Math 381: Assignment 6: Simulation

Ella Kim

November 19, 2021

1 Introduction

Let us consider a simple two-person game. With each person starting with a score of 0, each person/player rolls one even 6-sided die. They can either add the rolled value to their current total score, or they can subtract that value from the opponent's total score. In other words, the player currently rolling can only add to their own score or subtract from the opponent's score. The first player to reach a score of 40 wins. A player can never have a score less than 0 (so if a roll subtracted from a score results in a negative total, the score will be set to 0).

I would like to consider various strategies and their effectiveness in winning. Changing variables in the game that I considered were the current scores and the dice value rolled. These strategies are by no means the best or all-inclusive strategies, but the ones I will assess are listed below:

- Strategy 1: alternate between adding and subtracting each roll
- Strategy 2: always add
- Strategy 3: subtract if other player currently is more than integer α greater in score, add otherwise
- Strategy 4: add if roll value is 4-6, subtract otherwise (value 1-3)
- Strategy 5: subtract if other player has score β or greater, add otherwise

I simulate each game of a specific strategy using a Monte Carlo method. Let one player be Player 1 and the other be Player 2. I use a ratio to make an empirical estimate of the actual probability of Player 1 winning when using various strategies against Player 2. By repeatedly simulating a game 10,000 times* and keeping track of how many times Player 1 wins, the ratio/ empirical estimate of the probability is as follows:

$$\frac{\text{number of times Player 1 wins}}{\text{number of total games simulated (10,000)}}$$

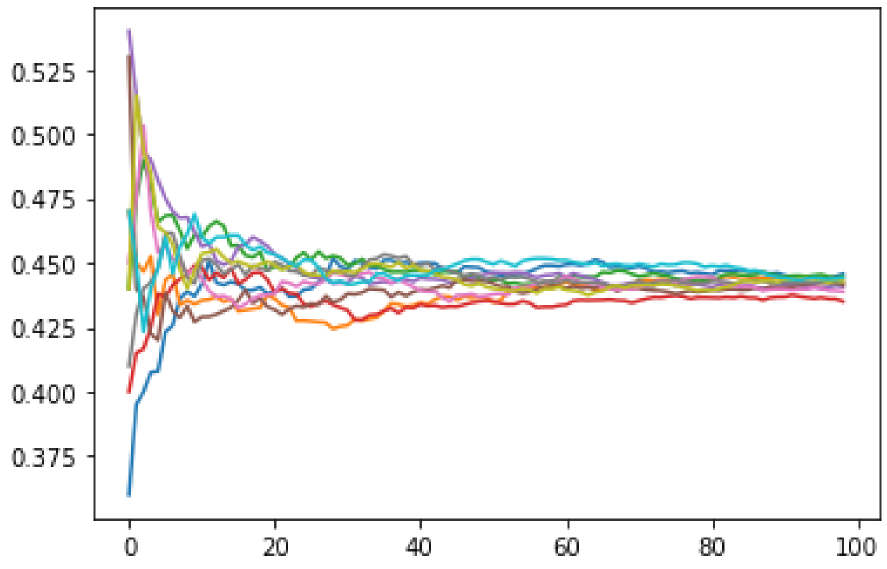
*Running each game 100,000 times had certain combinations of strategies running for 10-20+ minutes without any result/output. Additionally, the player who goes first is alternated for each of the 10,000 games.

2 Comparing Strategies:

As Monte Carlo methods are frequently used to estimate probabilities that are hard to solve with exact methods (i.e, to see how accurate the empirical estimates are), I use Python** to simulate each game 10,000 times, calculate the ratio every 100 games and after the final 10,000th game, and plot 10 runs of the 10,000 simulated games. The plot below is for games with Player 1 using Strategy 1, Player 2 using Strategy 2:

**As the Python codes for each strategy/output are lengthy, the code is provided at the end (in the Appendix), titled in accordance to its output described in the main text.

