Homework 1 Part 2

This is an individual assignment.

Description

Create or edit this Jupyter Notebook to answer the questions below. Use simulations to answer these questions. An analytical solution can be useful to check if your simulation is correct but analytical solutions alone will not be accepted as a solution to a problem.

Problem 1

Consider repeatedly rolling a fair 4-sided die.

1. Create a simulation to compute the probability that the top face will be 4 at least once on four rolls of the die?
2. Create a simulation to compute the probability that the top face will be 4 at least once on 20 rolls of the die?
3. Create a simulation to compute how many rolls of the die would you have to do to be 90% confident that you would see at least one 4?
4. Using the formula you have computed in problem 2 part 4, make a Python function that takes in the target value p and outputs the required number of rolls of an integer.
   A. Find the values for $p=0.95$ and $p=0.99$.
   B. Use your simulation to verify that the number of rolls you specified is sufficient to achieve $p\geq 0.95$.

In [14]:
```
# Part 1
import random
import numpy as np
import numpy.random as npr

num_sims = 100000
dice = npr.randint(1, 5,size=(4, num_sims))

print("The probability that the top face will be 4 at least once on four rolls is",np.
```

```
The probability that the top face will be 4 at least once on four rolls is 0.68327
```

In [15]:
```
# Part 2
import random
import numpy as np
import numpy.random as npr

num_sims = 100000
dice = npr.randint(1, 5,size=(20, num_sims))

print("The probability that the top face will be 4 at least once on twenty rolls is",r
```

```
The probability that the top face will be 4 at least once on twenty rolls is 0.99667
```

In [23]:
```
# Part 3
import random
import numpy as np
import numpy.random as npr

num_sims = 100000

for i in range(100):
    dice = npr.randint(1, 5,size=(i, num_sims))
    if np.sum(np.sum(dice==4, axis = 0)>=1)/num_sims >= 0.9:
        print("It would take",i,"rolls to be 90% confident you would see at least 1 fo
        break
```

```
It would take 9 rolls to be 90% confident you would see at least 1 four.
```

In [31]:
```
# Part 4
import random
import numpy as np
import numpy.random as npr

num_sims = 100000
p1 = 0.95
p2 = 0.99

for i in range(100):
    dice = npr.randint(1, 5,size=(i, num_sims))
    if np.sum(np.sum(dice==4, axis = 0)>=1)/num_sims >= p1:
        print("It would take",i,"rolls to be", p1*100,"% confident you would see at le
        print("To verify, the probability of rolling a 4 in",i,"rolls is",np.sum(np.su
        break
for i in range(100):
    dice = npr.randint(1, 5,size=(i, num_sims))
    if np.sum(np.sum(dice==4, axis = 0)>=1)/num_sims >= p2:
        print("It would take",i,"rolls to be", p2*100,"% confident you would see at le
        print("To verify, the probability of rolling a 4 in",i,"rolls is",np.sum(np.su
        break
```

```
It would take 11 rolls to be 95.0 % confident you would see at least 1 four.
To verify, the probability of rolling a 4 in 11 rolls is 0.95862
It would take 17 rolls to be 99.0 % confident you would see at least 1 four.
To verify, the probability of rolling a 4 in 17 rolls is 0.99271
```

Problem 2

Create a simulation function where you will roll a fair 6-sided die twice. Use simulation to find out the probability of getting a 4 or 6 on the first toss and a 1,2,3, or 5 on the second toss.

In [34]:
```
import random
import numpy as np
import numpy.random as npr

num_sims = 100000
p4or6 = 0
pNot4or6 = 0

for i in range(num_sims):
```

```
    die1 = random.choice(range(1,7))
    die2 = random.choice(range(1,7))

    if die1 == 4 or die1 == 6:
        p4or6 +=1
        if die2 <= 3 or die2 == 5:
            pNot4or6 +=1

print('The probability of getting a 4 or 6 on the first toss and a 1, 2, 3, or 5 on th
```

```
The probability of getting a 4 or 6 on the first toss and a 1, 2, 3, or 5 on the seco
nd toss is 0.22184
```

Problem 3

Suppose that you have a bag with 3 coins. One of them is a fair coin, but the others are biased trick coins. When flipped, the three coins come up heads with probability $\frac{1}{2}$, $\frac{1}{4}$, and $\frac{1}{6}$, respectively.

Consider the experiment where you pick one coin at random and flip it three times. Let H_i be the event that the coin comes up heads on flip i. What is the probability of the outcome $H_1\cap H_2\cap \overline{H_3}$?

With small modification in your code, find out the probability of the outcome $H_1\cap \overline{H_2} \cap \overline{H_3}$.

Use simulation to find out the probability.

In [47]:
```
def prob3(num_sims = 100000):
    heads12not3 = 0
    coins = ['F', 'UF', 'SUF']
    for sim in range(num_sims):
        coin = random.choice(coins)
        if coin == 'F':
            S = ['H', 'T']
        elif coin == 'UF':
            S = ['H', 'T', 'T', 'T']
        else:
            S = ['H', 'T', 'T', 'T', 'T', 'T']
        values = random.choices(S, k=3)
        if values[0] == 'H' and values[1] == 'H' and values[2] == 'T':
            heads12not3 += 1
    print('Probability of H1, H2, and not H3 is',heads12not3/num_sims)
```

In [48]:
```
prob3(num_sims)
```

```
Probability of H1, H2, and not H3 is 0.06611
```

In [50]:
```
def prob3pt2(num_sims = 100000):
    heads1not2not3 = 0
    coins = ['F', 'UF', 'SUF']
    for sim in range(num_sims):
        coin = random.choice(coins)
        if coin == 'F':
```

```
            S = ['H', 'T']
        elif coin == 'UF':
            S = ['H', 'T', 'T', 'T']
        else:
            S = ['H', 'T', 'T', 'T', 'T', 'T']
        values = random.choices(S, k=3)
        if values[0] == 'H' and values[1] == 'T' and values[2] == 'T':
            heads1not2not3 += 1
    print('Probability of H1, not H2, and not H3 is',heads1not2not3/num_sims)
```

In [51]:

```
prob3pt2(num_sims)
```

```
Probability of H1, not H2, and not H3 is 0.12782
```

Submit Your Solutions

Confirm that you've successfully completed the assignment.

Along with the Notebook, include a PDF of the notebook with your solutions.

`add` and `commit` the final version of your work, and `push` your PDF file to your GitHub repository.

Submit the URL of your GitHub Repository as your assignment submission on Canvas.