**Project #4**

Name - Herman Mann

Course - CMSC 350

Date - 03/07/2022

Table of Contents

[**UML Diagram** 3](#_Toc66280093)

[**Test Plan** 4](#_Toc66280094)

[**Test Case # 1** 4](#_Toc66280095)

[**Test Case # 2** 6](#_Toc66280096)

[**Test Case # 3** 8](#_Toc66280097)

[**Test Case # 4** 10](#_Toc66280098)

[**Test Case # 5** 12](#_Toc66280099)

[**Lesson Learned** 14](#_Toc66280100)

# **UML Diagram**

Diagram

Description automatically generated

# **Test Plan**

|  |  |
| --- | --- |
| **Test Case # 1** | |
| **Description** | A graph with circular dependencies |
| **Input** | File Name = Graph\_Test\_1.txt  ClassA ClassC ClassE ClassJ  ClassB ClassD ClassG  ClassC ClassA  ClassE ClassB ClassF ClassH  ClassJ ClassB |
| **Expected Output** | Print a graph in hierarchy and parenthesized format with circular dependency |
| **Actual Output** | A graph is printed in hierarchy and parenthesized format with circular dependency |
| **Pass** | **Pass** |
| **Screenshots** |  |

|  |  |
| --- | --- |
| **Test Case # 2** | |
| **Description** | A graph with no circular dependencies |
| **Input** | File Name = Graph\_Test\_2.txt  ClassA ClassC ClassE ClassJ  ClassB ClassD ClassG  ClassC ClassI  ClassE ClassB ClassF ClassH  ClassJ ClassB |
| **Expected Output** | Print a graph in hierarchy and parenthesized format with no circular dependency |
| **Actual Output** | A graph is printed in hierarchy and parenthesized format with no circular dependency |
| **Pass** | **Pass** |
| **Screenshots** |  |

|  |  |
| --- | --- |
| **Test Case # 3** | |
| **Description** | A graph with unreachable vertices |
| **Input** | File Name = Graph\_Test\_3.txt  ClassA ClassC ClassE ClassJ  ClassB ClassD ClassG  ClassC ClassH  ClassE ClassB ClassF ClassH  ClassJ ClassB  ClassI ClassC |
| **Expected Output** | Print a graph in hierarchy and parenthesized format with unreachable vertices |
| **Actual Output** | A graph is printed in hierarchy and parenthesized format with unreachable vertices |
| **Pass** | **Pass** |
| **Screenshots** |  |

|  |  |
| --- | --- |
| **Test Case # 4** | |
| **Description** | A graph with no unreachable vertices |
| **Input** | File Name = Graph\_Test\_4.txt  ClassA ClassC ClassE ClassJ  ClassB ClassD ClassG  ClassC ClassI  ClassE ClassB ClassF ClassH  ClassJ ClassB |
| **Expected Output** | Print a graph in hierarchy and parenthesized format with no unreachable vertices |
| **Actual Output** | A graph is printed in hierarchy and parenthesized format with no unreachable vertices |
| **Pass** | **Pass** |
| **Screenshots** |  |

|  |  |
| --- | --- |
| **Test Case # 5** | |
| **Description** | A graph with circular dependencies and unreachable vertices |
| **Input** | File Name = Graph\_Test\_5.txt  ClassA ClassC ClassE ClassJ  ClassB ClassD ClassG  ClassC ClassA  ClassE ClassB ClassF ClassH  ClassJ ClassB  ClassI ClassC |
| **Expected Output** | Print a graph in hierarchy and parenthesized format with circular dependency and unreachable vertices |
| **Actual Output** | A graph is printed in hierarchy and parenthesized format with circular dependency and unreachable vertices |
| **Pass** | **Pass** |
| **Screenshots** |  |

# **Lesson Learned**

Throughout the completion of this project, I got to learn many new things about the JAVA programming language data structures and things related to them overall. First, I got to learn about creating a graph using JAVA programming syntax. This caused me to think outside the box when I had to implement this from scratch. Second, I learned how to specifically perform the Depth First Search algorithm and implementation of the graph. I also learned about the adjacency list approach onto building the entirety of the graph in the JAVA programming language. Learned about reading the graph input data files when I had to test the working of my overall program. I learned and analyzed about how to create, implement, and use the DFSAction interface which was needed for most of the project’s implementation. Moreover, I learned about using more advanced generic classes and learning about the generic object and method(s) implementation using the JAVA programming language. I learned about overloading and the overriding of methods, and finally learning about the overall designing of this graph-related directed graph project. This project has made me more advanced I believe in solving smaller problems in today’s world with the JAVA programming knowledge that I gained after implementing of this project. Also, I believe that I will be able to become a proficient Java software developer, or software engineer with my advanced knowledge of the data structures skills I have gained through completing this project.