# CSCI-561 - Fall 2018 - Foundations of Artificial Intelligence Homework 1

**Due September 17, 2018 23:59:59**

**Description**

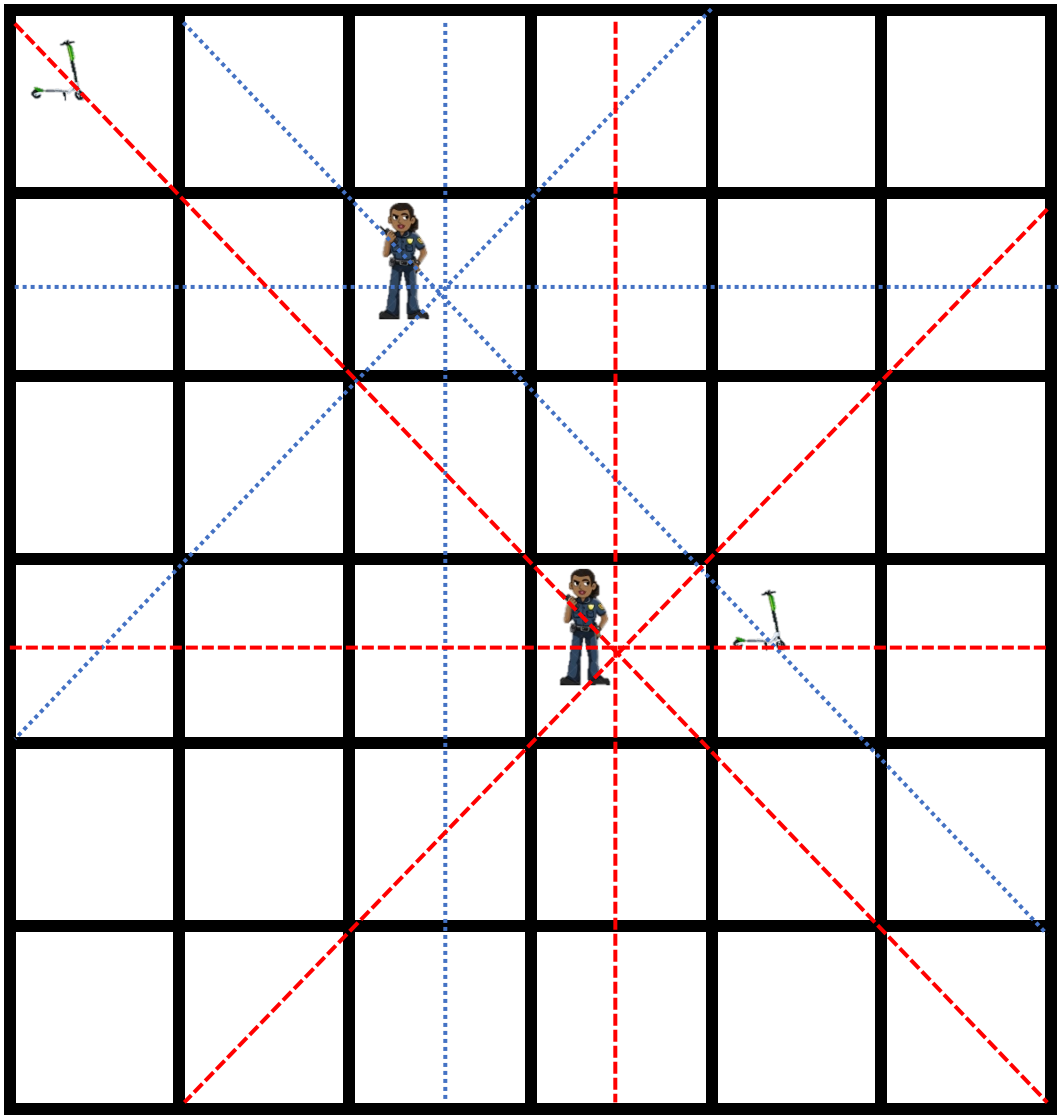
You are helping the LA Dept of Transportation (LADOT) to develop a pilot scooter program for LA. There are a limited number of police officers available to monitor and address issues that may arise from scooters, ranging from traffic and safety violations to accidents with cars, bikes, other scooters and pedestrians. The scooter companies have given LADOT access to scooter routes over the course of one day. In order to maximize the scooter activity monitored by the officers, you will take as input the route information, the monitored city area dimensions, and the number of officers available to then generate the best placement of the officers. The officers can only be in one place for one day, and there can only be one officer on each street. When an officer and scooter are at the same location at the same time, the officer is able to address a safety issue, and one "Activity point" is gained. **The goal is to place the officers in locations that do not conflict with each other, while maximizing the total "Activity points" for the day (12 time steps in a day).** The problem follows these rules:

- Officers cannot be in same square, same row, same column, or along the same diagonal. (Think of queens on a chess board)
- Officers cannot move.

