README for CSE 464

Project Part 2

Shantanu Shishodia
1225590054

GitHub Repository link: https://github.com/shantanushishodia/cse-464-2023-sshishod

Instructions to Run

- Download the cse-464-sshishod.zip file from this repository
- Run mvn package
- This should run all tests for the project
- This command will build the project in the target folder as well
- Alternatively, unzip cse-464-sshishod.zip and then open the GraphHandler folder in IntelliJ

APIs

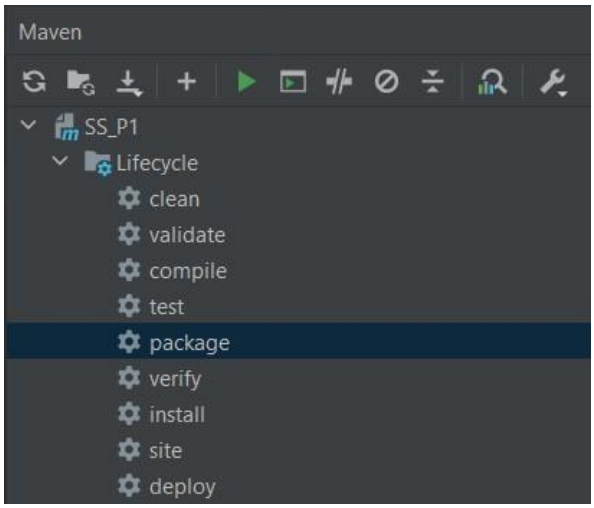
- void graphImporter(String filePath) - import a directed graph in a dot file
- String toString() - Graph information like number of nodes, edges and their directions
- void saveGraphToFile(String filePath) - Write the graph information to a file
- void addOneNode(String label) - Adds a new node to the graph with the given label if it does not exist
- void addMultipleNodes(ArrayList<String> labels) - Add multiple nodes to the graph
- boolean removeNode(String label) - Returns false if node does not exist or returns true if node is removed
- boolean removeNodes(String[] labels) - Returns true if all nodes are removed successfully otherwise returns false if even one node exists
- boolean addEdge(String initialNode, String targetNode) - Returns true if edge is added otherwise returns false if edge exists
- boolean removeEdge(String srcLabel, String dstLabel) - Returns true if edge is removed successfully otherwise returns false if edge does not exist in the graph
- void saveGraphDOT(String filePath) - Outputs the modified graph in DOT format to the specified file
- void saveGraphPNG(String filePath) - Output the modified graph to a PNG file (Graph Visualization)
- void GraphSearch(String src, String dst, Algorithm algo) - Find a path from src to dst node using BFS or DFS algorithm depending on enum specified. Possible values of the enum can be BFS, DFS

Adding Maven support to the project

- I was following the standard directory layout from when I had started my project by following the guide at https://maven.apache.org/guides/introduction/introduction-to-the-standard-directo ry-layout.html
- I created a pom.xml and added all my project dependencies with the feature 1 implemented. The commit which contains this change: https://github.com/shantanushishodia/cse-464-2023sshishod/commit/02bfb1b77b5f9b57182503ffa81ad2fad3011a4e
- A later commit fixed the tests not running while executing the mvn package command in which the maven-surefire-plugin needed to be changed to 2.22.0: https://github.com/shantanushishodia/cse-464-2023sshishod/commit/4775c744f1294f2ddfae27a2b43b71611f7085b9