Plot of Convergence for Player 1- Strategy 1, Player 2- Strategy 2:

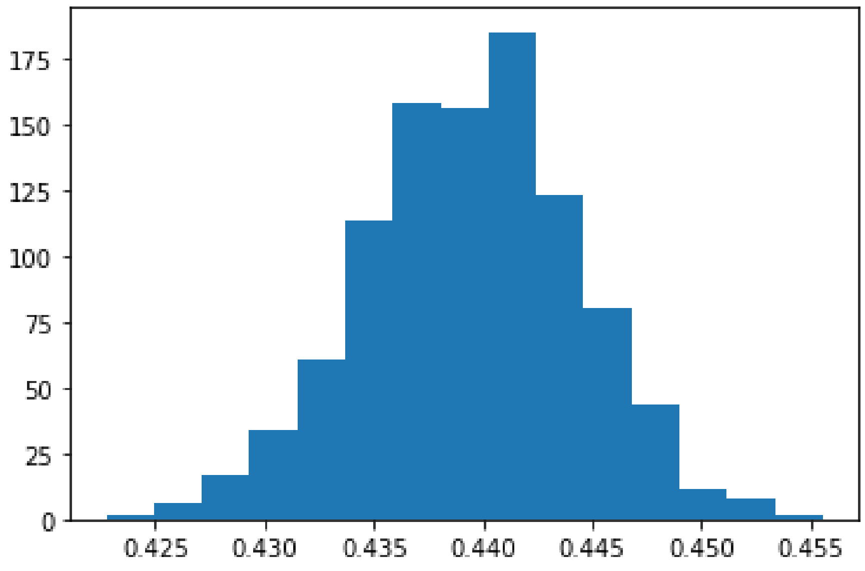


As the plot and title of the plot suggests, the ratio/estimated probabilities for the first few hundred simulated games differ greatly across each of the 10 runs. However, as more simulations of the game are ran, the ratio of Player 1 winning for each of the runs starts to converge to a smaller interval by the time it gets to the 10,000th total games (within .02 of each other).

Due to this convergence, I can therefore use this ratio and the Central Limit Theorem to solve for the confidence intervals of the probability of Player 1 winning with a certain strategy. Comparing these confidence intervals for the different strategies against each other will give likely estimates of which strategies are better.

If I simulate the game 10,000 times, I can also see this as my game being played 10,000 times, and when I run 10,000 games 1,000 times, that will yield 1,000 estimates of the winning probability for a certain strategy. Here is a histogram of the estimates for Player 1 using Strategy 1 and Player 2 using Strategy 2:

Histogram for Central Limit Theorem:



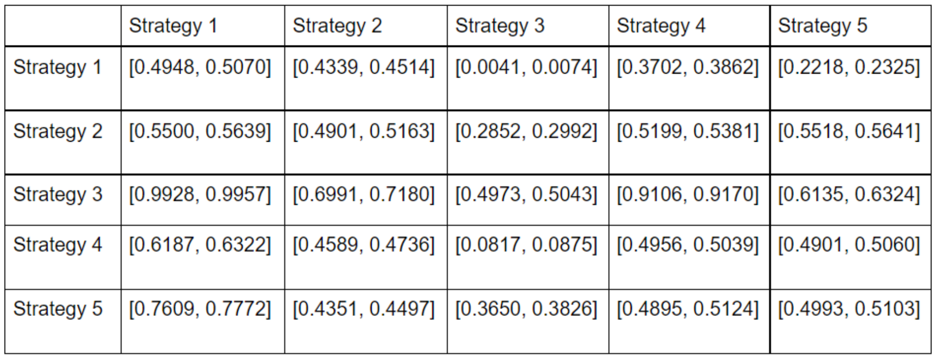
Although it's not perfect, the histogram does show a relatively normal distribution (symmetrical and uni-modal). Now that I have confirmed that these simulations follow do follow the Central Limit Theorem, I now test each combination of Player 1 and Player 2 using different strategies. For these tests, α for Strategy 3 is 0 (difference greater than 0), and β for Strategy 5 is 34, as a next roll of 6 would allow the opponent to win.

