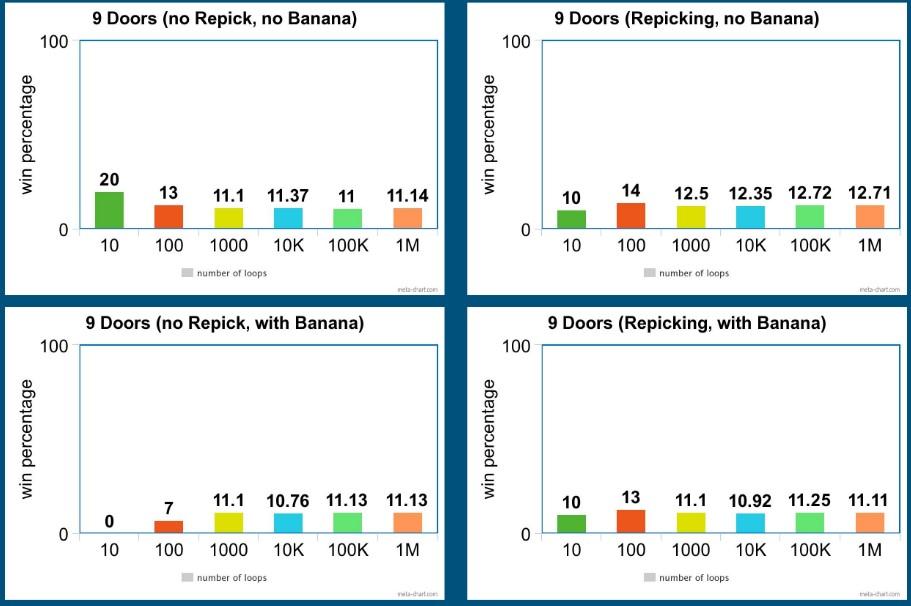
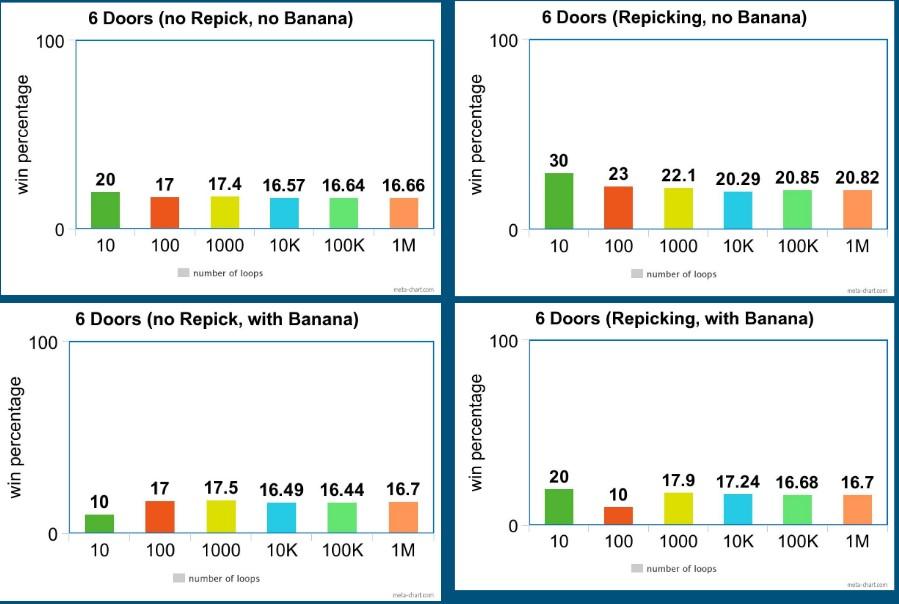
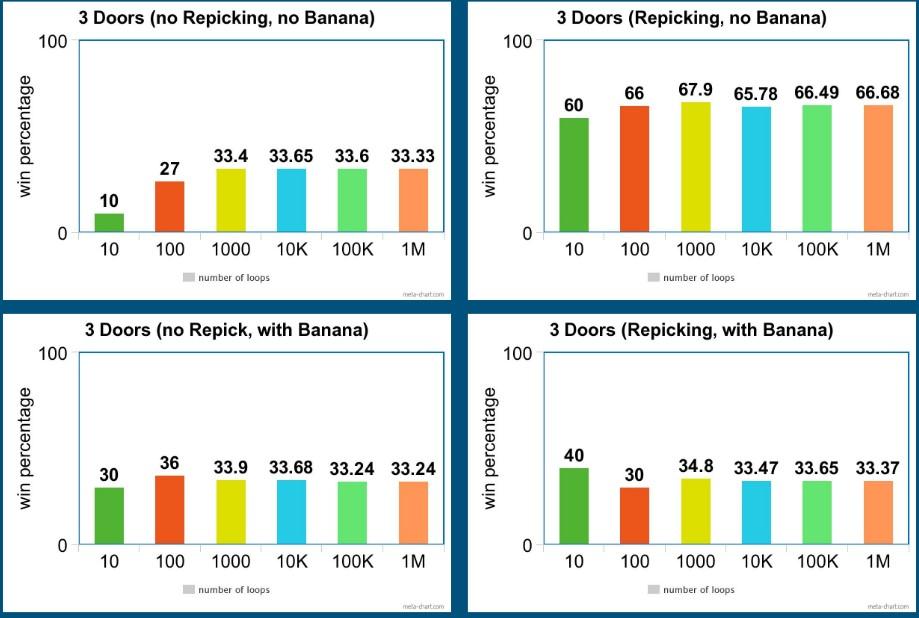
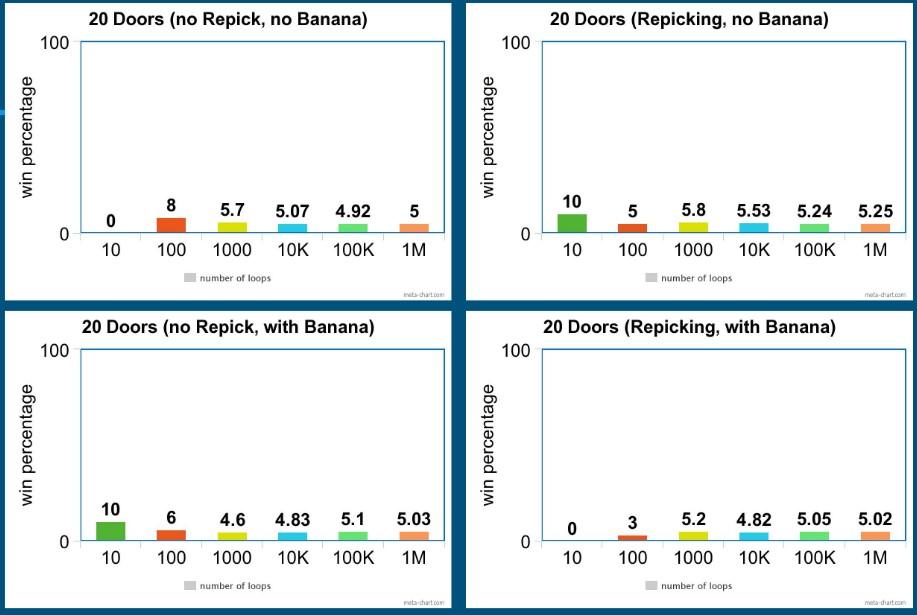
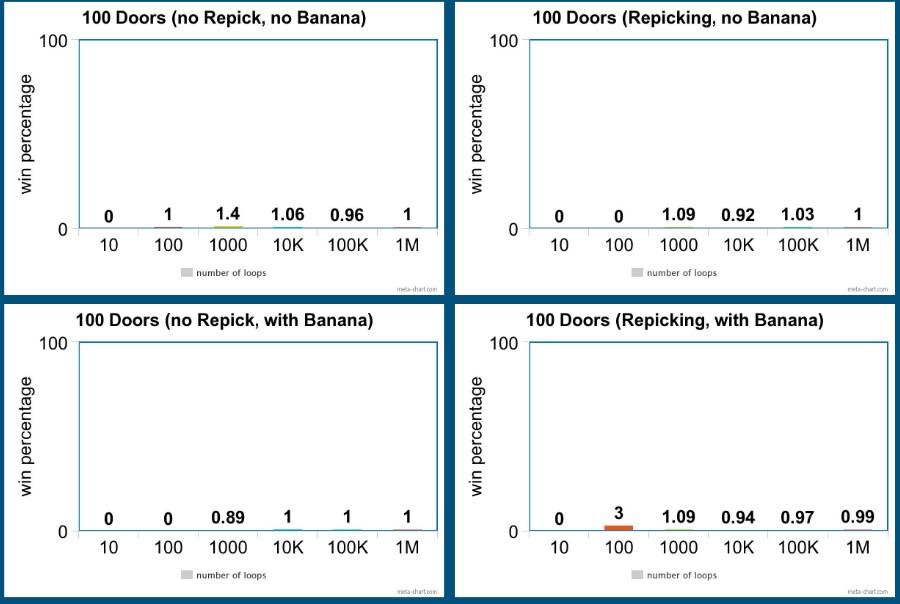
**Motivation:**  
 None of us had really heard about the Monty hall problem until it was introduced to us during our CSCI 154 class lecture, but that being said an interest in the brain teaser sparked pretty quickly. I think it interested us because it seemed so trivial to be the guest on a gameshow, and play the game. We figured it was all just luck, and there was nothing our actions within the rules could that would statistics predict our outcomes. Nevertheless, we decided to put our newly learned simulation skills to use and analyze the mathematical side of the famous problem for our class project.   
**Problem Statement:**  
 The Monty Hall Problem is a famous brain teaser made famous by the TV game show called “Let’s Make a Deal.” It was named after the show’s host Monty Hall. The game works as follows, a contestant is shown three doors: one which has a valuable item behind it, and two which have nothing or something of no value behind them. The Host first asks the contestant to pick a door at random. After which the host reveals one of the non-prize doors by opening it and then asks the contestant if he would like to switch doors. This is where the key part of the puzzle occurs, whether or not to switch doors. According to the mathematical claims it makes more sense to switch doors than to keep your original choice. Although we trust our professor and the information he is giving us, we decided to run some simulations.   
