Project #1 – Coin flips

EE 511: Fall 2019

Qiong Wang Id: 5906740674

Tool: PyCharm

1. Simulation of unbiased Bernoulli trial – Q1

Requirement: Simulate tossing a fair coin (a Bernoulli trial) 50 times. Count the number. of heads. Record the longest run of heads. Generate a histogram for the Bernoulli outcomes.

Analysis: Using int(rd.random()+0.5) to simulate a fair toss, 1: HEAD, 0: TAIL. Saving the result of each flip to list and calculate the frequency of appearance of the Head and Tail after toss and record the longest run of heads. For B～(50,0.5), the number of heads should be about 25 times.

Code:

```
import random as rd
import matplotlib.pyplot as plt

# using a list to save the result of each toss, 0: Tail 1: Head
result = []
# toss 50 times
for i in range(50):
    # using generate 0 or 1 with equal probability
    result.append(int(rd.random()+0.5))

# record the longest run of head
longest_run_of_head = 0
# record the length of each consequent head
counter = 0
for ele in result:
    if ele:
```

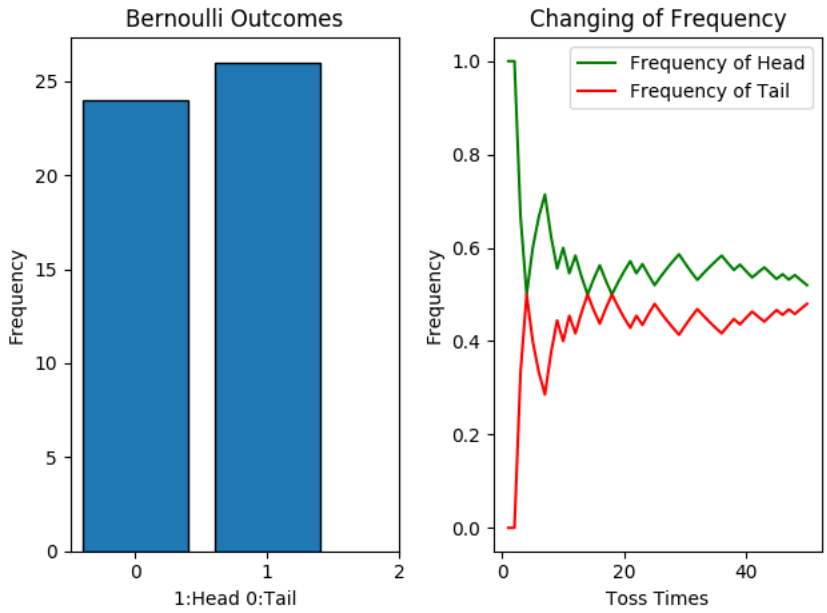
```
        counter += 1
    else:
        # compare each length of consequent head with the previous
longest run of head
        longest_run_of_head = max(longest_run_of_head,counter)
        counter = 0
longest_run_of_head = max(longest_run_of_head,counter)
# print result
print('Longest Run of Head:', longest_run_of_head)
print('Number of Head:', result.count(1))

# Calculate the frequency of appearance of the Head and Tail after
toss
# number of Heads
sum = 0
# frequency of appearance of the Head
prob_head = []
# frequency of appearance of the Tail
prob_tail = []
toss_times = 0
for ele in result:
    toss_times += 1
    if ele:
        sum += 1
    # frequency of appearance
    prob_head.append(sum/toss_times)
    prob_tail.append(1-sum/toss_times)

# plot a line chart and histogram
plt.figure()

plt.subplot(1,2,1)
plt.hist(result, bins=range(3), rwidth=0.8, edgecolor='black',
align='left')
plt.xticks(range(3))
plt.xlabel('1:Head 0:Tail')
plt.ylabel('Frequency')
plt.title('Bernoulli Outcomes')

plt.subplot(1,2,2)
plt.plot(range(1,toss_times+1), prob_head, color='green',
```

```
label='Frequency of Head')
plt.plot(range(1,toss_times+1), prob_tail, color='red',
label='Frequency of Tail')
plt.xlabel('Toss Times')
plt.ylabel('Frequency')
plt.title('Changing of Frequency')
plt.legend()

plt.show()
```

Result: When I simulate tossing a fair coin 50 times the result is:

Longest Run of Head: 5

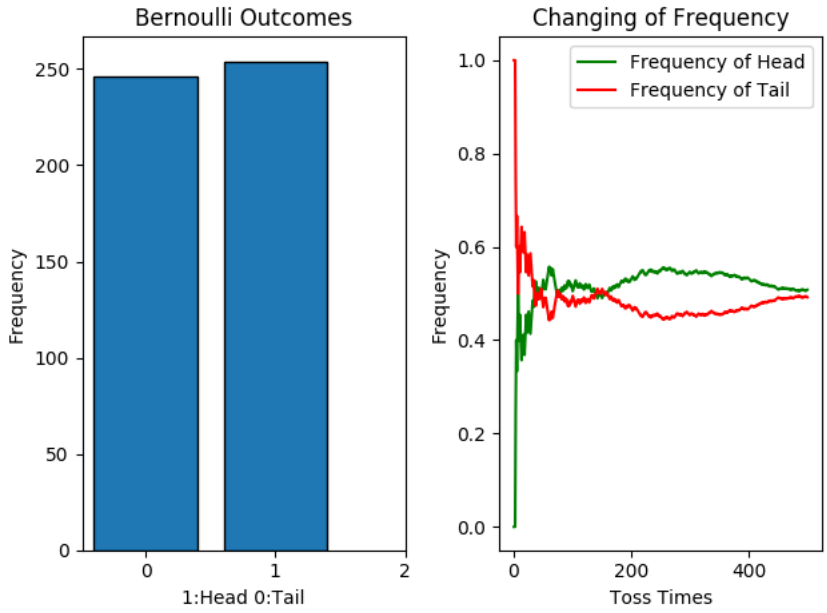
Number of Head: 26



From the right figure, we can easily find that, the frequency of head and tail is fluctuate at beginning and with the number of toss times increased, the line become stable. In order to get a clearly result, I enlarge the number of toss times to 500, the result is:

Longest Run of Head: 8

Number of Head: 254



And we can easily find that, the frequency of head and tail will converge to 0.5 when toss times is larger.

2. Number of heads – Q1a

Requirement: Repeat the above experiment 20, 100, 200, and 1000 times. Generate a histogram for each showing the number of heads in 50 flips. Comment on the limit of the histogram.

Analysis: From CLT we know that the distribution is about normal distribution when we have a large amount of dataset. So, I try different number of Bernoulli trial (20,100,200,1000,2000,10000). I