Again, because of the Central Limit Theorem, I can use confidence intervals to make precise estimates of the true probability. In order to compare how well Player 1 will do if they use a certain strategy against Player 2, I use the Confidence Interval Python code*** shown in the Appendix. From there, I get 10 estimates of each probability by running 10,000 games 10 times. As these were proven to be normally distributed by the Central Limit Theorem, there is an approx. 99.8% probability ($1 - \frac{1}{2^{10}} + \frac{1}{2^{10}} = 1 - \frac{2}{2^{10}} = 1 - \frac{1}{2^9} \approx 1 - 0.001953125$) that the true probability is between the highest and lowest estimates of the 10 runs. The intervals are rounded down for the lower interval and rounded up for upper interval. Below is a matrix the row strategies (Player 1), played against column strategies (Player 2).

***The different strategy combinations is adjusted in the while loop code, where

I replace the functions for strategy 1 and 2 (strat1 and strat2) with the corresponding strategies of each game

Matrix of Empirical Estimates of Probability for Person 1 Winning:



	Strategy 1	Strategy 2	Strategy 3	Strategy 4	Strategy 5
Strategy 1	[0.4948, 0.5070]	[0.4339, 0.4514]	[0.0041, 0.0074]	[0.3702, 0.3862]	[0.2218, 0.2325]
Strategy 2	[0.5500, 0.5639]	[0.4901, 0.5163]	[0.2852, 0.2992]	[0.5199, 0.5381]	[0.5518, 0.5641]
Strategy 3	[0.9928, 0.9957]	[0.6991, 0.7180]	[0.4973, 0.5043]	[0.9106, 0.9170]	[0.6135, 0.6324]
Strategy 4	[0.6187, 0.6322]	[0.4589, 0.4736]	[0.0817, 0.0875]	[0.4956, 0.5039]	[0.4901, 0.5060]
Strategy 5	[0.7609, 0.7772]	[0.4351, 0.4497]	[0.3650, 0.3826]	[0.4895, 0.5124]	[0.4993, 0.5103]

As seen in the matrix above, when players use the same strategy, it is very likely (with 99.8% confidence) that the true probability is approx. 0.5 for each player to win. Also seen in the matrix, let n be a strategy, and m be another that is not the same as n, then the probability that Player 1 will win using Strategy n against Player 2 using Strategy m is approximately the complimentary of Player 1 using Strategy m and Player 2 uses Strategy n, which I can see as intuitive. I say approximately, as I can only say with 99.8% confidence that the true probabilities are within the interval and could possibly not be exact complimentary of one another (only that the interval pairs contain certain ranges of each other's complimentary probabilities).

For example, I can say with 99.8% confidence that Player 1 using Strategy 3 has a 99.28-99.57% probability of winning against Player 2 using Strategy 1. In other words, Player 2 using Strategy 3 will have 99.8% confidence that they have a 0.43-0.72% probability (100% - 99.57%, 100% - 99.28%) of winning. On the reverse, I can also say with 99.8% confidence that Player 1 using Strategy 1 has a 0.41-0.74% probability of winning against Player 2 using Strategy 3. This [0.41, 0.74%] interval is within the complimentary interval for when Player 2 is using Strategy 1. As there is no bias between Player 1 and 2 (removed by alternating who goes first for each game simulated), I reiterate that this pattern of complimentary probability intervals makes sense to me.

