**AI-Project-1: Informed Search (Maze)**

**Student Name:** Javid Guliyev

**Student ID:** g34987427

**Introduction:**

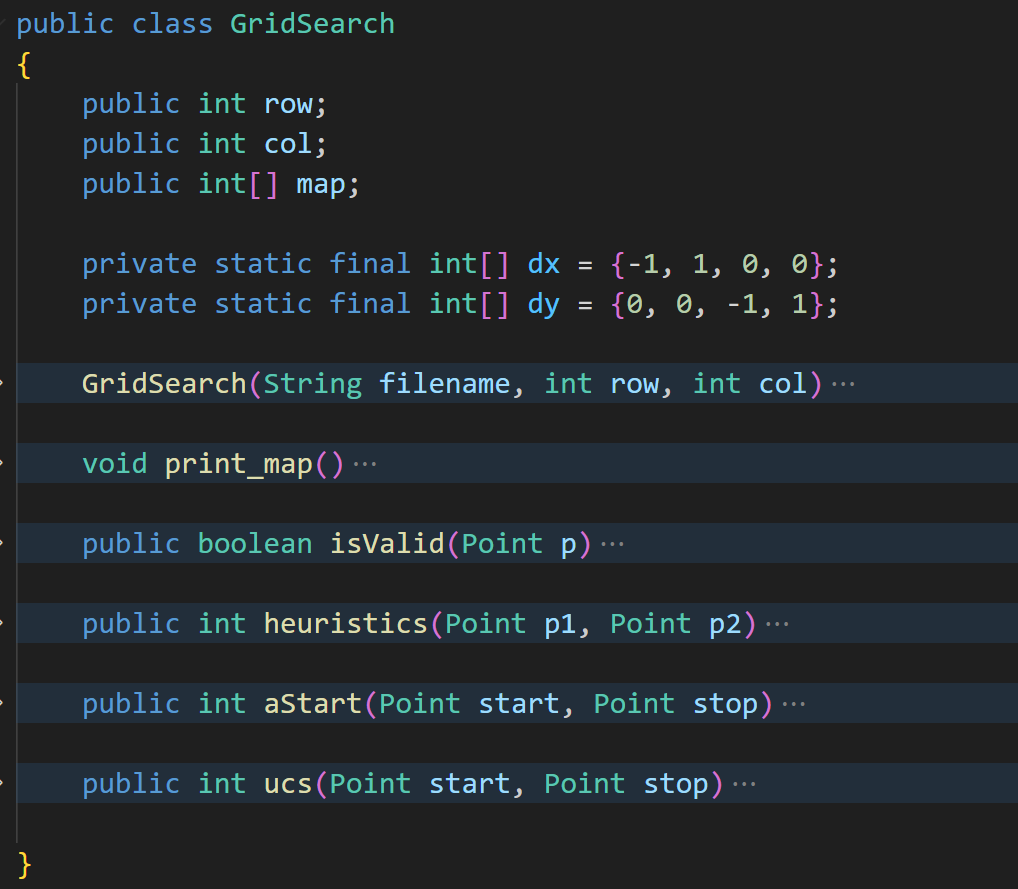
