15-259/559: Probability and Computing **Homework 2**
Due Friday 09/05/2025 at 12:50PM

Show all your work. Work through the problems carefully and **do not use online references, mathematical solvers, or GenAI** as a shortcut for finding the solutions. You will regret it in the quizzes and examinations.

Part I

Exercises 2.7, 2.8, 2.9, 2.10, 2.11, 2.12, 2.21, 2.22, 2.25, 2.28 in the textbook.

Part II

Consider again the Monty Hall problem from Exercise 2.22. Suppose you come up with the following strategy: after the host reveals the door with the goat, you will switch your guess with probability p or keep your guess with probability $1 - p$. Does this strategy help? How does it vary with the choice of $p \in [0, 1]$? Is there an optimal setting?

You will investigate this problem using a technique called *Monte Carlo simulation*. The key idea is to first write a computer program (called a *probabilistic program*) that implements a random experiment. We then estimate the probability of an event E by computing the fraction of times that E happens in a large number of independent runs of the program.

Step 1 Write a probabilistic program that takes as input a parameter $p \in [0, 1]$, and then generates a realization of the Monty Hall experiment as follows.

- Generate a random variable $C \sim \text{Uniform}(\{1, 2, 3\})$ for the door containing of the car.
- Generate a random variable $S \sim \text{Uniform}(\{1, 2, 3\})$ for the door selected by the player.
- Generate a random variable $R \sim \text{Uniform}(\{1, 2, 3\} \setminus \{C, S\})$ for the door revealed by Monty.
- Generate a random variable $B \sim \text{Bernoulli}(p)$ for the player's decision on whether to switch.
- Let G be your final guess defined as follows.
  - If $B = 0$, then set $G \leftarrow S$.
  - If $B = 1$, then set $G \leftarrow \{1, 2, 3\} \setminus \{S, R\}$.
- If $G = C$ then return 1 (player wins), else return 0 (player loses).

We recommend (but do not require) using Python. For example, you can easily generate Uniform and Bernoulli random variables using the following function in the NumPy library:

https://numpy.org/doc/stable/reference/random/generated/numpy.random.choice.html

Step 2 For each $p \in \{0, 0.1, 0.2, 0.3, \ldots, 0.9, 1\}$, run your program from Step 1 exactly 10,000 times using the argument p and then compute the fraction of times that it returns 1 (player wins). Produce a plot with p on the x-axis and the fraction of wins in 10,000 trials on the y-axis. What can you conclude? Submit your code and plot to Gradescope.