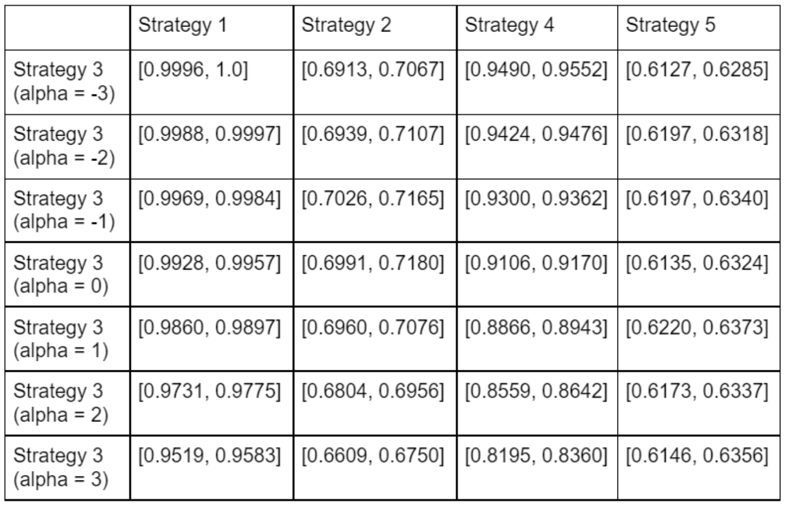
Focusing more on specific strategies and intervals, I define probability intervals whose highest boundary is under 0.5 as losing strategies (or more likely to lose), intervals which contain 0.5 as non-significant, and intervals whose lower boundary is over 0.5 as winning strategies (or more likely to win). Excluding the strategies playing against the same strategies (i.e, non-significant strategy combinations), when Player 1 uses Strategy 1, they are likely to lose against all other strategies, especially Strategy 3. On the other hand, when Player 1 uses Strategy 3, they are likely to win against all other strategies, especially when against Strategy 1 or 4, as there is 99.8% confidence that Player 1 will have a 99.28-99.57% and a 91.06-91.70% probability of winning, respectively.

3 A Closer Analysis on Strategy 3 & 5:

I return to Strategy 3 and 5, as they have α and β values that can be changed. With the $\alpha = 0$ and $\beta = 34$ set previously, Strategy 3 is the most effective of all the strategies (excluding Strategy 3 playing against Strategy 3). However, I will now take the time to try different values for α and β and see if they change the intervals for the true probability significantly, and whether Strategy 3 stays the most effective strategy. Although Strategy 5 was not the most effective overall, I say with 99.8% confidence it had the highest winning probability (36.50-38.26%) out of all the strategies Player 1 can use against a Player 2 using Strategy 3. Strategy 5 also does not have any significantly low winning true probability intervals like Strategies 1 and 4, but it does have what I define as non-significant intervals, as playing against Player 2 using Strategy 4 contains 0.5 probability.

The table below is for Player 1 using Strategy 3 (with Strategy 5's β as the default 34). Again, I simulate each combination of Strategies 10,000 times with 10 runs to use the same confidence percentage and interval (with the same calculations following the Central Limit Theorem). Therefore, the only places I change in my Python code are again the functions for each strategy used in the while loop, as well as the if statement's inequality in my strat3 function to the current testing α value. I remove Player 2 using Strategy 3's intervals in the table below, as they are always non-significant strategies/probabilities containing 0.5. The format of the table follows the matrix previously- Player 1's strategy are the row names, while Player 2's strategies are the column names (on next page):

Table of Person 1 using Strategy 3 w/ α = -3 to 3:



	Strategy 1	Strategy 2	Strategy 4	Strategy 5
Strategy 3 (alpha = -3)	[0.9996, 1.0]	[0.6913, 0.7067]	[0.9490, 0.9552]	[0.6127, 0.6285]
Strategy 3 (alpha = -2)	[0.9988, 0.9997]	[0.6939, 0.7107]	[0.9424, 0.9476]	[0.6197, 0.6318]
Strategy 3 (alpha = -1)	[0.9969, 0.9984]	[0.7026, 0.7165]	[0.9300, 0.9362]	[0.6197, 0.6340]
Strategy 3 (alpha = 0)	[0.9928, 0.9957]	[0.6991, 0.7180]	[0.9106, 0.9170]	[0.6135, 0.6324]
Strategy 3 (alpha = 1)	[0.9860, 0.9897]	[0.6960, 0.7076]	[0.8866, 0.8943]	[0.6220, 0.6373]
Strategy 3 (alpha = 2)	[0.9731, 0.9775]	[0.6804, 0.6956]	[0.8559, 0.8642]	[0.6173, 0.6337]
Strategy 3 (alpha = 3)	[0.9519, 0.9583]	[0.6609, 0.6750]	[0.8195, 0.8360]	[0.6146, 0.6356]

