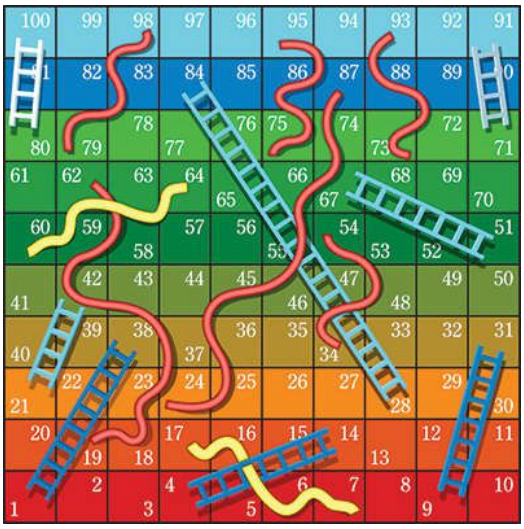
Lab 3: Chutes and Ladders

Due: Sunday, July 29th, 11:59 p.m.

Part 1 (50 pts):

Chutes and Ladders is a popular children's game where players compete to move along a game board the fastest. It involves both hazard squares that can set you back a few spaces and some helpful squares that can move you forward a few spaces.



If a player ends his turn on a Ladder, he immediately moves upwards along the ladder. E.g. If a player ends his turn on square 4, he would immediately move to square 14. If a player ends his turn on a Chute, he immediately moves down the slide. E.g. If a player ends his turn on square 17, he would automatically move to square 7.

In this lab, you will create a version of this game, with the chutes and ladders array provided to you in the lab3.py script.

Tasks:

- Make a die function that rolls a die and returns the resulting number.
- Write a function that creates a game with the given dimensions.
- Write a function that plays the game, keeping track of the number of players, their current position, and the number of moves they've taken (in an dictionary).
    - This function has two modes, which the user chooses from, one in which it asks the player to roll the die, and the second which plays itself until a player has won the game.

- Modify your game playing function so it returns the moves it took to win the game, and write another function that simulates games and finds the average number of moves it takes to reach the end for a single player.
    - Run the simulation (10 times, 100 times, and 1000 times) and record your findings in your readme.txt file.

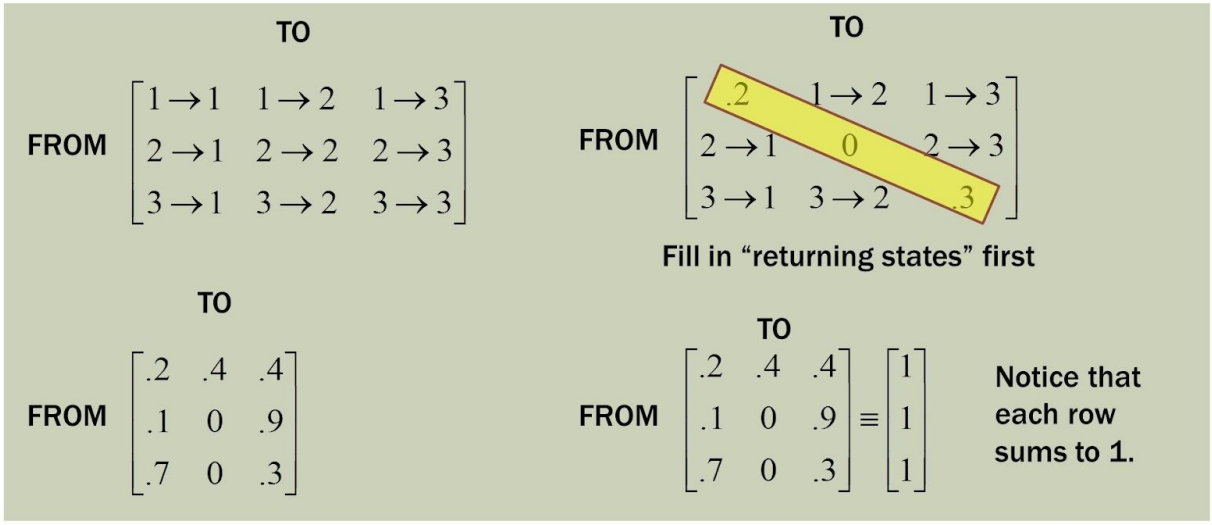
Testing
You should be writing test functions for each task, ensuring that the game runs correctly.

Part 2 (50 pts): *What's the average number of moves it takes for a player to win?*

In this part of the lab, we will work with matrices in the matrix.py file.

Tasks:

- Write a function that takes a width and a length and returns a matrix of those dimensions (Hint: list of lists).
- Write a function that takes two matrices, and multiplies them by each other (if possible), returning the resulting matrix.
- Map out a transition matrix for the chutes and ladders game in part 1.



- Write a function that simulates the chutes and ladders game with matrices to determine the average number of moves it takes to reach the end of the game.
    - To do this, you multiply a matrix of initial states by the transition matrix, which results in the updated states after every "move". As soon as the probability is 100% (or close, let's say 99%) of the game being in the last square, the game is over.
    - Calculate the weighted average of the game being over at each step based on the probability of being in the last square.
    - Run the same number of simulations as in part 1. Does anything change?

    - Record your findings in the readme.txt file.
- Are the simulations different? Which one is more accurate? What is the minimum number of moves required to win? The max?

Testing
You should be writing test functions for each task, ensuring that the game runs correctly.

Challenge (10 bonus pts)

Make bigger games of your own with more chutes and ladders and determine the average number of moves it takes to win, both simulating each game and with matrices.

Submit

To submit your work, simply commit and push the files to your lab3 directory in your git repository.