**Approach:**  
 We approached the problem by creating various types of policies to be tested at different iterations to help compare our different strategies. Our first policy would be to implement the choice to switch doors after randomly picking one and being shown a non-prize door. The second policy was the opposite of the first and instead of switching to the other door, the contestant would stick with his original choice. Our third policy was the same as the first, but we decided instead of the host choosing to open a non-prize door, we pretended he slipped on a banana peel and accidentally opened a door at random, and if the opened door happened to be the prize door then the contestant would lose. Policy 4 is the same as the second policy but with the introduction of the banana peel similar to that in policy 3. We also wanted to see how these policies would affect the outcomes with an increase in another factor, the number of total doors in the game. We figured the total number of iterations for each test would also need to be considered as smaller iterations might have different averages compared to larger ones. To sum up we decided to test our four policies with five different amounts of doors: 3, 6, 9, 20 and 100. We ran these simulations and averaged the results at different amounts of tests, those being, ten, one hundred, one thousand, ten thousand, one hundred thousand and one million iterations. Our choice for implementation was to create a python program to run/output the simulation tests. For every iteration the system randomly chose a door to be the prize door and randomly chose which door the contestant would pick to keep the results fair. It took a while to code the program and run the simulations because the test time increased substantially as the amount of iterations per test got bigger. This allowed us to get more stable estimates for the outcomes.  
  
