San José State University
Department of Applied Data Science

DATA 200
Computational Programming for Analytics

Spring 2024
Instructor: Ron Mak

Assignment #4

Assigned: Thursday, February 22
Due: Thursday, February 29 at 5:30 pm
100 points max

Monty Hall Puzzle
This assignment will allow you practice using Python lists and tuples, random numbers, and random sampling while performing a simulation to help answer a perplexing puzzle.

Choose the right door and win the new car!
Monty Hall was a popular game show host on American television. In one game on his show, you are a contestant from the studio audience. He shows you three doors, Door #1, Door #2, and Door #3. Hidden behind one door is a brand-new car. Hidden behind each of the other two doors is a goat.

Monty asks you to choose a door. You want to choose the right door and win the car.

Of course, Monty knows which door hides the car. After you've chosen a door, he opens one of the other doors and reveals a goat.

Monty then offers you the choice to stay with the door you originally chose, or you can switch your choice to the remaining third door. Would staying or switching give you a better chance to win the car? Or does it not matter?

The simulation of 1000 trials
Write a well-designed Python program in a Jupyter notebook to perform a simulation consisting of 1000 trials, each trial playing one game. For each trial, determine whether you would win the car by staying with your first door choice or by switching to your second door choice.

There is no user interaction with the program. During each trial, your program plays the roles of both Monty the game host and you the contestant. The sequence for each trial:

1. The program randomly hides the car behind door #1, #2, or #3.
2. You randomly pick a door as your first choice.
3. Monty opens a door to reveal a goat.
    - Monty knows behind which door the car is hidden.
    - If you had chosen the correct door, Monty randomly chooses one of the other two doors to open and reveal a goat. Use the function **`random.randint()`**.
    - Otherwise, Monty opens the only door he can to reveal a goat.
4. The remaining unopened third door is the one you could switch to as your second-choice door.
5. Record whether you win the car by staying with your first-choice door or by switching to your second-choice door.

Each trial generates a tuple
Each trial should generate a tuple consisting of these five integer elements:

- Which door hides the car.
- Your first-choice door.
- Which door Monty opens.
- Your second-choice door if you decide to switch.
- 0 if you win with your first-choice door, or 1 if you win with your second choice.

Examples:

- Generate the tuple (1, 3, 2, 1, 1) if the car is hidden behind Door #1 and your first choice is Door #3. Monty opens Door #2. If you decide to switch, your second choice is the remaining Door #1. In this trial, you would win by switching to your second-choice door (the fifth tuple element therefore is 1) .
- Generate the tuple (3, 3, 1, 2, 0) if the car is hidden behind Door #3 and your first choice is Door #3. Monty can open either Door #1 or #2 but randomly opens Door #1. If you decide to switch, your second choice is the remaining Door #2. In this trial, you would win by staying with your first-choice door (the fifth tuple element therefore is 0).

Store each generated tuple in a list. Therefore, at the conclusion of the simulation, you will have a list of 1000 tuples.

Random sampling
Based on these trials, you will estimate whether you win a car more often by staying or switching. Calculate the ratio of wins-by-switching over wins-by-staying. A ratio around 1 indicates that it doesn't matter whether you stay or switch. But if the ratio is significantly greater then 1, that indicates that you can win more often by switching. Conversely, a ratio significantly less than 1 indicates that you can win more often by staying.

Calculate this ratio for random samples of the tuples.

To generate a random sample of size n, again use the function **`random.randint()`**. Generate a list of n random integers from 0 through 999 inclusive — these will be the indexes of the tuples from the entire list of tuples for the random sample. Be sure the indexes are unique. Sort the list. Iterate through the tuples of the random sample to calculate and print the ratio.

Calculate and print the ratio for random samples of sizes 20, 50 (twice), 100 (twice), 250, and 500. Then calculate and print the ratio for the entire list of tuples. You will calculate and print 8 ratios.

Print the tuples of a random sample
Write a function that prints the tuples of a random sample. Example output for a random sample of size 20 tuples:

```
RANDOM SAMPLE SIZE: 20

         Car    Your   Monty    Your      Win
Trial  hidden   first  opened  second      if
index    here! choice    door  choice  switch?

   52       3       1       2       3     YES
  132       2       2       3       1
  138       3       3       1       2
  166       3       2       1       3     YES
  233       1       1       2       3
  240       2       1       3       2     YES
  271       2       3       1       2     YES
  300       2       1       3       2     YES
  328       1       3       2       1     YES
  368       1       3       2       1     YES
  402       2       2       3       1
  404       2       3       1       2     YES
  414       3       2       1       3     YES
  602       2       2       3       1
  627       2       1       3       2     YES
  735       2       3       1       2     YES
  846       2       2       3       1
  867       2       2       1       3
  928       1       1       3       2
  979       1       1       3       2

11 wins by switching out of 20, switch/stay win ratio is 1.22
```

Print the tuples of random samples of sizes of 20 and 50 (twice). Print the ratio for all the random sample sizes and for the entire list the tuples.

Program design
At this point in the semester, you should consider program design. Develop the program iteratively. Break up your program into small functions. Each function should be well-documented with its purpose, the purpose of each parameter, and any return values. Use other comments throughout as necessary. Choose meaningful names for your

variables and functions. The `NumberTranslator` program demonstrated in class is a good example of program structure, iterative development, and function documentation.

What to submit to Canvas
Submit your Jupyter notebook into Canvas: **Assignment #4.**

Rubric
Your program will be graded according to these criteria:

Criteria	Max points
Correct simulation	**60**
• Correct implementation of a trial.	• 40
• Random samples of various sizes.	• 15
• Correct calculation and printing of ratios.	• 5
Good program output	**20**
• Well-formatted output of the trials.	• 20
Good program design	**20**
• Each function is well-documented with its purpose, the purpose of each parameter, and any return values.	• 10
• Meaningful names for variable and functions.	• 5
• Meaningful comments inside functions.	• 5

Performing a computer-based simulation is a way data analysts can solve probability problems. Random sampling is an important technique for analyzing large amounts of data. What conclusions can you draw from this simulation regarding the Monty Hall puzzle?

For this assignment, you may choose one partner to work with. Clearly state in your comments at the beginning who the partners are.

Only one partner needs to submit. Both will receive the same score and comments.