

## Reasoning and Learning

**Assignment 2:** (10% of the total assessment in this subject)

**Submission:** Submit before 5:30pm 1 December via the submission link for assignment 2 on the subjects' Moodle site.

**One ZIP file is to be submitted.** The ZIP file is to contain: two source code files named:

- `main.pl`
- `main.c` or `main.c++` or `main.cpp`

The file `main.pl` is to contain your answer to question1 whereas the C or C++ file is to contain your answer to question2.

Important note: This assignment is individual work. **Late submission and plagiarism will result in a penalty** (refer to subject outline for details).

### Question 1

(6 marks)

You are facing the following problem: You are given a robot and your task is to “guide” the robot through a 2-dimensional maze such that the robot can reach a desired goal state. Assume that the maze is defined as a regular 2-dimensional grid with discrete grid points. Some of the grid points can be visited by the robot while other grid points (such as those denoting walls and other obstacles) are off limit. Lets call the set of grid points that a robot can visit “states”.

Assume that the following is given:

- A list of valid states (spaces that the robot can visit). It can be assumed that the list is complete and hence, the list defines the maze.
- A starting state. This defines the starting position of the robot.
- A goal state which defines the target state for the robot.

Your task is to write a rule called `robby` in Prolog. The rule is to find **the shortest path** from the given start state to the given goal state. The robot can take one step at a time and permitted are the moves up, left, right, and down. Diagonal movements are not allowed. This means that the robot can only move to states which are directly adjacent to a current state.


Your Prolog program is to read the list of states, the goal state, and the target state from a database containing “facts” called `DB.pl`.

You are given the files `DB.pl` and `main_template.pl`. The file `DB.pl` contains a description of the maze which is shown on the next page. Your task is to:

- Extend the content of `main_template.pl` file such that it uses the facts as defined in `DB.pl` to compute the shortest path from the given start state to a given goal state. Your program should work correctly for any other maze, too.
- Use SWI Prolog to solve the task. Do not make use of additional libraries (your code must be stand-alone, without further dependencies).
- Ensure that your solution will work for other mazes, too. Your code will be tested on different versions of `DB.pl`. Some of these mazes may not have a solution or may have more than one shortest paths (i.e. several solutions of the same length). Your program should produce a correct response (i.e. list all shortest paths, or should “fail” if there is no solution).

For example, for the maze defined in the provided `DB.pl` your code should produce the following sequence as output:

$s(7,3), s(7,4), s(7,5), s(7,6), s(6,6), s(5,6), s(4,6), s(4,5), s(4,4), s(5,4)$   
 Note that some of the states in this maze are out of reach for this robot or, if the robot had started from  $s(8,1)$  then there would have been no solution.

	x1	x2	x3	x4	x5	x6	x7	x8
y7								
y6								
y5								
y4					<u>G</u>			
y3								
y2								
y1								

## Question 2

(4 marks)

Implement the value iteration algorithm for MDP which computes the solution to the situation shown below. You may write your code in either C or C++. Your code must be implemented as a self-contained single source code file which does not require any additional libraries during compilation, does not require any additional data files during run-time, and does not expect any user inputs. For each value of  $k$ , your program is to print (to the screen) the reward vector  $J$ . Your program is to terminate when convergence is observed (use  $\epsilon=0.0001$ ). For each time step  $k$  print the optimal policy. Use the discount factor  $\lambda = 0.9$ . Your name and student number should be in the comment header of the source code file.