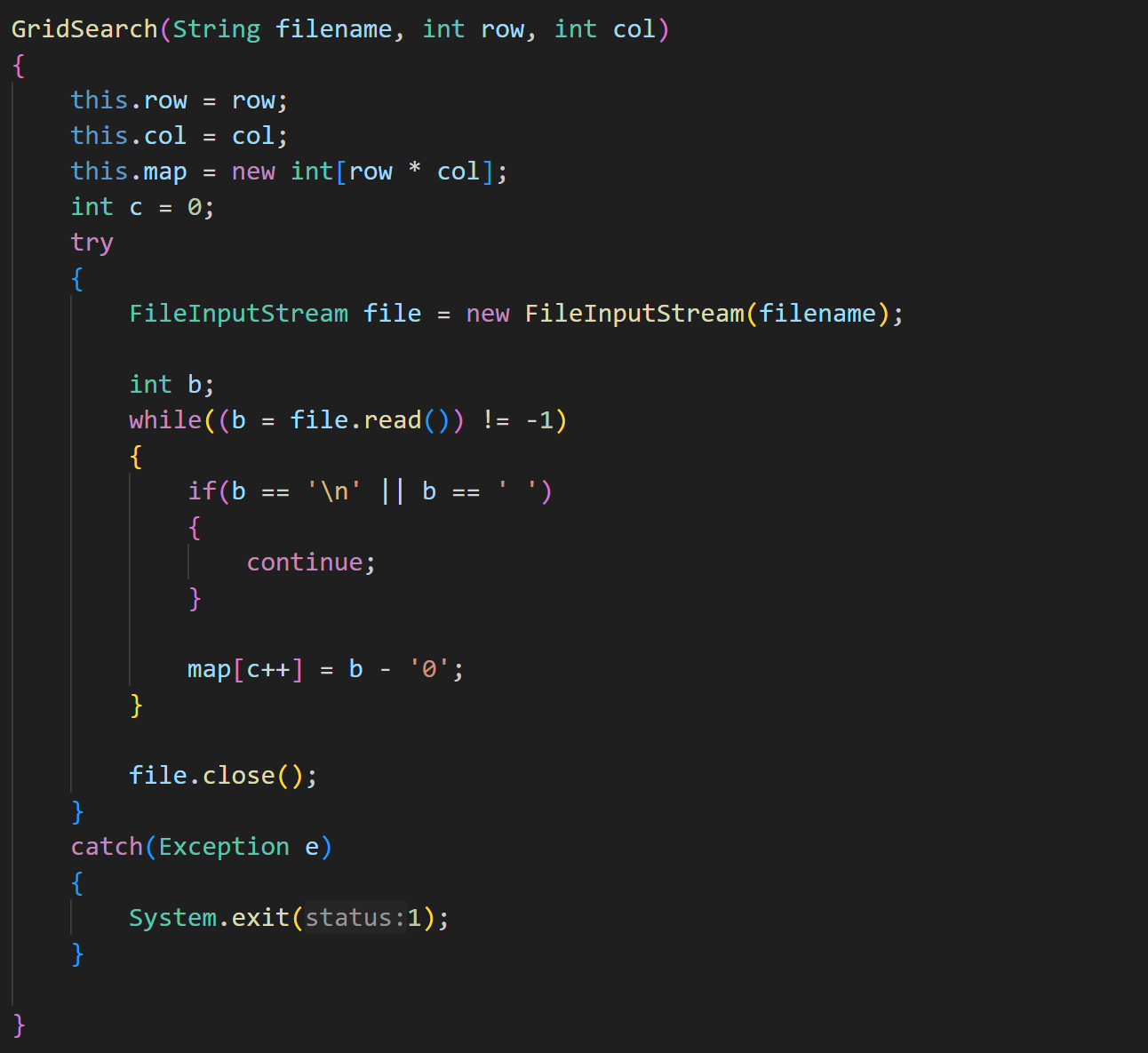
The following report serves as explanation to maze option using informed search:

Option 1 requirements states to find whether the path between points of the maze grid exists using the informed search algorithms. Maze is formed in combination of 0-1 grid which corresponds to space and obstacle respectively.

The language of choice for this task was **java** as the writer of this report is comfortable using the languages with strong typing. The complete source code is available publicly in GitHub using the following link: <https://github.com/Cavid2002/AI-Project-1>

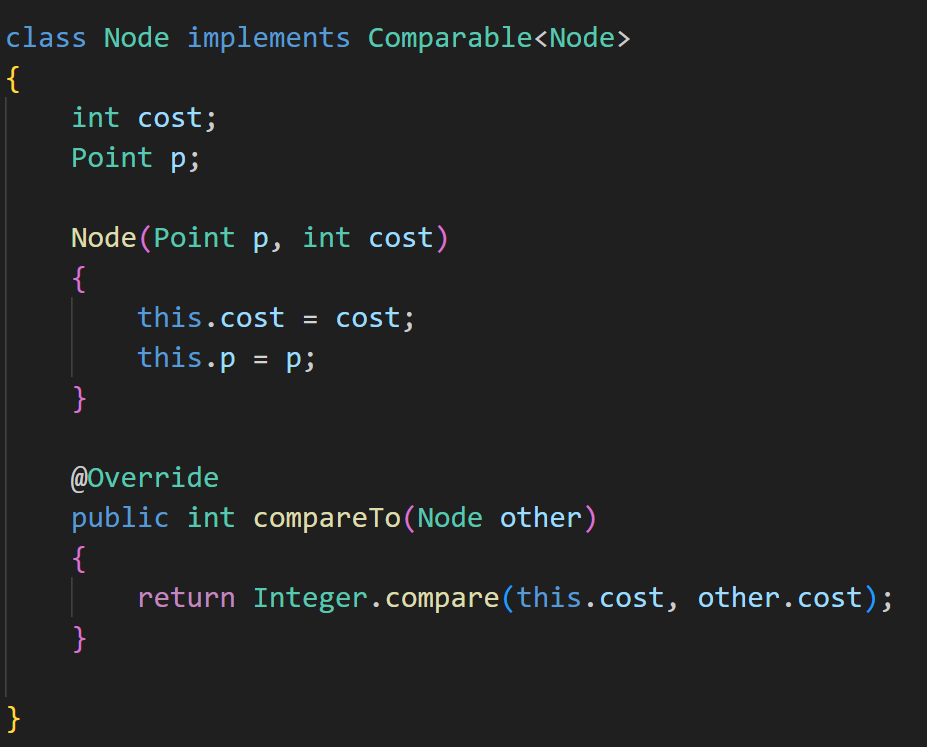
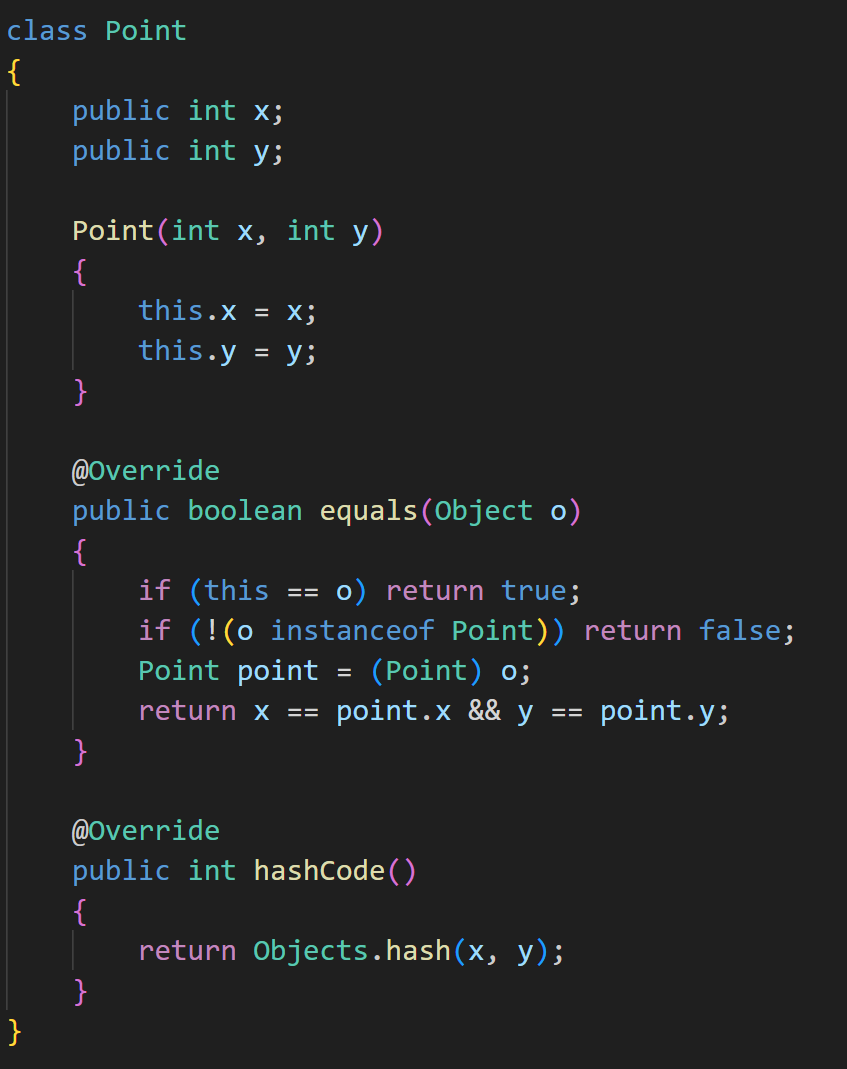
**General Structure and Classes:**