1. Output for mvn package command (test performed using test1.dot as initial input)

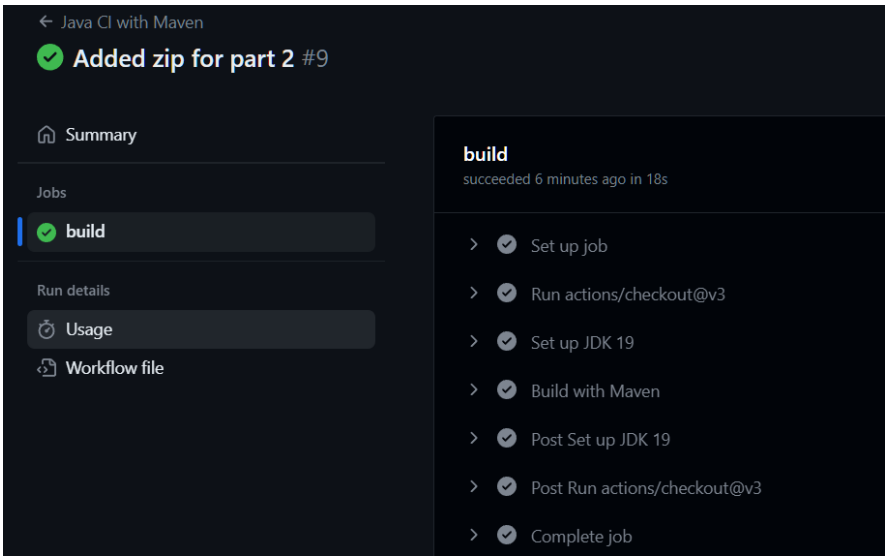
Can use both way to initiate maven package

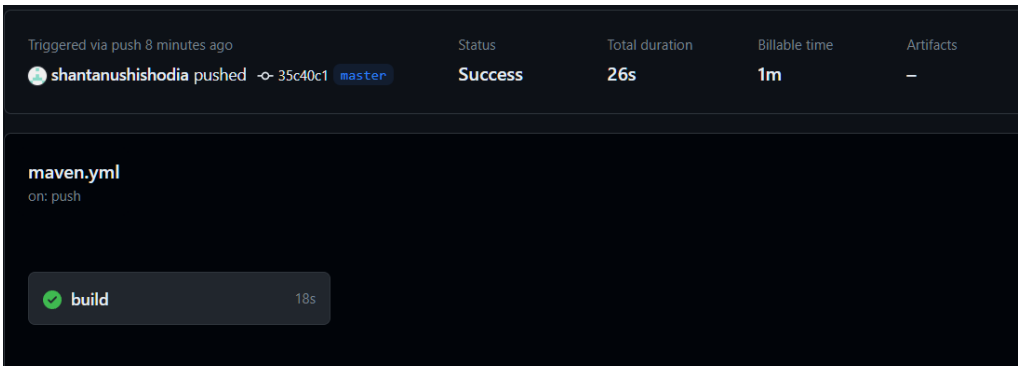



```
[INFO] Tests run: 12, Failures: 0, Errors: 0, Skipped: 0, Time elapsed: 0.256 s - in GraphHandlerTest
[INFO]
[INFO] Results:
[INFO]
[INFO] Tests run: 12, Failures: 0, Errors: 0, Skipped: 0
[INFO]
[INFO]
[INFO] --- maven-jar-plugin:2.4:jar (default-jar) @ SS_P1 ---
[INFO] Building jar: C:\Users\shant\IdeaProjects\SS_P1\target\SS_P1-1.0-SNAPSHOT.jar
[INFO] ------------------------------------------------------------------------
[INFO] BUILD SUCCESS
[INFO] ------------------------------------------------------------------------
[INFO] Total time:  4.895 s
[INFO] Finished at: 2023-11-05T11:45:32-07:00
[INFO] ------------------------------------------------------------------------

Process finished with exit code 0
```

2. 2. Adding Github CI to the project

- The initial commit where I added the workflow file to the repository is: https://github.com/shantanushishodia/cse-464-2023-sshishod/commit/36768004acefd162c82dfcee6093fc1878759c41
- The workflow runs on every push to the main branch or when a new pull request is opened to the main branch
- Updated the workflow in the following commit: https://github.com/shantanushishodia/cse-464-2023-sshishod/commit/b2c5b9671150ec1be08cca34e3412bb097ad7944
- Some screenshots to show the Github CI working and producing builds of the project





