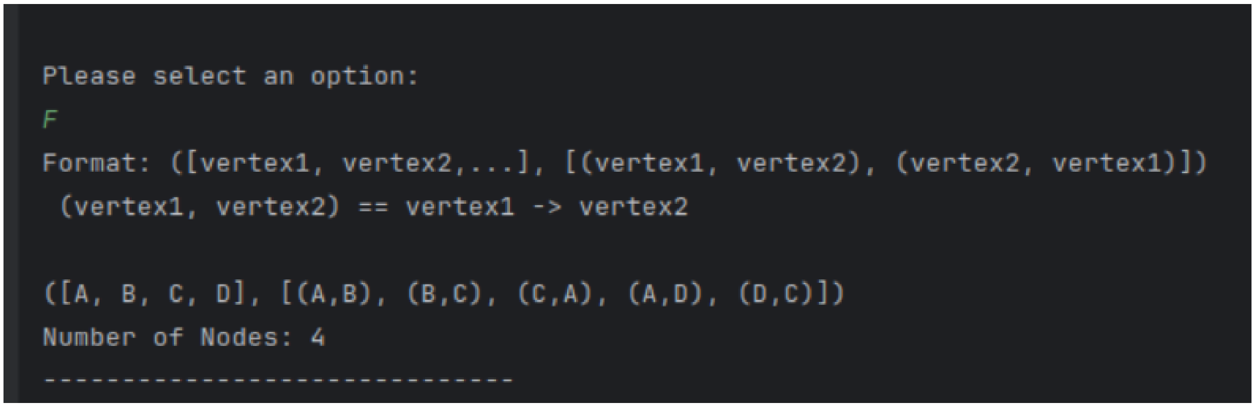
```
Please select an option:
F
Format: ([vertex1, vertex2,...], [(vertex1, vertex2), (vertex2, vertex1)])
 (vertex1, vertex2) == vertex1 -> vertex2

([A, B, D, C], [(D,C)])
Number of Nodes: 4
------------------------------
```

BFS or DFS search:

Inputs:

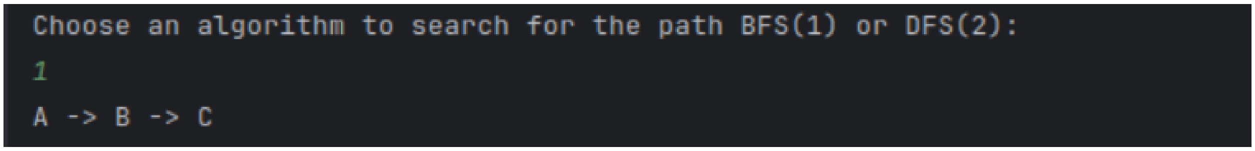
A
src/dot files/graph1.dot
F
J
A
C
1 or 2 *here is where you decide which search algorithm to use*

Importing graph for faster test:



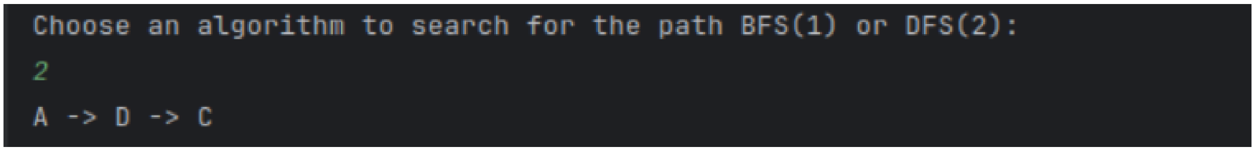
```
Please select an option:
F
Format: ([vertex1, vertex2,...], [(vertex1, vertex2), (vertex2, vertex1)])
 (vertex1, vertex2) == vertex1 -> vertex2

([A, B, C, D], [(A,B), (B,C), (C,A), (A,D), (D,C)])
Number of Nodes: 4
------------------------------
```

After choosing 1:



```
Choose an algorithm to search for the path BFS(1) or DFS(2):
1
A -> B -> C
```

After choosing 2:



```
Choose an algorithm to search for the path BFS(1) or DFS(2):
2
A -> D -> C
```