- Activity points are collected at each time step t when officers are in same square as scooters. One point per each scooter.
- The grid coordinate system will be indexed starting from the top-left corner. An example of a 5 by 5 grid is given below with each cell's coordinates:

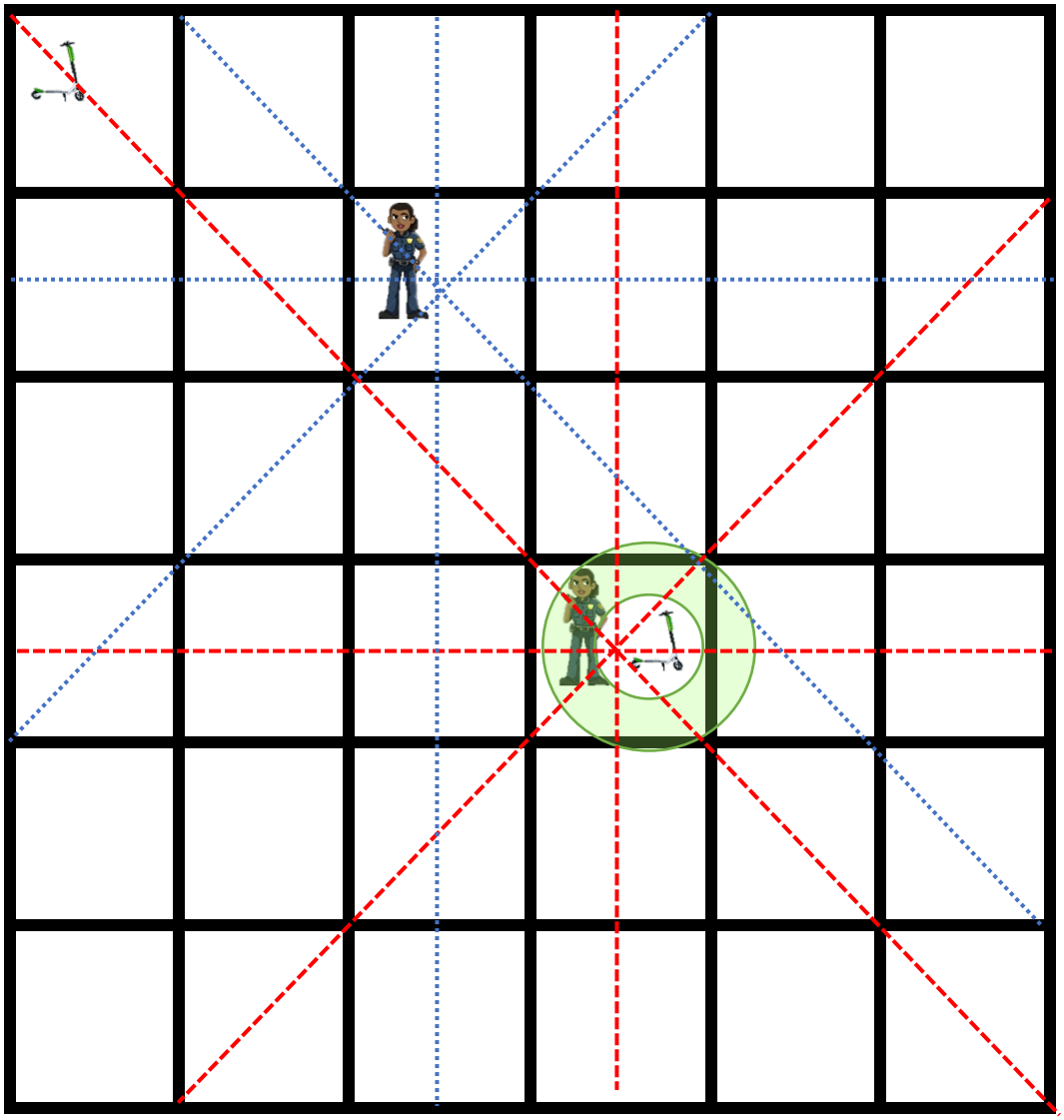
| | | | | |
|---|---|---|---|---|
| 0,0 | 1,0 | 2,0 | 3,0 | 4,0 |
| 0,1 | 1,1 | 2,1 | 3,1 | 4,1 |
| 0,2 | 1,2 | 2,2 | 3,2 | 4,2 |
| 0,3 | 1,3 | 2,3 | 3,3 | 4,3 |
| 0,4 | 1,4 | 2,4 | 3,4 | 4,4 |

**T = 1**



The two police officers have been placed and are unable to move. The red and blue lines show the limitations on placing officers; no officer may be placed on the same row, column, or diagonal as another officer. Since no scooters are in the same square as either of the officers, no value is gained at T=1.

