Odd Semester (2018)



**BINUS UNIVERSITY**

**BINUS INTERNATIONAL**

**Assignment Cover Letter**

**(Group Work****)**

|  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- |
|  | | | | |  | |  | |
| **Student information:**  **1.Krishita Sukhani 2101716773**  **2. Dean Jourdan Dumais 2101705164**  **3.Ryan Divas Tjahya 2101704413** | | | | |  | |  | |
|  |  |  |  | |
| **Course Code** | **: COMP6340** |  |  | | **Course Name** | | **: Analysis of Algorithms** | |
| **Class** | **: L3BC-BLK** |  |  | | **Name of Lecturer(s)** | | **:** 1. **MARIA SERAPHINA ASTRIANI** | |
|  |  |  |  | |  | | 2. **VINCENT ALEXANDER SELIANG** | |
| **Major** | **: CS** |  |  | |  | |  | |
| **Title of Assignment**  (if any) | : Navigation system using A\* Algorithm and Flood fill algorithm | |  |  | |  | |  | |
| **Type of Assignment**    **Submission Pattern** | **: Final Project** |  |  | |  | |  | |
| **Due Date** | **: 15-1-2019** |  |  | | **Submission Date** | | **: 15-1-2019** | |

**“Navigation system”**

1. **Description**

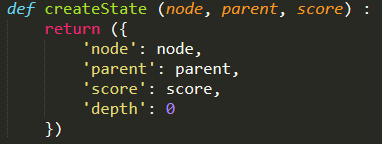
**The function of this program:**

This program is meant to help people find the fastest route towards their destination as well as depicting how A\* Algorithm works and its difference with Flood fill Algorithm

**II. Explanation of Each file**

**A\* Algorithm : ( *Astar.py* )**

**Outside the class**



helper function to create a new state

**Inside the class**

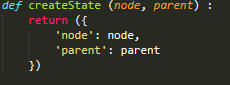
* Finds path with A\*
* Gets next top state from the priority queue
* Gets the current node's index
* Adds current state to the closed state
* Adds state index to do not check
* Gets current state's node neighbor
* Adds new state to the open priority queue
* Sorts the open stack

**Entity: ( *entity.py* )**

* Basically for the green dot that moves from one destination to another
* In charge of the target and speed of the dot

**Floodfill Algorithm: (***Floodfill.py*)

**Outside the class**



Helper function to compute new state

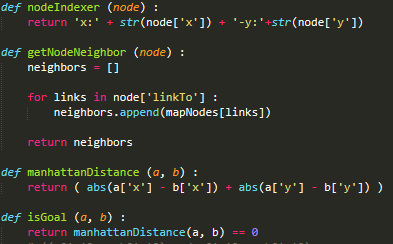
**Inside the class**

* Outside the class
* Get next top state from the stack
* Get the current node's index
* Increment node search count
* Add current state to the closed state
* Add state index to do not check
* Get current state's node neighbor
* Add new state to the open stack

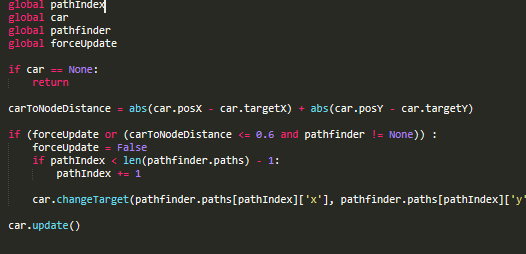
**Pathfinder: (***pathfinder.py*)

* basically loads the astar and flood fill module so that to run the pathfinder function, you only need to call 1 command
* Allows you to easily swap pathfinding algorithm and store the path that the algorithm found

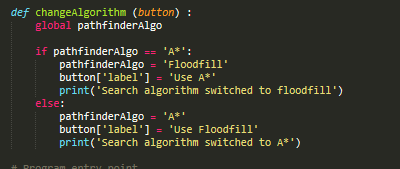
**Pathfinder main: (***pathfinder\_main.py*)



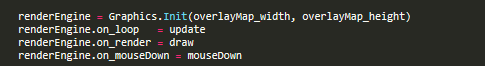
Pathfinder helper functions



Makes the car follow the path



This function allows the user to change between the default A\* algorithm to floodfill algorithm.



This will render the car

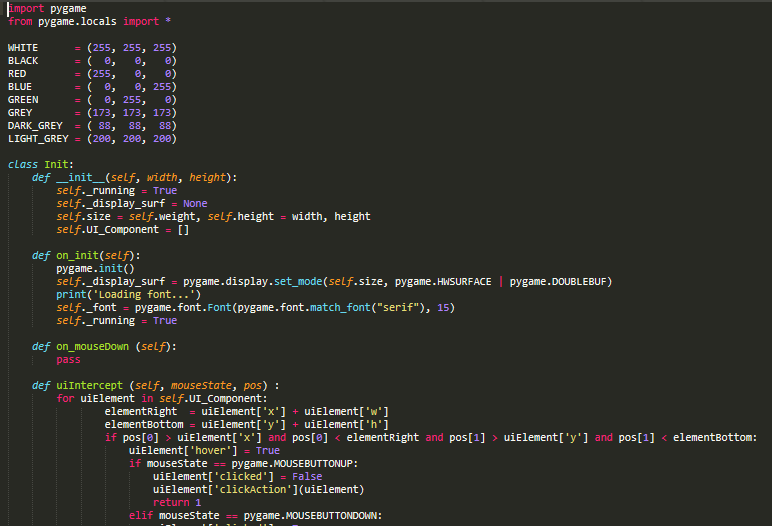


This overlays the map



This is for the button

**Graphics: (***Graphics.py*)



This file consist of the graphics that makes up the maps and also the car movements and lastly the user interface buttons that helps users switch from a\* to floodfill