UNIVERSITY OF THE WITWATERSRAND, JOHANNESBURG

School of Computer Science and Applied Mathematics
University of the Witwatersrand, Johannesburg

COMS3005A: Advanced Analysis Algorithms
Assignment 2

Sheng Yan Lim

Semester II, 2019

1 Aim

This assignment is intended to give you some exposure to the experimental nature of Computer Science - specifically the concept of measuring the performance of an algorithm and relating these measurements to the theoretical analysis of the said algorithm.

2 Group Registration

You do not have to register a new group if you have registered for assignment 1. Otherwise, visit Moodle to register your group. A reminder that the maximum number of students per group is 2.

3 Peg Solitaire - Backtracking

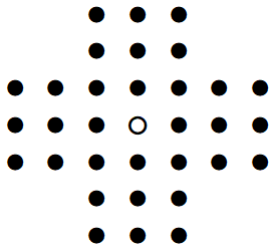
In this assignment, you are required to study the backtracking algorithm for puzzle solving, perform theoretical and empirical analysis by implementing the algorithm to solve *Peg Solitaire*. Note that the *English* board configuration will be used in this assignment.

4 Instructions

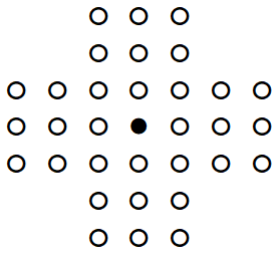
4.1 Task 1

Deadline: 3rd October 2019 at 23:59

You are required to implement the backtracking algorithm to solve a game of peg solitaire such that the initial configuration is transformed to the final configuration. The standard initial and final English boards are as follows:



(a) Standard Initial English Peg Solitaire Board Configuration



(b) Standard Final English Peg Solitaire Board Configuration

where a filled hole contains a peg, empty otherwise. A configuration is a board with some subset of peg filled holes.

To transform the initial board configuration to the final board configuration, the player would perform *jumps*. A jump involves three consecutive horizontal or vertical holes and it will be considered as a **legal** jump if the first and second holes are filled and the third hole is empty. This jump transforms the configuration by removing two pegs from the first two holes and filling the third hole with one peg.

A formal representation of a jump can be defined as a tuple (i, j, dir) where i and j is the position of the first hole and dir is the direction of the jump *UP, DOWN, LEFT, RIGHT*.

Input

Two boards representing the initial and final configuration are given. Each board is a 7×7 matrix where 0 indicates an invalid hole, 1 indicates a filled hole and 2 indicates an empty hole.

Output

If there exist a solution to transform the initial configuration board to the final configuration board, output the jump sequence (represented by the tuple described above), otherwise output `no solution`.

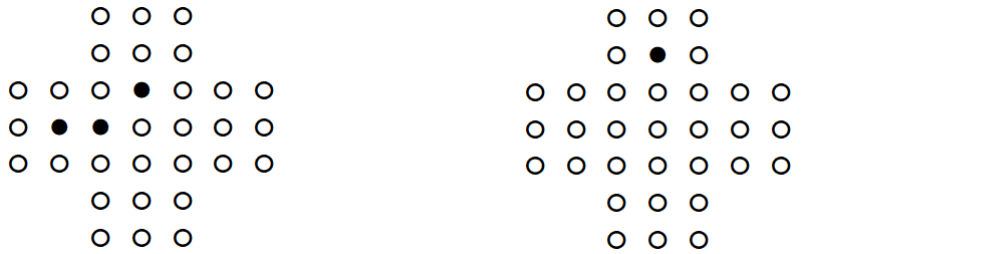
Sample Input

```
0 0 2 2 2 0 0
0 0 2 2 2 0 0
2 2 2 1 2 2 2
2 1 1 2 2 2 2
2 2 2 2 2 2 2
0 0 2 2 2 0 0
0 0 2 2 2 0 0
0 0 2 2 2 0 0
0 0 2 1 2 0 0
2 2 2 2 2 2 2
2 2 2 2 2 2 2
2 2 2 2 2 2 2
0 0 2 2 2 0 0
0 0 2 2 2 0 0
```

Sample Output

```
3,1,RIGHT
3,3,UP
```

Sample



(a) Sample Initial Board Configuration (b) Sample Final Board Configuration

4.2 Task 2

Deadline: 13th October 2019 at 23:59

The questions will be posted on Moodle.

5 Submissions

For task 1, you are required to submit a zipped file containing the main java program and any other relevant files. The name of the main java program has to be *Program.java* not *Main.java* in order for the marker to mark correctly. There are no restrictions on naming the zipped file.

For task 2, you are required to answer the questions directly on Moodle.

Note that using code downloaded from the web is a form of **plagiarism**. Refer to the general course outline for more information

The backtracking algorithm is examinable in the final exam.