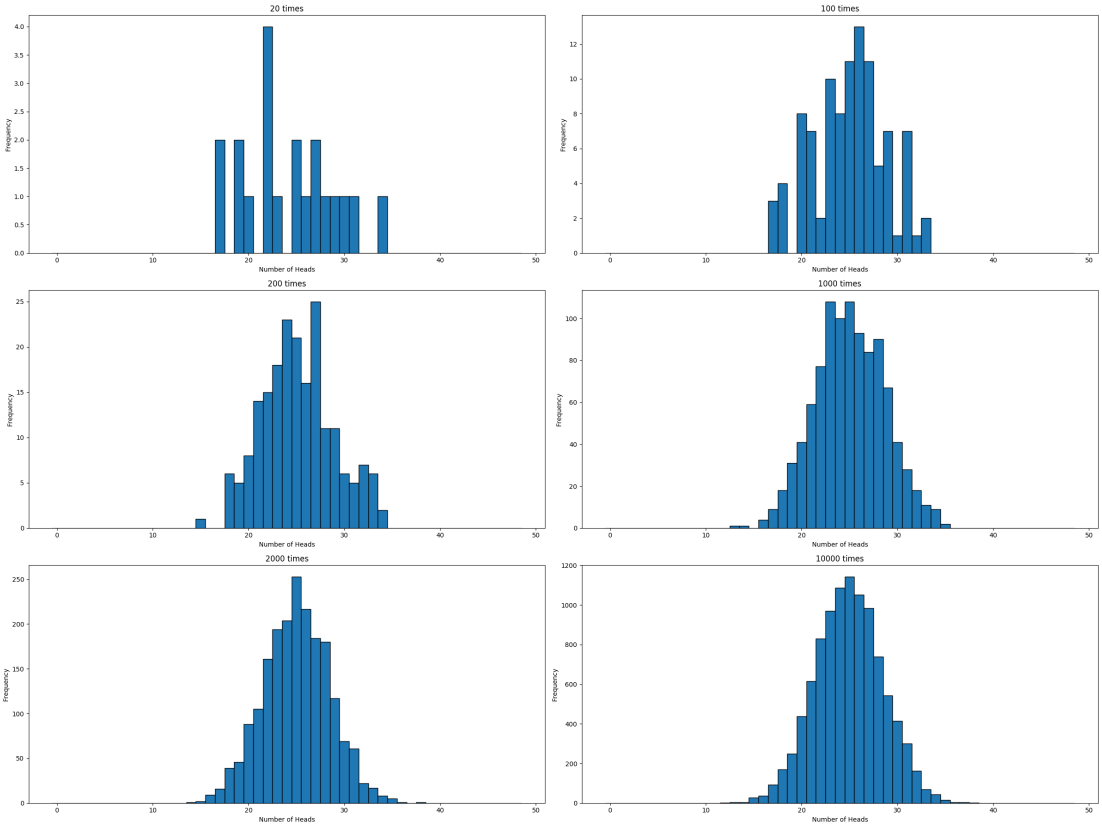
Code:

```
import random as rd
import matplotlib.pyplot as plt
```

```
def main():
    # get the dataset for different number times of Bernoulli trial
(n=50, p=0.5)
    result20 = experiment(20)
    result100 = experiment(100)
    result200 = experiment(200)
    result1000 = experiment(1000)
    result2000 = experiment(2000)
    result10000 = experiment(10000)

    # set the size of the figure, and plot 6 histograms
    plt.rcParams['figure.figsize'] = (24, 18)
    plt.rcParams['savefig.dpi'] = 300
    plt.figure()

    result = [result20, result100, result200, result1000, result2000,
result10000]
    times = [20, 100, 200, 1000, 2000, 10000]
    for i in range(1, 7):
        plt.subplot(3, 2, i)
        plt.hist(result[i - 1], bins=range(50), rwidth=1,
edgecolor='black', align='left')
        plt.title(str(times[i - 1]) + ' times')
        plt.xlabel('Number of Heads')
        plt.ylabel('Frequency')

    plt.show()


# return the number of heads for each Bernoulli trial (n=50, p=0.5)
def experiment(times: int) -> list:
    result = []
    for j in range(times):
        head = 0
        for i in range(50):
            head += int(rd.random() + 0.5)
        result.append(head)
    return result
```

```
if __name__ == '__main__':
    main()
```

Result: From the figures below, we find that, as the number of trials increased, the histogram converges to a normal distribution. Mean = 0.5 * 50 = 25, Variable = 50 * 0.5 * 0.5 = 12.5. So, the limit of the histogram is N∼(25,12.5)



3. Simulation of biased Bernoulli trial – Q2

Requirement: Simulate tossing a biased coin 200 times where *P* [HEAD] = 0.80. Count the number of heads. Record the longest run of heads. Generate a histogram for the Bernoulli outcomes.

Analysis: When we toss a bias coin, P(HEAD) = 0.8, 200 times, the number of head = 0.8 * 200 = 160, and the number of the longest run of head is more likely to larger than a fair coin.

Code:

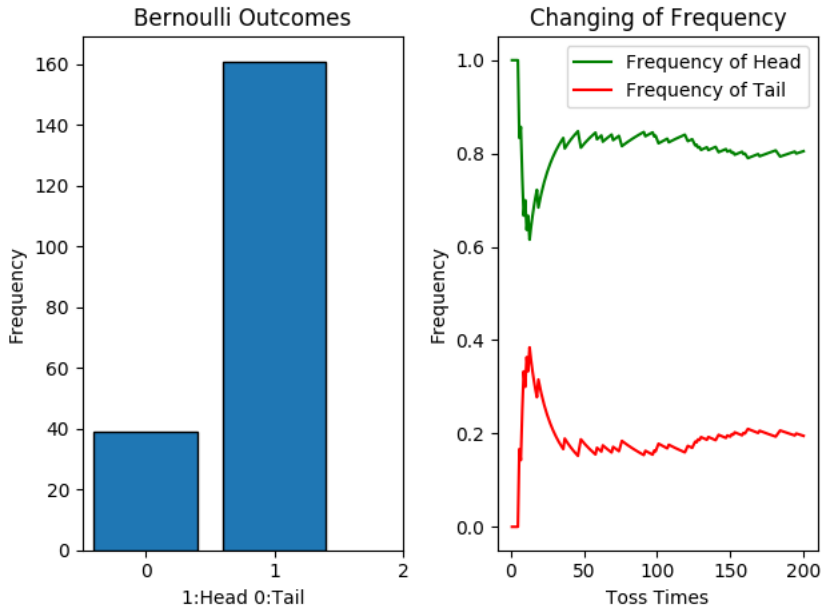
```
import random as rd
import matplotlib.pyplot as plt

# using a list to save the result of each toss, 0: Tail 1: Head
result = []
# toss 200 times
for i in range(200):
    # using generate 0 or 1 p(1)=0.8 p(0)=0.2
    result.append(int(rd.random()+0.8))

# record the longest run of head
longest_run_of_head = 0
# record the length of each consequent head
counter = 0
for ele in result:
    if ele:
        counter += 1
    else:
        # compare each length of consequent head with the previous
longest run of head
        longest_run_of_head = max(longest_run_of_head,counter)
        counter = 0
longest_run_of_head = max(longest_run_of_head,counter)
# print result
print('Longest Run of Head:', longest_run_of_head)
print('Number of Head:', result.count(1))

# Calculate the frequency of appearance of the Head and Tail after
toss
# number of Heads
sum = 0
# frequency of appearance of the Head
prob_head = []
# frequency of appearance of the Tail
prob_tail = []
```

```
toss_times = 0
for ele in result:
    toss_times += 1
    if ele:
        sum += 1
    # frequency of appearance
    prob_head.append(sum/toss_times)
    prob_tail.append(1-sum/toss_times)

# plot a line chart and histogram
plt.figure()

plt.subplot(1,2,1)
plt.hist(result, bins=range(3), rwidth=0.8, edgecolor='black',
align='left')
plt.xticks(range(3))
plt.xlabel('1:Head 0:Tail')
plt.ylabel('Frequency')
plt.title('Bernoulli Outcomes')

plt.subplot(1,2,2)
plt.plot(range(1,toss_times+1), prob_head, color='green',
label='Frequency of Head')
plt.plot(range(1,toss_times+1), prob_tail, color='red',
label='Frequency of Tail')
plt.xlabel('Toss Times')
plt.ylabel('Frequency')
plt.title('Changing of Frequency')
plt.legend()

plt.show()
```

Result: From the simulation the result is:

Longest Run of Head: 17

Number of Head: 161



And we find that, as the tossing time increased, P(HEAD) -> 0.8, P(TAIL) -> 0.2.

4. Heads run lengths – Q3

Requirement: Simulate tossing a fair coin 100 times. Generate a histogram showing the heads run lengths.

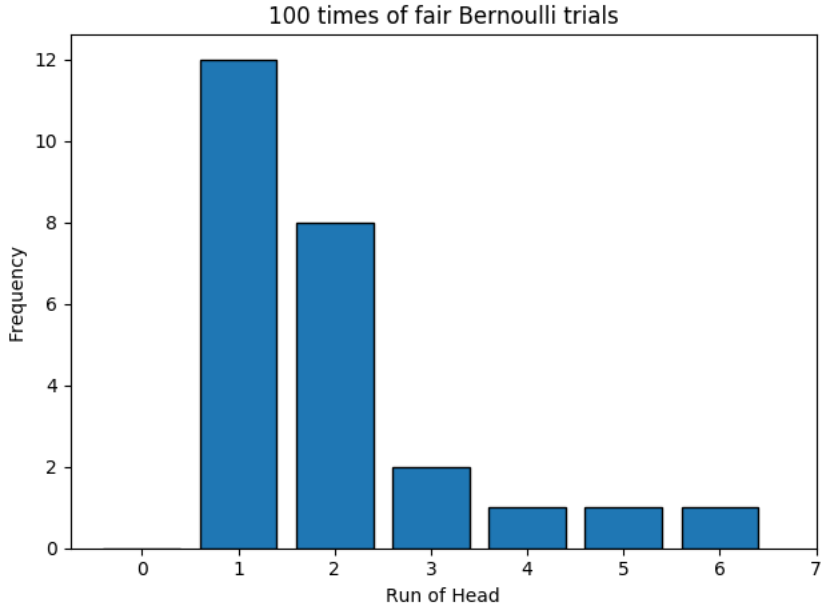
Analysis: When I toss a fair coin 100 times, I use a list to record the result of each toss. Then I traverse the list to record all head run lengths. I think the histogram should be monotonically decreasing, because P (length = i) = 2^{-i}

