**CS4006 A-Star Program Description**

Summary:

The program consists of two classes. The main class is is18225136 where the A-Star pathfinding is done, application is created and user input obtained. Then there is the node class which keeps track of relevant data for the nodes in the open and closed lists.

Program keys:

The start and goal positions can be entered (format: A-H,1-8 ) in the textbox at the bottom of the screen with enter buttons.

When the enter key is pressed the a-Star method is called and the path is drawn on screen.

When the r key is pressed the application resets the screen removing the path and start and end nodes so a new start and goal can be entered.

New start and goal positions can only be entered after a reset.

Classes:

node:

This class keeps track of the x and y co-ordinates of nodes in the open list and the x and y co-ordinates of their parent node pX and pY. It also stores their f, g and h values which represent f(x), g(x) and h(x). The class has two constructors the first of which takes an x, y, pX and pY and it sets f, g and h to 0 as they will be set later. The class also overrides the equals method of the object class so that nodes are compared based on their x and y values. This is so that if a node is already in a list it can be checked which is useful as it allows the program to check if the node on the open list has a better g value than what it currently has.

Is18225136:

A – Star:

startRow, startCol, endRow ,endCol – these are integers input by the user representing the start and goal positions

currentNode – this represents the current node being assessed by A Star it is a node object.

Open – this represents the open list it is a priorityQueue of node objects and implements a comparator so that the first node is the node with minimum f value. So that it can be removed easily and placed on the closed list. This choice does make it difficult to find a specific node (compared to an arrayList) in the list but it has advantages in deletion speed (arraylist has to shuffle nodes down but the Priority queue doesn’t ) and it has easy access to node with min f based on the comparator given unlike arraylist where it would need to be sorted after every insert.

Closed – this represents the closed list it is an arrayList of node objects as it doesn’t need to be sorted as long as the last node is the end node which will always be the case as A-Star stops when its added the end node to the closed list.

Children – this is an arrayList of child nodes of the current node. It is set to size 4 as the current node can only ever have 4 children at most. It is cleared after every loop in the A-Star method.

A-Star method:

The first thing A-Star does is add a new node representing the start node with its parent x and y as -1 as it doesn’t have a parent. Then a while loop is entered as long as the open list isn’t empty.

When inside the first node of the open list is removed as it is sorted so that this node has the minimum f value and it is stored as current node and added into the closed list. If this node is the goal node then the method is stopped with a return statement.

If not then method findOpenNodes is called to find the child nodes of the current node which are placed in the arraylist children. A star then loops through this list. If the node is in closed then the loop moves on. If not then it is assigned its f, g and h values. If the child node is on the open list already then it iterates through the open list until it finds it and compares g values and if the g value of the child node is higher than what it has on the open list than it removes the node from the open list and adds the child node with better g value. If the child isn’t on the open list it is added to it. Finally the arraylist children is cleared for the next loop through.

After it has looped through there is then a check that the goal is in the closed list. If not the application is closed and error message goal unreachable is printed out.

Obstacle Creation:

The obstacle is placed randomly and is either a T, L or I. The T position is randomly chosen but is prevented from being placed in the bottom left or right to prevent it encasing the goal or start positions. The L is prevented from being placed in the top right border for the same reasons. The I has no limitations. The positions they are placed at are set to -1 on the Grid 2d array.

Application:

The application is made up of a border pane with a grid pane pane in the center and a grid pane buttons in the bottom. The first grid pane is filled with rectangles. The first row and first column are given the text 1-8 and A-H to indicate grid positions to the user. The rest are white apart from the obstacles which are shown with black rectangles. The buttons grid pane has two labels, two buttons and two text fields. This is so the user can enter start and goal positions the labels say which is for start and end and the format the positions should be entered in.