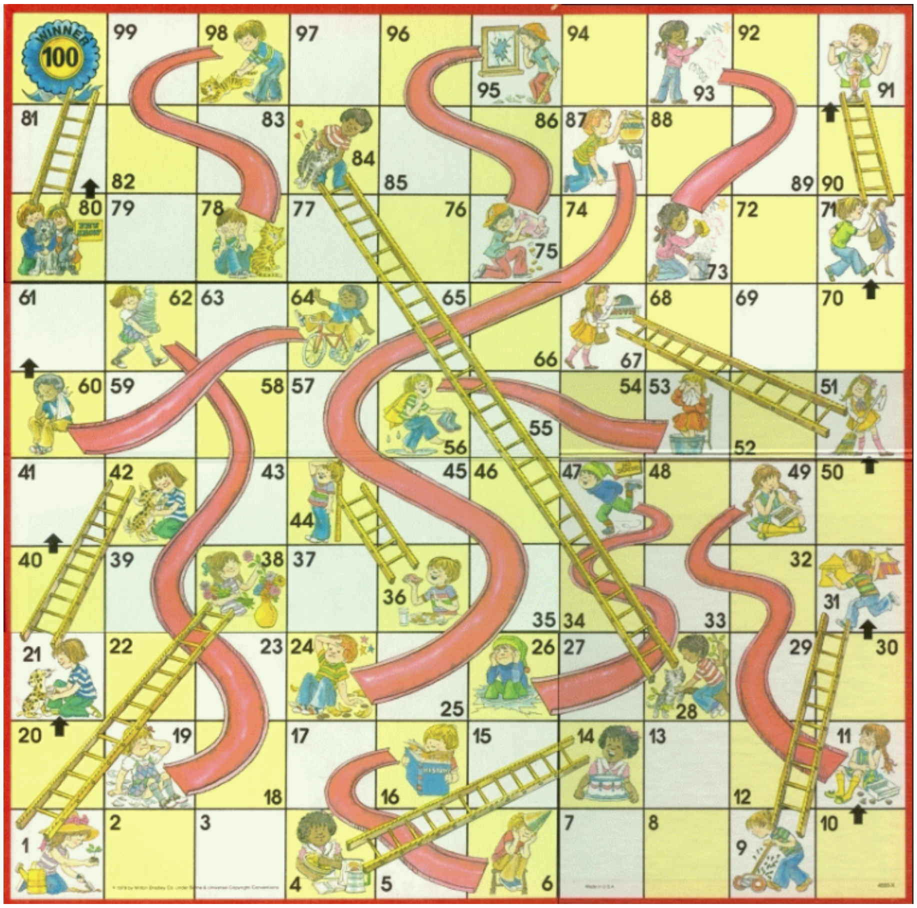
Python Assignment #6: Simulations!



Computers can do all sorts of stuff that are really hard for us to do. For instance, chutes and ladders is a game in which players spin a spinner (1,2,3,4,5,6) and move that many spaces on the board. However, if they land on the base of a ladder they climb all the way up. Unfortunately, if your move ends your character at the top of a slide you tumble all the way down. My question for you is ***on average*** how many turns does it take a single player starting at position 0 (not the ladder located at 1) to get all the way up to position 100?

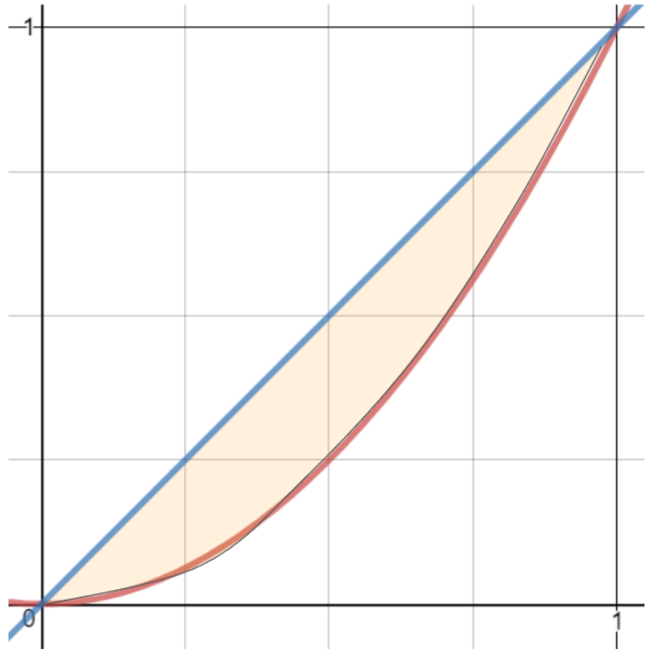
Strategy Outline:

1) Code the game first to run 1 full time correctly. I would print off the position at the end of each turn and print a special message if you hit a chute or ladder to confirm they are working. Have some sort of counter to keep track of how many times the user went.

2) Put it in a loop 10 times. Store the total amount of turns it took each of the ten rounds. I would print them off to make sure the counter is resetting each time. Can you find the average of that list?

3) Okay, now change the loop to 10,000. What is the average number of turns?

+ 50 Experience Points



The blue line is represented by the line y = x. The red line is $y = x^2$. Over the long run $y = x^2$ grows at a faster rate, but over the interval (0,1) y = x is greater. Unless you know calculus, finding the area between these two lines is impossible, well unless you have a computer.

Strategy Outline:

1) Can you make a loop that finds random x and y values between 0 and 1? For example: (0.924, .461) (0.323, .87) (0.716, .18) (0.86, .293). Use 3 decimal places.

2) Can you figure out if those points are above or below those lines? What points are in that orange region? Can you create a way for the computer to check?

3) Can you save all of those points in a list and check the length of that list versus the length of the list of numbers created?

4) With those two list lengths could you estimate the area in between those two curves?

+ 50 Experience Points

Challenge 3: Matchmaker

Everyone out there has that perfect someone. You both cry at the end of Finding Nemo, put ketchup on your scrambled eggs and are just a little bit afraid of the dark. Let's call that person a 100. There is also a pig farmer that refuses to eat pizza unless it has anchovies and is obsessed with obscure Norwegian cartoons. Let's call them a 1. Then there are all the people in between.

+ 50 Experience Points

If you pick 5 random numbers from 1-100 and are presented them one at a time, who would you choose? For example, if the first number presented was 56 would you settle? Then you are presented a 72? Then a 16? If you don't choose one of the first four numbers then you are stuck with the last value given in the list.

Here are some samples with the rule to select anyone presented that's over a 70.

-
```
[62, 98, 20, 67, 53]
98
```
They lucked out, a 98 was the second one offered and it fit the rule!

-
```
[68, 80, 58, 4, 92]
80
```
Uggh, got an 80 but there was a 92 if they would have just been more patient!!!!

-
```
[38, 3, 66, 6, 5]
5
```
When you end up being married to a 5 you can only look back at that 66 and wonder what if….

[Want a little starter code?]

Strategy Outline:

1) Can you make a randomly generated list of 5 numbers?
2) Can you test all 5 positions in the list to see if they are greater than 70?
    a) Confirm that you are selecting **only the first** number above 70 or selecting the last number.
3) Run your program 5 times and see if you can create a list of all the resulting life partners. While you are at it check each round and the results. Is everything going good?
4) Okay now try averaging the list of selected partners given in each trial.
5) Alright now run it 10,000 times to comes up with your best estimate for a predictive value for your rule.
6) Mess with your rule, should you choose only people higher than 60? 80? 97?