To make this assignment I used multiple custom classes which can be found in the Graph.py file.

## Node Class

This class is simple and hold information such as x and y coordinates as well as an id for the node.

## Car Class

This class is used to hold the information about the car and the trailers attached to it.

## My Math Class

Used for basic computation that might be needed outside of the graph class.

## Graph Class

This class is used as the main structure to hold nodes and perform computation based on those nodes. This holds a list of all the nodes, all the edges, and methods to return distance between nodes, nearest node to a given node, successful path from origin to goal, check for success, check for if a new node is in a valid position, and methods to update the items of the graph.

## Star Graph Class

This holds useful methods for computing RRT Star algorithm that may not be necessary for the Graph class and would ultimately slow down RRT do to unnecessary computations. These methods are used to find the best node to connect the new node to.

## General Performance Metrics

Based on several random samples the algorithms tend to finish calculations in under 10 seconds with the set parameters. Several parameters can be changed, these changes may result in worse performance and more regular returns of a failed result. Changing the max edge length as well as the goal distance will result in a more optimal path as well as a smoother result, however you may also need to increase the max number of iterations allowed as increasing these will also increase the number of fail results.

## General Methodology

* Create a graph and car.
* Generate a new node and find best node connection
* Check to make sure the angle created with the new node is within +– (pi/6) radians
* Check that car would not collide with obstacle at that point with the given angle
* Add node to graph
* Check for success
* If successful connection is made, plot graph. Otherwise, continue