Habib University
Assignment #1

Course Code: CS 352	Introduction to Reinforcement Learning	Given: 29/01/2023
Due: 12/02/2023	Spring 2023	Maximum Marks: 100

Instructions:

1. Please type the answers or write neatly in longhand.
2. Upload your word or PDF (if you write in longhand, scan and convert to PDF) file to Canvas.
3. Upload your Python code as .py file and the output of the programs as PDF files to Canvas.
4. No late submission will be accepted.

Q1. Exercise 3.1, page 85.

Programming Questions

Q2. *Symmetries in tic-tac-toe*
Let the row headings of a tic-tac-toe board be A, B, C, and the column headings be 1, 2, 3, so that the cells may be identified as A1, B2, etc. At any given instant during a game, the position of the board is a *state* of the game. In Python, the state may be saved as either a nested list (list of lists) or a dictionary such as {`‘A2’: ‘x’, ‘B3’:‘o’`}. Two positions of the board are *rotationally equivalent*–and can be represented by a single state–if one can be obtained from the other by 90°, 180°, or 270° rotation of the board. Write a program that accepts a dictionary as the position of the board and then determines if the position is a valid state, and: (a) prints an appropriate message, and returns a list containing an empty dictionary, if the input is not valid or is not a valid state; (b) prints an appropriate message, and returns an empty list, if the input is valid but has no rotational equivalents; (c) prints all the rotationally equivalent board positions and returns a list of corresponding dictionaries, if the input is valid.

Q2. *Shortest Path Problem*
In this problem we will implement the Shortest Path Algorithm that we discussed in the class. You have to write Python program that implements the algorithm discussed in the class as follows:

(i) The input to your program is a symmetric matrix (list of lists) representing a weighted undirected graph.
(ii) Your code should return two dictionaries: the first corresponding to the value function and the second corresponding to the optimal policy or policies.
(iii) You code should cater to the case of non-unique optimal policies (see example below).

You may use the `math` library but no other library such as `NumPy` or `SciPy`.

Example:

$$\text{Input}:[[0,2,3,\texttt{math.inf}],\ [2,0,\texttt{math.inf},3],\ [3,\texttt{math.inf},0,2],\ [\texttt{math.inf},3,2,0]]$$

$$\text{Output}:\{'S':5,'A':3,'B':2,'T':0\},\ \{'S':['A','B'],'A':'T','B':'T','T':'T'\}$$