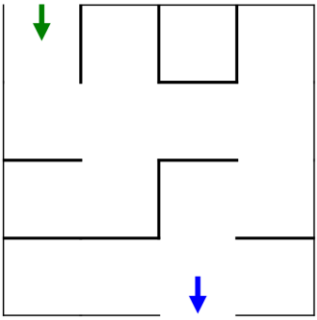
Tutorial 3 - Dynamic programming

Exercise N°1

We consider the problem of solving a Maze, as illustrated in the figure below. The agent begins at the start cell and must find its way to the exit cell by navigating through the maze.



- The maze is represented as a 2D grid. Each cell in the maze has four walls, one on each side (top, right, bottom, left). These walls can either be open or closed. Two adjacent cells share the same wall.
  - The start state is located at the (0, 0) cell
  - The exit state is located in the (3, 2) cell.
- To reach the exit cell, the agent moves through the maze in a series of steps. The agent wins if it reaches the exit cell and fails if it exceeds the time limit, which is set to 16 steps.
- The agent has four possible actions: move up, move right, move down, and move left, with a discount factor $\gamma = 0.9$.
- the agent can only transition between cells if there is no wall blocking the way.
  - When the agent takes an action, it does not always succeed in moving in the intended direction. With probability 0.8, the agent moves in the intended direction (one cell in the respective direction). With probability 0.2, the agent either moves in a opposite direction or stays in the same cell if there is a wall in the opposite direction.
  - If the agent takes a non-permitted action (e.g. colliding with a wall), it remains in the same call

- The agent receives a reward of (+50) for reaching the exit cell; a penalty of (-10) for colliding with a wall; a penalty of (-1) for a move that did not result in finding the exit cell.

Question

1. Formalize the problem as a Markov decision process (MDP).
2. what the objective of penalizing the agent for executing no permitted action and for executing an action that not resulted in exit cell
3. What is the objective of penalizing the agent for executing no permitted action or taking an action that does not lead to the exit cell.
4. We consider the following policies:

cell	Up	Down	Left	right
(0,0)		1		
(0,1)		1		
(0,2)		1		
(0,3)		1		
(1,0)				1
(1,1)				1
(1,2)				1
(1,3)		1		
(2,0)				1
(2,1)	1			
(2,2)		1		
(2,3)			1	
(3,0)				1
(3,1)				1
(3,2)		1		
(3,3)			1	

cell	Up	Down	Left	right
(0,0)		0,8	0.2	
(0,1)	0.2	0.8		
(0,2)	0.2	0.8		
(0,3)		0.7		0.3
(1,0)			0.1	0.9
(1,1)			0.2	0.8
(1,2)			0.1	0.9
(1,3)	0.2	0.8		
(2,0)			0.5	0.5
(2,1)	1			
(2,2)	0.1	0.9		
(2,3)			1	
(3,0)			0.1	0.9
(3,1)			0.1	0.9
(3,2)	0.2	0.8		
(3,3)			1	

a. Apply the policy evaluation algorithm to evaluate $\pi 1$ and $\pi 2$.

b. Apply the policy improvement algorithm, by starting with π1, then with π2.

5. Apply the policy iteration algorithm to calculate the optimal policy.
6. Apply the value iteration algorithm to calculate the optimal value and then derive the optimal policy.

Exercise 2 (Additional)

We consider a system of two states s_1 and s_2. The system transits to any of the two states by taking actions in a set of two actions a_1 and a_2, with a discount factor $\gamma = 0.9$. The transition probabailities and the resulted rewards are given as follows.

s	a	$s0$	$p(s^0\|s,a)$	$r(s^0\|s,a)$
s_1	a_1	s_1	0.7	-1
s_1	a_1	s_2	0.3	1
s_1	a_2	s_1	0.8	$-1/2$
s_1	a_2	s_2	0.2	$3/2$
s_2	a_1	s_1	0.9	$-2/3$
s_2	a_1	s_2	0.1	$5/4$
s_2	a_2	s_1	0.5	-1
s_2	a_2	s_2	0.5	1

We consider the following policies:

- π_1 defined: $\pi_1(s_1) = \pi_1(s_2) = a_1$.
- π_2 defined: $\pi_2(s_1) = (0.2, 0.8)$ and $\pi_2(s_2) = (0.5, 0.5)$.

1. Apply the policy evaluation algorithm to evaluate π_1 and π_2.
2. Apply the policy improvement algorithm, by starting with π_1, then with π_2.
3. Apply the policy iteration algorithm to calculate the optimal policy.
4. Apply the value iteration algorithm to calculate the optimal value and then derive the optimal policy.