Surprisingly, the highest intervals are not in the same α's for each Strategy, as the highest intervals for Strategy 1 and 4 are when α is -3. I can say with 99.8% confidence that, from α's from -3 to 3, Strategy 3 is most likely to win Strategy 4 when α is -3, but I cannot say with the same confidence for Strategy 1, as the highest value in α -2 is 0.9997, which is inside the interval for α -3 [0.9996, 1.0], so there is a possibility the true probabilities are the same.

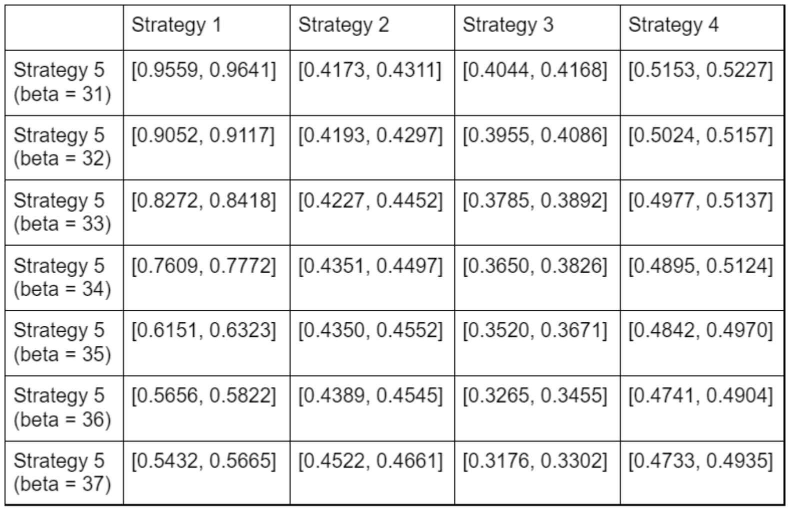
A similar lack of confidence can be said when Player 2 is using Strategy 2. Although Player 1 using an α -1 has the most narrow interval, α -3 through 1 all have probability intervals that contain values of the other, so I cannot say with any confidence that one α (in other words, the interval) is better than the other, as there is a possibility all these intervals have a true winning probability of 0.7. I can say with more confidence, however, that when Player 2 is ahead by more than 2 in score (α 2 and 3), that I have 99.8% confidence that the true probability decreases). I would guess that the probability will continue to decrease as Player 2 gets more ahead, but it's not something I can assume or conclude without more simulations of other α values.

Like Player 2 using Strategy 2 above, Player 1 going against Strategy 5 has intervals for all α that contain values of each other. Again, I cannot say with any confidence that one α is better than the other in this case, as there is a possibility that all the α have a 0.62 true probability.

In summary, Player 1 using Strategy 3 and going against Strategy 1 has the highest interval at α -1, has such similar intervals for α = -3 to 1 going against Strategy 2, has the highest interval at α = -3 going against Strategy 4, and has similar intervals for all α going against Strategy 5. Therefore, I would say, from α's -3 to 3, the best overall α value to use is -3, but I do not say this with much confidence, given the amount of overlap in the intervals and the individual confidence of the true probability for each strategy.