**Results:**

**3 DOORS  
Fig. 1  
  
  
  
  
  
  
  
  
  
  
  
  
  
6 DOORS  
Fig. 2  
  
  
  
  
  
  
  
  
  
9 DOORS  
Fig. 3**

**20 DOORS  
Fig. 4  
  
  
  
  
  
  
  
  
  
  
  
100 DOORS  
Fig. 5**

**Conclusions:**

Many people have difficulties believing that switching doors after picking the first time has an outcome because they always think of the possibility that they have already chosen the correct door the first time. When you break down the first policy with three doors, you have a 1/3 chance of picking the prize door and 2/3 chances of picking the wrong door initially. When the host opens a non-prize door you remove an option and the odds move to 0 for the open door and the odds of the other door you didn’t pick have a 2/3 odds of being the correct door. In layman's terms, chances are you’re going to pick the wrong door first since the odds are against you, and then the host is forced to reveal another wrong door as well, thus leaving the chances of you switching to the prize door being 66.66 percent. If you look at figure 1, you can see that if you select the policy of switching doors(without banana), as the iterations increase you end up winning close to 66.6 percent of the time, thus proving that it makes more mathematical sense to switch. If you look at the figures 1-5 you can see that switching doors seems to make a bigger difference in winning but the difference is only noticeably significant with smaller amounts of doors, and as the number of doors increases the winning percentages tend to be the same as the outcome of one divided by the number of doors. Another thing to note is that with the number of increasing iterations, it seems that one thousand iterations seems to be enough to get stable results, and anything less than that amount of iterations tends to be really volatile in terms of winning percentages. Policies 3(banana, no repicking) and 4(banana, repicking) seem to always be very close to policy 1(no banana, no repicking. It’s important to note with 9 doors or less the banana seems to have the effect on canceling out the benefits of switching doors. As the number of doors increases past 9 doors, the banana stops having much of an effect on the outcomes and so does switching the doors since the gain from switching increases your chances of winning so minimally to see significant changes. To conclude, in the famous Monty Hall problem it always makes more sense to switch doors after a non-prize door is opened as this will give you the highest chances of winning.