# E1 277 Reinforcement Learning
# Programming Assignment 2

March 2023

## 1 Problem Statement

A Treasure Hunter agent is trapped in an $m \times n$ island from which it cannot escape. Its task is to dig for buried treasures that are spread across different locations on the island. The island also contains several traps which always incur damage to the agent whenever visited. At any location on the island (indicated by a cell), the agent can take possible actions among $left$, $right$, $up$, and $down$, but not all actions are valid in every cell, i.e., if the agent tries to go outside the island, it is considered as an invalid action. On visiting a cell, the agent tries to dig for a possible treasure which causes a small digging cost. If the agent finds the buried treasure, it earns a positive reward, whereas if the cell turns out to be a trap, it receives a negative reward. The cells containing treasure are assumed to have an infinite amount of buried treasure.
Your task is to implement the Q-Learning algorithm for the above problem.

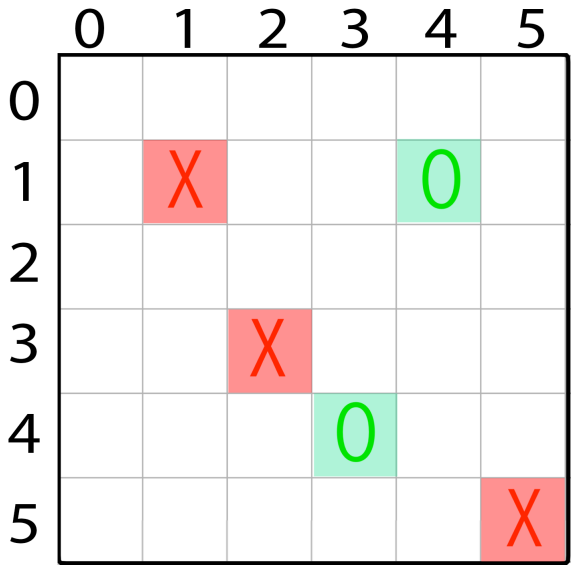


Figure 1: A sample island. The $X$-marked red cells indicate traps, and 0 marked green cells indicate buried treasure.

- For the above island, indexing starts from 0 (starting from left), At cell (0,5), the agent has only two valid actions: $left$ and $down$. If the agent decides to move $down$, he goes to (1,5) and will get a reward of -1(digging cost).
- At state (1,5), valid actions are $up$, $down$, and $left$. if the agent decides to move left, it goes to (1,4), getting a reward of +10(treasure reward).

- At (1,2) agent can take all four actions. On taking the action $left$, it goes to the state (1,1) and receives a reward -5(trap).

## 2 Implementation Details

**Your task is to implement the function *q_learning*()** that returns a numpy array $Q\_Table$ in which Q_Table[ rowIndex, colIndex, actionIndex ] is Q-value of state ( rowIndex, colIndex ) and the action corresponding to the state's valid action $actionIndex$.
The function ***reset*()** of Environment Class initializes the start state of the agent in the given Environment.
The function ***step*()** of Environment Class takes as an input an action index ( $up : 0, down : 1, right : 2, left : 3$) that can be taken in the current state and returns the next state and its corresponding reward. Invalid Actions that will take the agent out of the Environment has to be taken care of while implementing Q Learning.
You are also required to implement step size $\alpha$ as follows: At the $n^{th}$ iteration, $Visit\_n(s, a)$ is the number of times the state action pair (s,a) is seen. So the step size $\alpha = 1/Visit\_n(s, a)$.
You are required to run the code for $maxIter$ number of times, and each iteration should run for $maxTimesteps$ time steps, and do not modify these values.
The values returned by $Q\_Table$ will be checked upto 5 decimal places.

## 3 Instructions for Submission

- Write your code only in the space indicated in the template file $RL\_Assignment2.py$.
- Try not to hard-code any variable values. The hidden test cases can be of different environments, and your code may fail if any hard coding is present.
- Submit your solution code as $RL\_Assignment2\_$ <Last 5 digits of your SR No.> $.py$ **(Case Sensitive)**
- We have very strong plagiarism checks in place. Any malpractice would lead to severe consequences, as per the institute's policy.
- Make sure that your code works on Python 3.8. Any other version will cause difficulty in the evaluation and will not be accepted.