**T = 2**



At T=2, the scooters have both moved one square. Since a scooter has moved onto a square with an officer, 1 Activity point is gained at T=2. This process continues at T steps forward, with the officers keeping their positions.

**Input:** The file `input.txt` in the current directory of your program will be formatted as follows:

First line: strictly positive 32-bit integer ***n***, the width and height of the ***n*** x ***n*** city area, ***n*** <= 15.

Second line: strictly positive 32-bit integer ***p***, the number of police officers

Third line: strictly positive 32-bit integer ***s***, the number of scooters

Next ***s****12 lines: the list of scooter x,y coordinates over time, separated with the End-of-line character LF. With ***s*** scooters and 12 timesteps in a day, this results in 12 coordinates per scooter.

Output:

Max activity points: strictly positive 32-bit integer ***m***

**Guidelines**

This is a programming assignment. You will be provided sample inputs and outputs. Please understand that the goal of the samples is to check that you can correctly parse the problem definitions, and generate a correctly formatted output. The samples are very simple and it should not be assumed that if your program works on the samples it will work on all test cases. There will be more complex test cases and it is ***your task to make sure that your program will work correctly on any valid input***. You are encouraged to try your own test cases to check how your program would behave in some complex special case that you might think of. Since **each homework is checked via an automated A.I. script**, your output should match the specified format *exactly*. Failure to do so will most certainly cost points. The output format is simple and examples are provided. You should upload and test your code on vocareum.com, and you will submit it there.

**Grading**

Your code will be tested as follows: Your program must not require any command-line argument. It should read a text file called "input.txt" in the current directory that contains a problem definition. It should write a file "output.txt" with your solution to the same current directory. Format for input.txt and output.txt is specified below. End-of-line character is LF (since Vocareum is a Unix system and follows the Unix convention).

The grading A.I. script will

- Create an input.txt file, delete any old output.txt file.
- Run your code.
- Test your output.txt file

**Academic Honesty and Integrity**

All homework material is checked vigorously for dishonesty using several methods. All detected violations of academic honesty are forwarded to the Office of Student Judicial Affairs. To be safe you are urged to err on the side of caution. Do not copy work from another student or off the web. Keep in mind that sanctions for dishonesty are reflected in *your permanent record* and can negatively impact your future success. As a general guide:

- **Do not copy** code or written material from another student. Even single lines of code should not be copied.
  **Do not collaborate** on this assignment. The assignment is to be solved individually.
  **Do not copy** code off the web. This is easier to detect than you may think.
- **Do not share** any custom test cases you may create to check your program's behavior in more complex scenarios than the simplistic ones considered below.
  **Do not copy** code from past students. We keep copies of past work to check for this.
- **Do** ask the professor or TA if you are unsure about whether certain actions constitute dishonesty. It is better to be safe than sorry.

**Homework Rules**

1. Use Python 2.7 to implement your homework assignment. You are allowed to use standard libraries only. You have to implement any other functions or methods by yourself.

2. Create a file named "hw1cs561f2018.py". When you submit the homework on labs.vocareum.com, the following commands will be executed:

   python hw1cs561f2018.py

3. Create a file named "output.txt" and print its output there. For each test case, the grading script will put an "input.txt" file in your work folder, runs your program (which reads "input.txt"), and check the "output.txt" file generated by your code. The grading script will replace the files automatically, so you do NOT need to do anything for that part.

4. Homework must be submitted through Vocareum. Please only upload your code to the "/work" directory. **Don't create any subfolder or upload any other files**. Please refer to http://help.vocareum.com/article/30-getting-started-students to get started with Vocareum.

5. Your program must handle all test cases within a maximum runtime of **3 minutes** per test case on Vocareum.

6. It is recommended to submit your program 24 hours ahead of the deadline to avoid any submission issues on Vocareum. Late submissions will not be graded.

**Helpful Hints:**

1. **We will not give unsolvable inputs.** This means that we won't give more officers than can be assigned without conflicts, or any irregular inputs that don't conform to the format we've described in this document.
2. **All officers must be allocated.**
3. **Some inputs may have more than one valid solution.** We will accept all solutions that achieve the maximum possible Activity points for the given input.
4. **Each scooter route for a day consists of 12 timesteps.** For example, if a testcase has 2 scooters the input file will contain a list of 24 scooter positions.
5. **Think about representing the problem.**
    a. What is a good representation of states and operators?
    b. While the scooters may be moving at each timestep, the officers cannot move. How can you use this to simplify the problem representation?
    c. How will you evaluate the "score" of a state?
6. **Think about complexity.**
    a. How does grid size affect branching factor?
    b. How does the number of officers affect search tree depth?
    c. How can you use the input parameters to determine which algorithm will be able to generate a solution within 3 minutes?

**Sample input and output files are posted on DEN.**