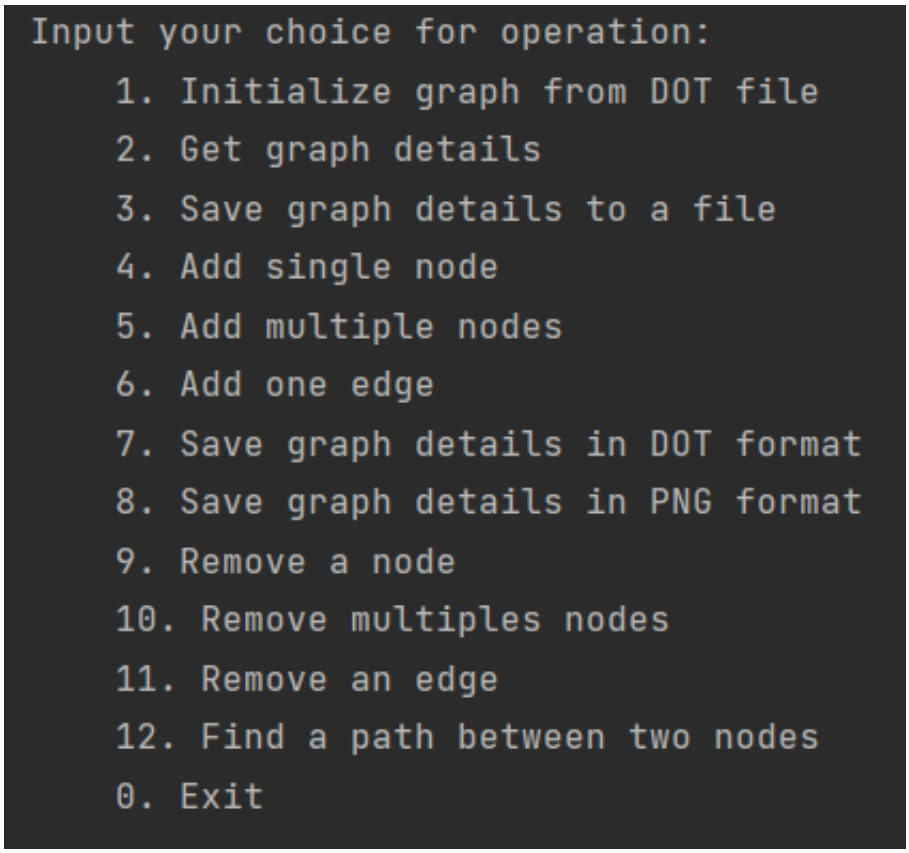
3. **Added three new features including removing one node, multiple nodes and removing an edge. Also updated all test cases with edge case handling.**

Commit for removing a node: https://github.com/shantanushishodia/cse-464-2023-sshishod/commit/04cfb73417d17b80eaa29487413d6618c8f28091

Commit for removing multiple nodes: https://github.com/shantanushishodia/cse-464-2023-sshishod/commit/2d61bd5b2909280c5989ec98a10d661a8765c3f8

Commit for removing an edge: https://github.com/shantanushishodia/cse-464-2023-sshishod/commit/d69211b4ca65a6ad77a42e2724a859b0157c5ac7

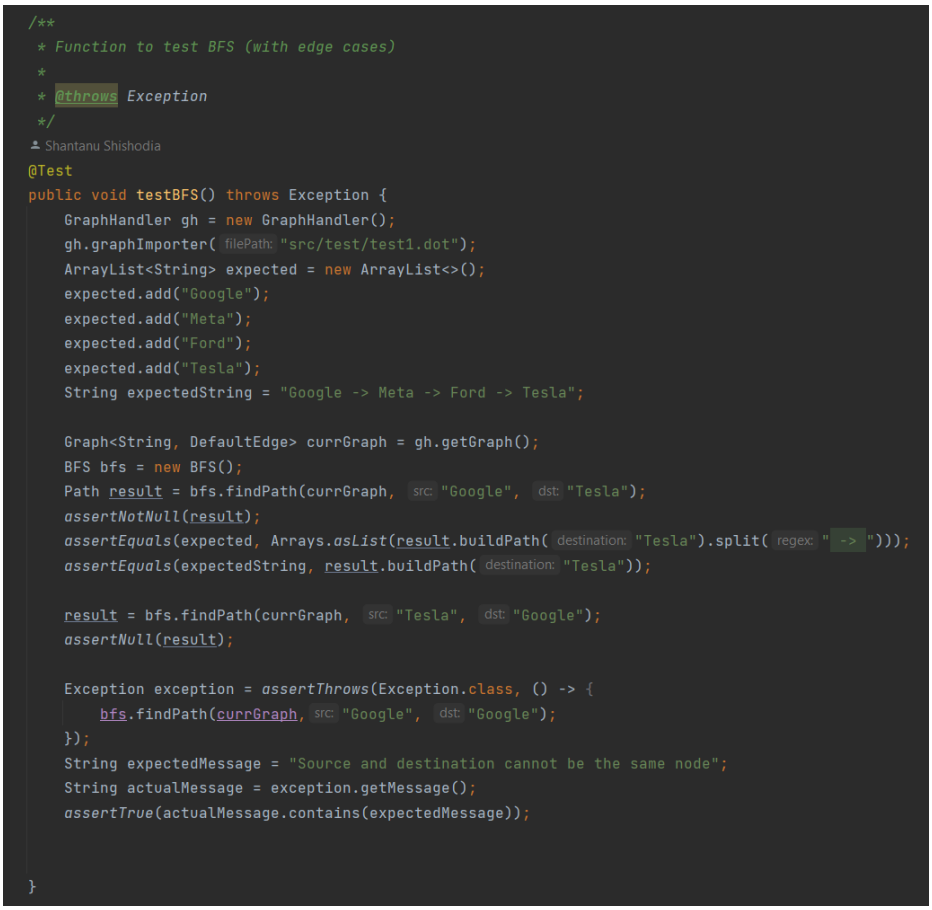
Test case update commit: https://github.com/shantanushishodia/cse-464-2023-sshishod/commit/f85a7c2e8531144127498a4b4bd1fb44c8bccf12