There are 2 files first of which is just a Main class that gets user input passes control to GridSearch class that is stored in second file. The Gridsearch class contains several like (rows, columns and reference to array of 1-0 ints). The purpose 2 static arrays dx, dy are for direction(up, down, left, right).



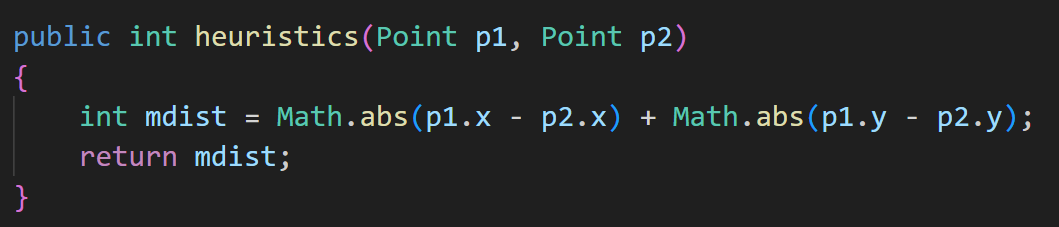
By calling the constructor of the GridSearch class picture of which was also provided above and passing the filename with dimensions of the grid alongside with it, we will populate array with zeros and ones.

There are also 2 additional helper classes first of which called the **Point class** that stores cartesian coordinates of the point. The second important helper class that worth noting is the **Node class** that contains reference to the point and the sum of cumulative cost from the start to that exact point combined with heuristics. This Class is only utilized by the priority queue that will rearrange the Node based on the least cost value. The contents of both classes are provided in images below:



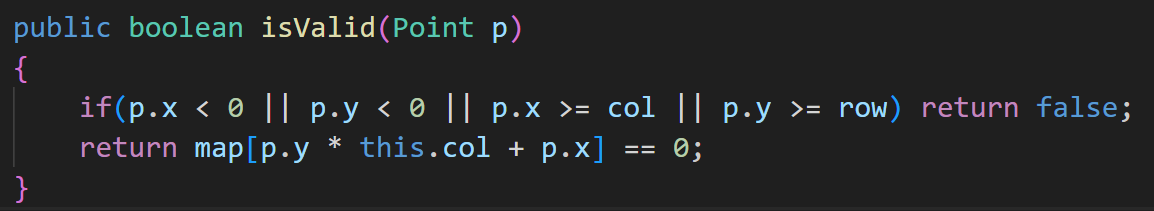
**Informed Search:**

As the informed algorithm of choice was the A\*. Unlike uninformed search(UCS) where we choose the path with the least cumulative cost without considering distance to goal node, A\* combines estimated cost from current point to goal(Like in Greedy Search) with cumulative cost from the start to current point(Like in UCS). A\* is the combination of UCS and Greedy Search or UCS search with heuristics:

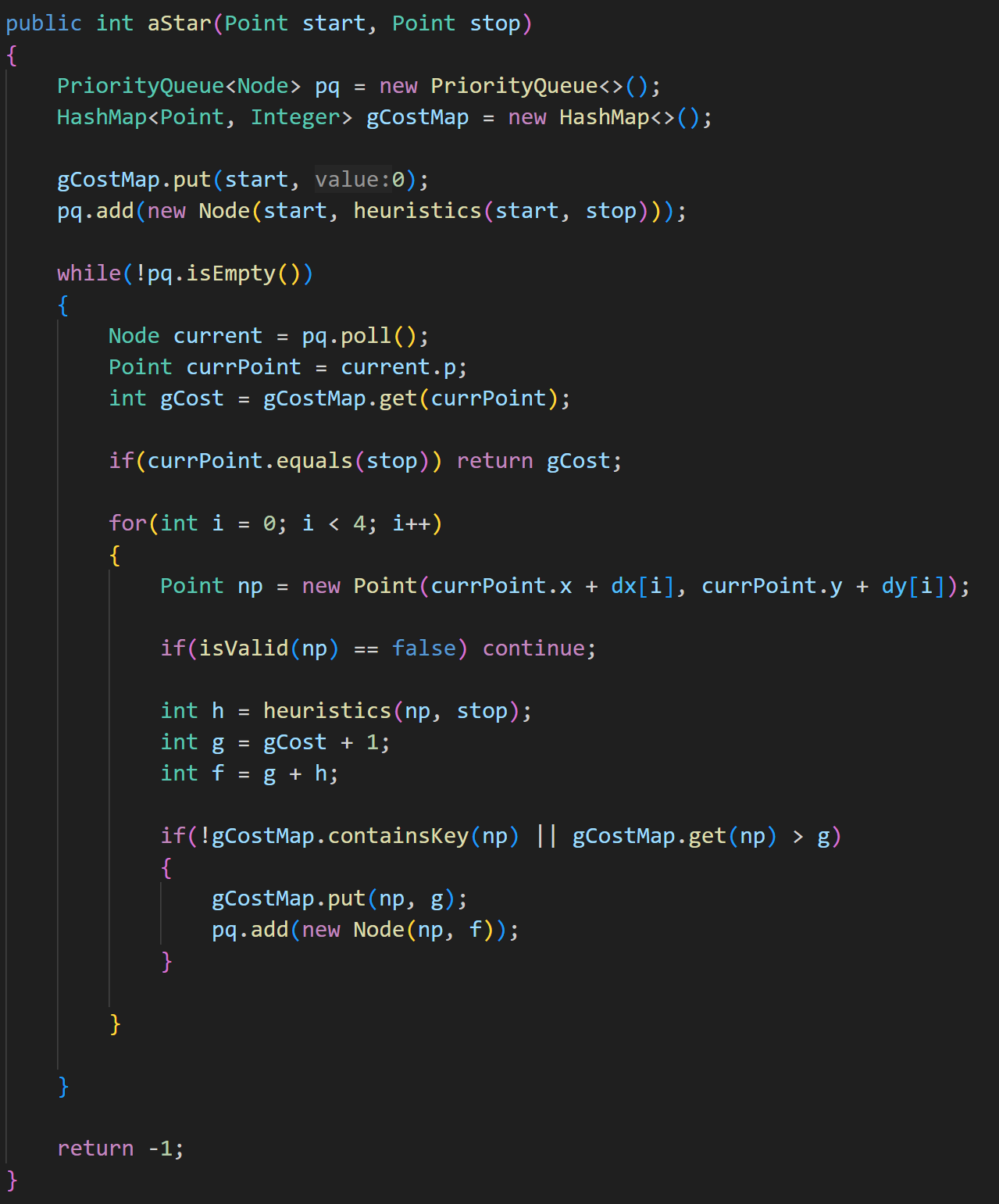


As it can be observed with image above heuristics is calculated via finding manhatan distance between two points. The result of the calculation will added with cumulative cost to obtain final cost that will be pushed onto priority queue to make the further decisions to which direction to expand.

The other method to point out before starting to explain the algorithm is following one which will check whether the given point is not an obstacle or out of bounds:



The complete algorithm for A\* in grid maze with explanation is provided below:



**Priority queue:** is responsible for rearranging the Nodes based on the least fCost value which is cumulative cost combined with heuristics: f(n) = g(n) + h(n). This queue would be responsible for providing the cheapest path to expand to.

**HashMap:** is keeping track the minimal cumulative cost(value) to reach the particular Point(key).

**The algorithm steps:**

1. Insert start point into priority queue and into hash map;
2. Repeat until queue is empty
   1. Pop the Node from the queue and extract point
   2. Check if the current point is a goal point if yes return cumulative cost to reach that point from hash map; else proceed further
   3. Do for all 4 directions(upward, downward, leftward, rightward): check if it is not an obstacle; calculate heuristics from the point to the goal; insert them into the queue if point is not visited or if cost is less than current cost;
3. Return -1 if path is not found