Assignment #1

Total Score: 100

Due date: **July 21**

Team (of max. 5 members) is allowed to complete this homework.

Objectives: Define proper heuristic functions and implement a heuristic algorithm to solve a problem

---

Problem descriptions

In this assignment, you will develop **an intelligent agent program** that can navigate through a 20 x 20 grid form of **maze** to find an **exit** from a randomly selected **starting location**. Each location on the maze can be represented by a Cartesian coordinate system with top left corner of the maze is (0,0) and bottom right corner is (19,19). The agent can move forward, backward, left and right on the maze, taking only **one step at a time**, where each step allows the agent to move from one location (coordinate point) to next location (coordinate point) forward, backward, left, and right but **no diagonal move** is allowed as it is invalid path.

There may be some dangerous objects such as **quick sands** and **spider nets** on the maze the agent should avoid. If the agent enters into any dangerous area where quick sand or spider net is, the agent will be trapped and the mission failed. A **quick sand** will capture any object on the location. For example, if a quick sand is on (7,7), only the object that enters into the location (7,7) will be captured but objects on other areas are safe. Unlike quick sand, a **spider net** will be triggered if there is any object near one step within the net. For example, if there is a spider net on (5,5), any object located on (4,5), (5,5), (6,5), (5,4), and (5,6) will trigger the net and will be captured. However, objects located on (4,4), (6,4), (4,6), and (6,6) will be safe.

Each step the agent takes will cost $1. Furthermore, certain locations on the maze require the agent to pay some **fees** to pass through in addition to the cost incurred by the agent's move. For example, if the agent has to go through locations (7,10), (7,11) and (7,11) requires $10 fee, then the total cost incurred for the agent to go through this area will be $12.

Fortunately, the agent has a powerful **sensor** that can detect any object and area including exit, fee information, and dangerous areas **within two steps away** from the current agent's location. In other words, the agent can **look ahead two steps** in **advance**. The agent can **freely move around** even **going back and forth** the same locations if necessary as long as the route is safe and doing so is beneficial in saving the overall travel cost.

The **ultimate mission for the agent** is to find and reach the **exit taking minimum cost without being captured**. Once the agent is either captured or unable to reach the exit, the mission failed.

Requirements

1. Implement the steepest-ascent or -descent **hill-climbing algorithm** or **A* algorithm** using **ipython notebook** (or **Jupyter notebook**) that allows writing and executing your program using a browser. Please see me if you or your team really want to use different programming language for this assignment.

   Note: You will earn **20 bonus points** if your A* algorithm works correctly.

   Your program will need to read an **input file**, "**input.txt**" that contains all the necessary information about the maze including starting location, exit location, fees, and locations for dangerous areas. However, the agent can look ahead ONLY two steps in advance. Therefore, you should NEVER program your agent assuming all this information is known to the agent. An "**input.txt**" file will contain the **following format** of **input data**, start location on the first line, exit on the second line, all the locations for quick sands on the third line, all the locations for spider nets on the fourth line, and all the fees on locations on the last line. Each input element on a line is separated by **one blank space**.

   (2,3)
   (19,10)
   (0,0) (5,5) (15,15)
   (10,10) (17,8)
   2:(1,3) 10:(3,3) 7:(10,15) 200:(19,9)

You need to create your own input files to test your program in various scenarios. **No test cases for testing your program will be provided**. I will use my own test case(s) to test the correctness of your program.

Note: The input file will contain all the necessary information to run the program. Therefore, your program should **NOT prompt asking any input** to enter.

2. Properly **modularize** the program clearly separating it into different modules (classes or package) including at least the **driver program** called "**Maze**" like the main function in Java or C# and required data structures; **algorithms** and related data structures; **heuristic function(s)**; a **reporting function** so that the algorithm, heuristic functions, and reporting function can be easily replaced by others if necessary for future maintenance. This type of programming and modularization is called programming using "strategy pattern". The strategy pattern allows change of algorithms or heuristic functions without requiring any change of other parts of the program.

   Document your program properly naming and commenting modules or functions so that one can easily understand their purposes.

   Note: The **main program name** should be "**maze**". In other words, if I execute "maze", your program should work.

   Note: A **heuristic function** returns a numerical value(s) for a given input that helps decision making for the agent but it should NOT include **any control logics**, e.g., moving agent forward, backward, left, or right, etc.

   If your agent **cannot move any further** because of ambiguous heuristic value, use a **random decision making approach** or other heuristics to make it move into certain direction, of course to the safe direction without being captured. If your agent is unable to reach to the given exit, your program will be considered as incomplete.

3. The **reporting function** should produce an **output file** "**output.txt**" that print a trace of the agent's movement, e.g., a series of the coordinate points the agent has moved. Once it finds an exit, print the **total cost** and **total number of steps** taken for the travel. The "**output.txt**" should have the **following format**:

   Total cost: $100
   Total number of steps taken: 30
   Traveled route: (2,3), (2,4), ..., (19, 10)

4. **Write a brief report** in Word format including (a) the name(s) and contact email addresses of the developer(s); the percentage contribution to this assignment if the assignment was completed by a team. If a team cannot reach a consensus on the individual contribution, include the individual's claimed percent contribution with a brief description on specific tasks performed (b) a brief explanation about the implemented algorithm (specifying hill-climbing or A*) and how your program is structured (c) a pseudo code for your heuristic function and a brief explanation about the strategies used, (e) optionally any discussion, comment, or reference you want to add related to this assignment.

   Warning: Although the code reuse from existing source codes on the Internet is allowed, copying the code from other person or team in this class is strictly prohibited. Any one or team violated this policy will receive ZERO score for this assignment.

How to submit this assignment

Zip both the report and program files into **ONE file** and submit the zipped file **to Titanium** by the due date. I strongly recommend you to write the report in **Word format** so that I can provide feedbacks directly in the report when necessary. If a PDF format of report is submitted, no feedback will be provided.

Grading policy

The full credit will be given to the team or individual that fully followed all the instructions meeting all the requirements. Otherwise, some points will be deducted.