I repeat similar steps in my Python code, where I compute confidence intervals with 10 runs of 10,000 games for each type of combination of Player 1 using various β for Strategy 5 (with Strategy 3's α as the default 0 value previously). I also remove Player 2 using Strategy 5 in the table below, as they are also always non-significant strategies/probabilities containing 0.5:

Table of Person 1 using Strategy 5 w/ $\beta = 31$ to 37:



	Strategy 1	Strategy 2	Strategy 3	Strategy 4
Strategy 5 (beta = 31)	[0.9559, 0.9641]	[0.4173, 0.4311]	[0.4044, 0.4168]	[0.5153, 0.5227]
Strategy 5 (beta = 32)	[0.9052, 0.9117]	[0.4193, 0.4297]	[0.3955, 0.4086]	[0.5024, 0.5157]
Strategy 5 (beta = 33)	[0.8272, 0.8418]	[0.4227, 0.4452]	[0.3785, 0.3892]	[0.4977, 0.5137]
Strategy 5 (beta = 34)	[0.7609, 0.7772]	[0.4351, 0.4497]	[0.3650, 0.3826]	[0.4895, 0.5124]
Strategy 5 (beta = 35)	[0.6151, 0.6323]	[0.4350, 0.4552]	[0.3520, 0.3671]	[0.4842, 0.4970]
Strategy 5 (beta = 36)	[0.5656, 0.5822]	[0.4389, 0.4545]	[0.3265, 0.3455]	[0.4741, 0.4904]
Strategy 5 (beta = 37)	[0.5432, 0.5665]	[0.4522, 0.4661]	[0.3176, 0.3302]	[0.4733, 0.4935]

While Player 2 uses Strategy 1, I can say that it is very likely that, of β's $=$ 31 - 37, the most effective value is 31. I can use these intervals because my simulations followed the Central Limit Theorem, so I can say with 99.8% confidence that Player 1 using Strategy 1 with $\beta = 31$ has a true probability of winning against Player 2 using Strategy 1 between 95.59-96.41%. This interval also does not overlap with any of the other intervals for the different β's while Player 2 is using Strategy 1, so I am fairly certain that $\beta = 31$ is the best option from 31-37.

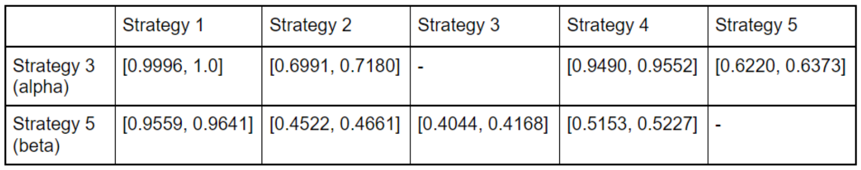
While Player 2 uses Strategy 2, the highest interval is when $\beta = 37$, but I cannot say with the same confidence as previously that it is truly the best out of the options in the table, as the interval for $\beta = 36$ overlaps ($\beta = 36$'s highest boundary 0.4545 is greater than $\beta = 37$'s lowest boundary 0.4522). So while there is a possibility $\beta = 37$ is the best, I cannot say this with much confidence, $\beta = 37$'s true probability is less than $\beta = 36$.

While Player 2 uses Strategy 3, the highest interval is when $\beta = 31$, but, like when Player 2 uses Strategy 2, while I can say with 99.8% confidence that the true probability for Player 1 winning when $\beta = 31$ is between 40.44-41.68%, the lowest bound 0.4044 is less than the highest bound for $\beta = 32$ (0.4086), so there is a possibility that $\beta = 31$'s true probability is less than $\beta = 32$.

And finally, while Player 2 uses Strategy 4, the highest interval for probabilities of Player 1 winning is when $\beta = 31$. However, like the previous two strategies, while there is a 99.8% confidence that the true winning probability is between 51.53-52.27%, the lowest bound 0.5153 is smaller than the highest bound for $\beta = 32$ 0.5157. While there is only a 0.0004 overlap, I still cannot rule out the fact that there is a possibility the true winning probability is smaller for $\beta = 31$. Therefore, I cannot say with great confidence that $\beta = 31$ is the best of $\beta = 31$-37 for going against Strategy 4.