4. Adding BFS algorithm

- I created a new API GraphSearch which finds a path between two input nodes and displays it using Path class. It uses a custom BFS class to implement the algorithm.
- I created a separate branch 'bfs' https://github.com/shantanushishodia/cse-464-2023-sshishod/tree/bfs
- And then opened a pull request to merge the code from the bfs branch to master branch https://github.com/shantanushishodia/cse-464-2023-sshishod/pull/1
- The commit that added the initial BFS functionality and tests is https://github.com/shantanushishodia/cse-464-2023-sshishod/commit/a494653dee2c781310a33efc7c884e84c597058c

The figure below shows the test function for BFS:



```
/**
 * Function to test BFS (with edge cases)
 *
 * @throws Exception
 */
Shantanu Shishodia
@Test
public void testBFS() throws Exception {
    GraphHandler gh = new GraphHandler();
    gh.graphImporter( filePath: "src/test/test1.dot");
    ArrayList<String> expected = new ArrayList<>();
    expected.add("Google");
    expected.add("Meta");
    expected.add("Ford");
    expected.add("Tesla");
    String expectedString = "Google -> Meta -> Ford -> Tesla";

    Graph<String, DefaultEdge> currGraph = gh.getGraph();
    BFS bfs = new BFS();
    Path result = bfs.findPath(currGraph, src: "Google", dst: "Tesla");
    assertNotNull(result);
    assertEquals(expected, Arrays.asList(result.buildPath( destination: "Tesla").split( regex: " -> ")));
    assertEquals(expectedString, result.buildPath( destination: "Tesla"));

    result = bfs.findPath(currGraph, src: "Tesla", dst: "Google");
    assertNull(result);

    Exception exception = assertThrows(Exception.class, () -> {
        bfs.findPath(currGraph, src: "Google", dst: "Google");
    });
    String expectedMessage = "Source and destination cannot be the same node";
    String actualMessage = exception.getMessage();
    assertTrue(actualMessage.contains(expectedMessage));


}
```

5. Adding DFS Algorithm

- I created a separate dfs branch from master which did the same changes as the BFS but for DFS algo. https://github.com/shantanushishodia/cse-464-2023-sshishod/tree/dfs
- The pull request to merge this: https://github.com/shantanushishodia/cse-464-2023-sshishod/pull/2
- Commit that added initial DFS functionality: https://github.com/shantanushishodia/cse-464-2023-sshishod/commit/0fae00f6210484a7755ebb33b80412db0b3bb20e

