

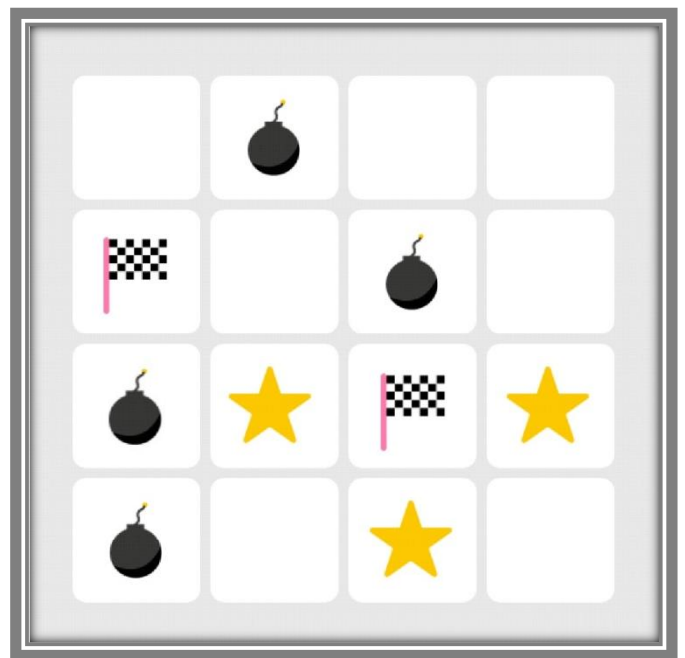
## Artificial Intelligence Course Assignment 1 – Part 2

---

### Path to Safety (Game): [2 marks]

#### ❖ About the game:

Given an  $M \times N$  board, the goal of the game is to move from the 'Start' square and safely reach the 'Finish' square. In order to "safely" reach the finish square, you can't pass through any squares containing bombs. You should also try to collect as many stars as possible in your path. The game has several levels; each level is different in configuration and some levels have more than one path. For example, the level shown in the opposite figure has 3 possible safe paths. The shortest path contains 1 star while the longest contains 3.



#### ❖ What you are required to do:

You will be given a prolog source file containing the level data (like the file attached with this document) and you are required to make a prolog program that produces a list of moves for each possible safe path from the start to the finish square and states the number of stars that can be collected in each path. Your prolog source file should only contain the predicates and rules needed by the solver; however, it shouldn't define any facts about the level configuration.

### ❖ A sample test case:

This test case shows the sample input that should be entered and what the output should look like. Notice that this output is the output produced when we run the solver on the level shown in the first figure.

```
?- load_files("C:/ Level3.pl"). %the level file (test case) given by your TA
true
?- load_files("C:/ Solver.pl"). %the file containing your program
true
?- play(Moves,Stars). %the user will only enter this predicate (play/2)
Moves = [right, down, down, right, up],
Stars = 2 ;
Moves = [right, down, down, right, right, up, left],
Stars = 3 ;
Moves = [right, down, right],
Stars = 1 ;
false.
```

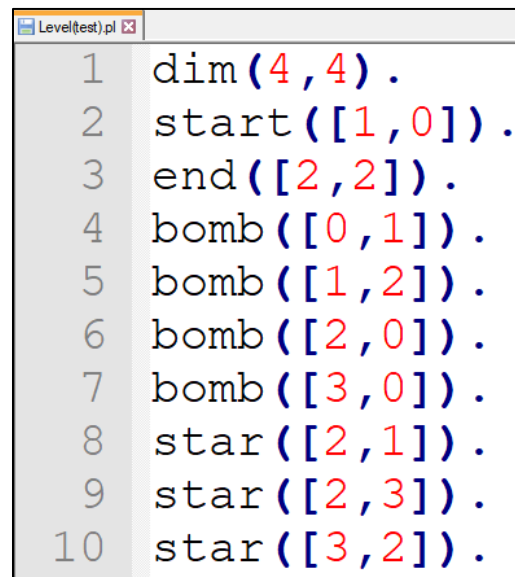
### ❖ The structure of the input file:

You will find the level file (for the level in the first figure) attached with this document. All level files will have the same structure shown in the opposite figure. A level file contains the following predicates:

- **dim/2**: the dimensions of the board.
- **start/1**: the start square.
- **end/1**: the finish square.
- **bomb/1**: the square that contains a bomb.
- **star/1**: the square that contains a star.

You will notice that each square is represented by a list containing 2 elements;

the index of the row and the index of the column in which this square is located. Level files will have the **same structure** (same predicate names, number of arguments), however they will have **different data** (different dimensions, start/end locations, different numbers and locations of bombs and stars). **You are not allowed to change the structure of the level file**; you should adjust your program to work with this structure.



```
1 dim(4,4).
2 start([1,0]).
3 end([2,2]).
4 bomb([0,1]).
5 bomb([1,2]).
6 bomb([2,0]).
7 bomb([3,0]).
8 star([2,1]).
9 star([2,3]).
10 star([3,2]).
```

❖ **Important notes:** *(Please read these notes carefully to avoid losing grades)*

- The user will interact with your program through the **predicate “play” which takes exactly 2 arguments**; a variable for the list of moves and another for the number of stars.
  - Various level files will be used to test your program, so **don’t put the level facts (dim, start, ...)** in your prolog source file.
  - Make sure your program **handles cycles** (i.e. don’t let your program get stuck by returning to the same square repeatedly).
  - The output of your program should look like the **output in the sample test case**.
  - For this problem, **don’t use built-in predicates** like findall, aggregate\_all, foreach, ...
- 

**Remember:** *(Please read these notes carefully to avoid losing grades)*

- In this assignment, you will work in teams. **The minimum number of students in a team is 2 and the maximum is 3.**
- Please submit **one compressed folder** containing your **(.pl)** file. The folder name should follow this structure: **ID1\_ID2\_ID3\_GX,Y**
- **Cheating students will take -6** and no excuses will be accepted. If you have any problems during the submission, contact your TA but don’t, under any circumstances, give your code to your friends.

**Grading Criteria:**

<i>Problem 1</i>	
Suitable predicates & rules	<b>1</b>
Output (moves list & number of stars)	<b>1</b>

*Good luck*