**Powerball Lottery**

jueves, 12 de enero de 2023

10:19 p. m.

Source: [Python Tutorial: Simulate the Powerball Lottery Using Python](https://www.youtube.com/watch?v=HZ8uXq5VG2w)

Notes from the app building to consider onwards

* It started of by stating the constants:

* The possible numbers a white balls could take and the possible numbers the red ball could take.

* The number of tickets per each drawing in the simulation.

* The number of drawings to be simulated.

* (To keep track of how much is earned and spent): Two numeric variables were initiated in 0.

* There is also a dictionary created to keep track of how many time which prices were hit.

* The first loop created, which is supposed to simulate each draw from the drawings stated in the constants, was created with the "iterator" named after the actual iterations, meaning: It was stated " for drawing in range(num\_drawings) " and not " for i in range(num\_drawings) " for readability purposes, because is possible to loss track for that "i" within whatever comes from the loop.

* For the white\_drawings, even when used the .sample() function from the random module, it will return a list, which is totally fine to work with, but since we are going to compare later with the number from our ticket, instead of looping to compare one list with the other, it would be more efficient to work with sets that have a method to compare within themselves.

* The winning numbers is the combination of the white\_drawings and the red\_draw, and the combinations is not even required for the simulation to work, but from Corey's experience it'd be better to have one point of reference and that is a dictionary.

* I think this is the most important lesson from this exercise: Since we're working in the loop of drawing the numbers and per each drawing we're suppose to have two shots (one per ticket bought), when the numbers are already drawn, we need to evaluate each of the tickets, so for this a nested loop is created to check each ticket against the winning numbers.

At this point we don't really know the logic to calculate how much is won by each ticket, according to the comparison*. Here Corey recommends to work each step at the time, so he stated the name of a function, yet to be created, and when needed, he goes and create the function outside of the loop, below the constants*. For me, this is valuable, this teach that I need to develop modular, by bits, not condensate everything in just one single loop, and If I got stuck at some point, I could try to breakdown everything in little pieces and make it work apart before mashing everything up .

And also, when creating the function, we probably won't know the logic yet, but a good practice is to stated the name, the input (parameters), the constants and the output, this give the mind a logic to follow and to take away the worry about those little things.

elif vs if: This is one of the things that caught my attention the most, when calculation the amount won, the first conditional is an if but for the rest of the cases (i.e.: If 4 ballots hit, if 3, if 2…) Corey's used an elif, and the explanation is simple: If and "if" were used, then if a 5 ballots hit, the code still would go through the validation of 4 and then to 3, etc… so the "elif" works for the cases **when we need to evaluate something only if the previous validation fails**.