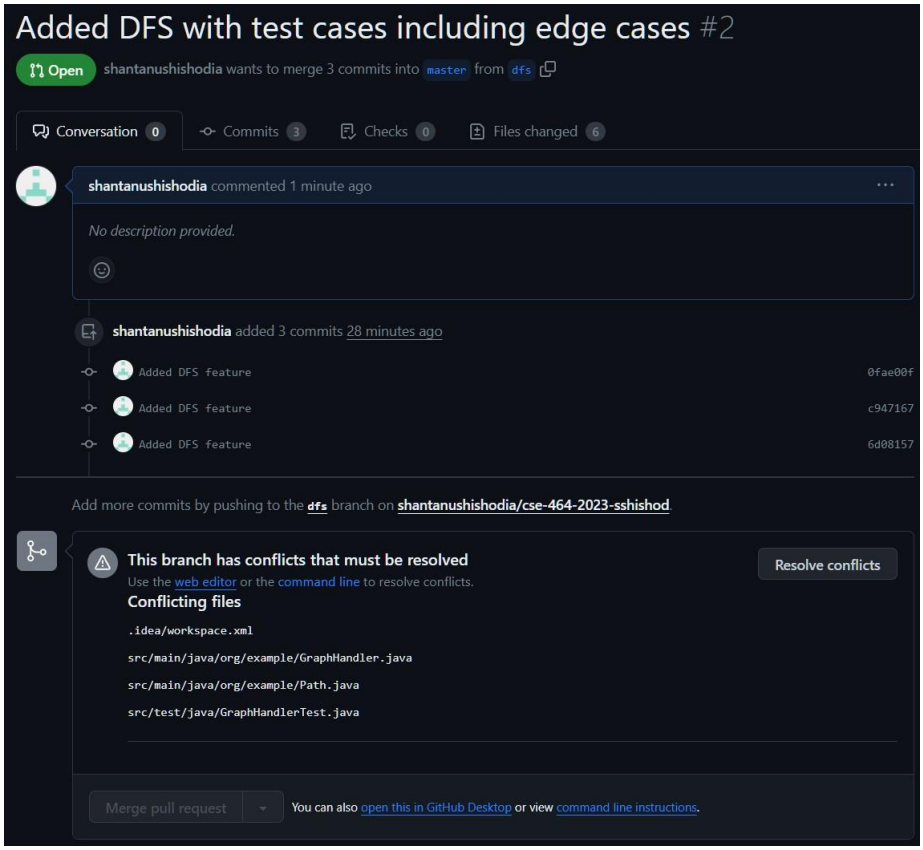
The figure below shows the test function for DFS:

```java
/**
 * Function to test DFS (with edge cases)
 *
 * @throws Exception
 */
Shantanu Shishodia
@Test
public void testDFS() throws Exception {
    GraphHandler gh = new GraphHandler();
    gh.graphImporter( filePath: "src/test/test1.dot");
    ArrayList<String> expected = new ArrayList<>();
    expected.add("Google");
    expected.add("Meta");
    expected.add("Ford");
    expected.add("Tesla");
    String expectedString = "Google -> Meta -> Ford -> Tesla";

    Graph<String, DefaultEdge> currGraph = gh.getGraph();
    DFS dfs = new DFS();
    Path result = dfs.findPath(currGraph, src: "Google", dst: "Tesla");
    assertNotNull(result);
    assertEquals(expected, Arrays.asList(result.buildPath( destination: "Tesla").split( regex: " -> ")));
    assertEquals(expectedString, result.buildPath( destination: "Tesla"));

    result = dfs.findPath(currGraph, src: "Tesla", dst: "Google");
    assertNull(result);

    Exception exception = assertThrows(Exception.class, () -> {
        dfs.findPath(currGraph, src: "Google", dst: "Google");
    });
    String expectedMessage = "Source and destination cannot be the same node";
    String actualMessage = exception.getMessage();
    assertTrue(actualMessage.contains(expectedMessage));


}
```

6. Merge conflict handling

- First merged bfs into master. It went smoothly.

- When tried to merge dfs into master after bfs, got a merge conflict. As can be seen below.



- Created Enum with DFS and BFS to resolve conflict.
  Commit for adding enum: https://github.com/shantanushishodia/cse-464-2023-sshishod/commit/acb67d7c51e1a21fbf2873ae37e59d9911281d8b
  Commit for merging dfs into master: https://github.com/shantanushishodia/cse-464-2023-sshishod/commit/7b8d4486cf40af27344d507b152347e091b3637a