Brett Stelly

March 28, 2018

Assessment Four

1. **Requirements Documentation**
   1. **Description of problem**

**Name:** A\* Unit Test

**Problem Statement:** The A\* implementation must pass the unit test

**Problem Specifications**: The Actual result must be the same as the Expected result

* 1. **Input information**

The user only has to launch the run\_test.bat file and the unit test will run

* 1. **Output Information**

When the unit test is finished running, the console will display the test results

* 1. **User Interface**

None

1. **System Architecture**

**astar\_for\_unit\_test.py**

Prototype: def \_\_init\_\_(self)

Arguments: None

Description: Creates an instance of the AStar class

Precondition: None

Postcondition: An instance of the AStar class is created

Prototype: def find\_current(self)

Arguments: None

Description: Finds the current node in the path, adds it to the closed list and removes it

from the open list

Precondition: There must be an instance of the AStar class

Postcondition: The current node is found, added to the closed list, and removed from open list

Prototype: def find\_path(self, start, goal, graph)

Arguments: A Node object for start, a Node object for goal, and a graph object for graph

Description: Finds the path from start node to goal node

Precondition: There must be an instance of AStar

Postcondition: The path is found and returned as a list of nodes

1. **Read Me**

In order to run the unit test, all you need to do is double click the “run\_test.bat” file and the unit test will run. When it is finished it will display the results to the console.

In order to use the unit test with my A\* implementation, I had to inject the A\* implementation into the unit test. I had to make my implementation compatible with the unit test by changing a few variable names.