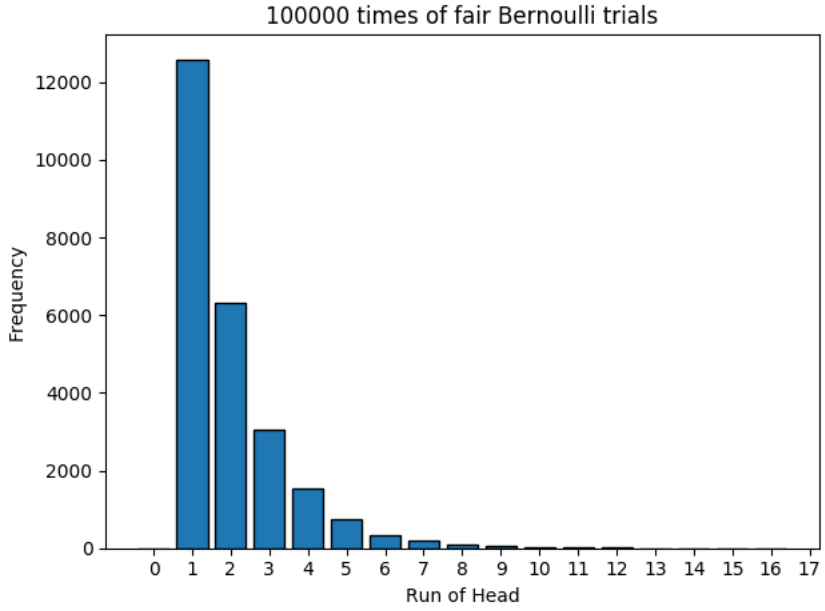
Code:

```
import random as rd
import matplotlib.pyplot as plt

# using a list to save the result of each toss, 0: Tail 1: Head
result = []
# toss 50 times
for i in range(100):
    # using generate 0 or 1 with equal probability
    result.append(int(rd.random()+0.5))
```

```
# record the run of head
run_of_head = []
# record the length of each consequent head
counter = 0
for ele in result:
    if ele:
        counter += 1
    elif counter:
        run_of_head.append(counter)
        counter = 0
if counter:
    run_of_head.append(counter)

plt.figure()
plt.hist(run_of_head, bins=range(max(run_of_head)+2), rwidth=0.8,
edgecolor='black', align='left')
plt.xticks(range(max(run_of_head)+2))
plt.xlabel('Run of Head')
plt.ylabel('Frequency')
plt.title('100 times of fair Bernoulli trials')
plt.show()
```

Result: After simulation, I find the result is decreasing, but 100 times of fair Bernoulli trials cannot generate enough data, so I simulate 10000 times of fair Bernoulli trials.





5. Reach a number of heads – Q4

   Requirement: Simulate tossing a fair coin and count the number of tosses until reaching a user-specified positive number of heads.

Analysis: For a fair coin, P(HEAD) = 0.5. In order to get k heads, we should toss the coin approximately 2*k times. When I make a simulation, I try different target number (range (1,30)).

Code:

```
import random as rd
import matplotlib.pyplot as plt


def main():
    result = []
    # set the user-specified positive number
    threshold = range(1, 30)
    for ele in threshold:
        result.append(toss_to_special_number_of_head(ele))
    print(result)

    plt.figure()
    plt.plot(threshold, result, color='red',marker='o',
label='Simulation')
    plt.plot(threshold, range(2, 60, 2), ls=':', label='y=2x')
    plt.xlabel('Target Number')
    plt.ylabel('Toss Times')
    plt.legend()
    plt.show()


# count the number of tosses until reaching a user-specified
positive number of heads
def toss_to_special_number_of_head(number: int) -> int:
    tossing_times = 0
    counter = 0
    # toss a coin continually
    while True:
        tossing_times += 1
        counter += int(rd.random()+0.5)
        # reach the user-specified positive number
        if counter == number:
            return tossing_times
```

```
if __name__ == '__main__':
    main()
```

Result: From the result, I find that the simulation line is unstable, there exist a large amount of noisy. So, I enlarge the target number (range (1,1000,100)), the result is more stable.

user-specified number	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20	21	22	23	24	25	26	27	28	29
number of toss	3	4	5	10	16	11	15	19	20	20	22	29	22	23	28	35	36	48	34	59	43	46	41	40	50	39	44	54	66