Regardless of which α is best for Strategy 3 and which β is the best for Strategy 5, here is a table with the intervals that had the largest upper bound for each strategy (i.e, they do not have the same α or β values):

Table of Highest Upper-Bound Intervals for Strategy 3 & 5 vs Other Strategies:



	Strategy 1	Strategy 2	Strategy 3	Strategy 4	Strategy 5
Strategy 3 (alpha)	[0.9996, 1.0]	[0.6991, 0.7180]	-	[0.9490, 0.9552]	[0.6220, 0.6373]
Strategy 5 (beta)	[0.9559, 0.9641]	[0.4522, 0.4661]	[0.4044, 0.4168]	[0.5153, 0.5227]	-

The reasoning for this comparison is that, because of the Central Limit Theorem, I can say with 99.8% confidence that the true winning probability for their corresponding α and β values is within these intervals. So although this does not guarantee these particular α and β values have the highest winning probability, regardless of what the true winning probability is, the highest it could every truly be is the largest winning probability in all the intervals. Even with this, Player 1 using Strategy 5 is not likely to win against Strategy 2 or 4 (as the intervals are below 0.5 probability), and will barely have a better probability of winning playing against Strategy 4. In comparison, all of Player 1 using Strategy 3's intervals all have lower bounds greater than 0.6.

Even when comparing the intervals I can with Player 1 using Strategy 5, none of the lowest bounds for Strategy 3 are less than the highest bounds for Player 1 using Strategy 5. So I am also fairly certain that Strategy 5 will never be more effective than Strategy 3 with the values of α and β that I simulated and computed confidence intervals for their true winning probabilities. In conclusion, Strategy 3 is the most effective strategy (Player 1 subtracts if other Player 2's score is more than integer α greater in score, add otherwise) for Player 1 to use against Player 2, and the α value would depend on which Strategy Player 2 is using (as described after the Strategy 3 table).

A Appendix: Python Code

Person 1- Strategy 1, Person 2- Strategy 2: Confidence Interval:

```
import random
import math
import pandas as pd

# how many games we will play
tries = 10000
# track how many wins player 1 has
wins = 0
# score to win
winScore = 40
# trials for CI
runs = 10

# for interval
estimates = []

def strat1(scoreFirst, scoreSec, turn, roll):
    # alternating turns, add here
    if turn%2 == 1:
        scoreFirst += roll
    else:
        scoreSec -= roll
        # in case score goes below 0 (should not be possible)
        if (scoreSec < 0):
            scoreSec = 0
    return [scoreFirst,scoreSec]

def strat2(score, roll):
    score += roll
    return score

def strat3(scoreFirst, scoreSecond, roll):
    # subtract if other player is winning
    if scoreSecond - scoreFirst <= 0:
        scoreFirst += roll
    else:
        scoreSecond -= roll
        if (scoreSecond < 0):
            scoresSecond = 0
    return [scoreFirst, scoreSecond]

def strat4(scoreFirst, scoreSecond, roll):
    # add if roll is greater than 3
    if roll > 3:
        scoreFirst += roll
    else:
        scoreSecond -= roll
        if (scoreSecond < 0):
            scoreSecond = 0
    return [scoreFirst, scoreSecond]

def strat5(scoreFirst, scoreSecond, roll):
    # add if other opponent is at beta score or greater
    if (scoreSecond >= 34):
        scoreSecond -= roll
        if (scoreSecond < 0):
            scoreSecond = 0
    else:
        scoreFirst += roll
    return [scoreFirst, scoreSecond]

for trial in range(0,runs):
    # current game #
    for game in range(0,tries):
        # track current score for players 1 & 2
        score1 = 0
        score2 = 0
        rolls = 0

        while (score1 < winScore and score2 < winScore):

            # rolls for players 1 & 2
            roll1 = random.randint(1,6)
            roll2 = random.randint(1,6)
            rolls += 1

            # player1 goes first
            if game%2 == 0:
                [score1, score2] = strat1(score1, score2, rolls, roll1)
                # if player1 wins
                if score1 >= winScore and score2 < winScore:
                    #print(score1, score2)
                    wins += 1
                    break
                score2 = strat2(score2, roll2)
            else: #player 2 goes first
                score2 = strat2(score2, roll1)
                # if player2 wins
                if score2 >= winScore and score1 < winScore:
                    break
                [score1, score2] = strat1(score1, score2, rolls, roll2)
                if score1 >= winScore and score2 < winScore:
                    #print(score1, score2)
                    wins += 1


    estimates.append(wins*1./game) # add final estimate
    wins = 0 #reset for next run
print(min(estimates), max(estimates))
```

* * *

Player 1- Strategy 1, Player 2- Strategy 2: Convergence Plot:

```
import matplotlib.pyplot as plt
# set x axis interval
x = list(range(0,99))

y = [df["0"].tolist(),df["1"].tolist(),df["2"].tolist(),df["3"].tolist(),
    df["4"].tolist(),df["5"].tolist(),df["6"].tolist(),df["7"].tolist(),
    df["8"].tolist(),df["9"].tolist()]
# for each function
for i in y:
    # plot each function
    plt.plot(x,i)
# display all of them at once
plt.show()
```

* * *

Player 1- Strategy 1, Player 2- Strategy 2: Histogram:

```
import random
import math
import pandas as pd
import matplotlib.pyplot as plt

# how many games we will play
tries = 10000
# track how many wins player 1 has
wins = 0
# score to win
winScore = 40
# trials for CI
runs = 1000
```

```
# for interval
estimates = []

# for plot
df = pd.DataFrame()

def strat1(scoreFirst, scoreSec, turn, roll):
    # alternating turns, add here
    if turn%2 == 1:
        scoreFirst += roll
    else:
        scoreSec -= roll
        # in case score goes below 0 (should not be possible)
        if (scoreSec < 0):
            scoreSec = 0
    return [scoreFirst,scoreSec]

def strat2(score, roll):
    score += roll
    return score

def strat3(scoreFirst, scoreSecond, roll):
    # subtract if other player is winning
    if scoreSecond - scoreFirst <= 0:
        scoreFirst += roll
    else:
        scoreSecond -= roll
        if (scoreSecond < 0):
            scoresSecond = 0
    return [scoreFirst, scoreSecond]

def strat4(scoreFirst, scoreSecond, roll):
    # add if roll is greater than 3
    if roll > 3:
        scoreFirst += roll
    else:
        scoreSecond -= roll
        if (scoreSecond < 0):
            scoreSecond = 0
    return [scoreFirst, scoreSecond]

def strat5(scoreFirst, scoreSecond, roll):
    # add if other opponent is at beta score or greater
    if (scoreSecond >= 34):
        scoreSecond -= roll
        if (scoreSecond < 0):
            scoreSecond = 0
    else:
        scoreFirst += roll
    return [scoreFirst, scoreSecond]

for trial in range(0,runs):
    run = []
    # current game #
    for game in range(0,tries):
        # track current score for players 1 & 2
        score1 = 0
        score2 = 0
        rolls = 0

        while (score1 < winScore and score2 < winScore):

            # rolls for players 1 & 2
            roll1 = random.randint(1,6)
            roll2 = random.randint(1,6)
            rolls += 1

            # player1 goes first
            if game%2 == 0:
                [score1, score2] = strat1(score1, score2, rolls, roll1)
                # if player1 wins
                if score1 >= winScore and score2 < winScore:
                    #print(score1, score2)
                    wins += 1
                    break
                score2 = strat2(score2, roll2)
            else: #player 2 goes first
                score2 = strat2(score2, roll1)
                # if player2 wins
                if score2 >= winScore and score1 < winScore:
                    break
                [score1, score2] = strat1(score1, score2, rolls, roll2)
                if score1 >= winScore and score2 < winScore:
                    #print(score1, score2)
                    wins += 1

            # add to nth run for plot
        if((game % 100) == 0 and (game>0)):
            run.append(wins*1./game)
    df[str(trial)] = run # add each run to df for plot
    estimates.append(wins*1./game) # add final estimate
    wins = 0 #reset for next run

# plot histogram
plt.hist(estimates, density=False, bins=15)
plt.show()
```