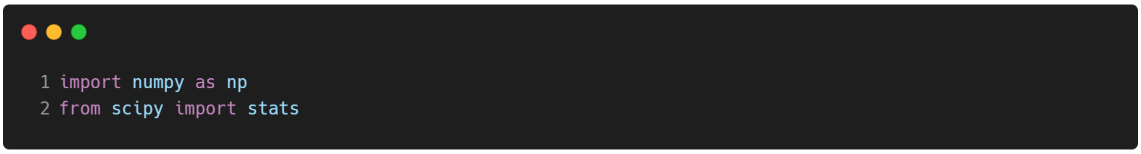
1 Code and Result of Problem 11

Here is a Python program for a fair coin tossed:

1.1 Import Necessary Library

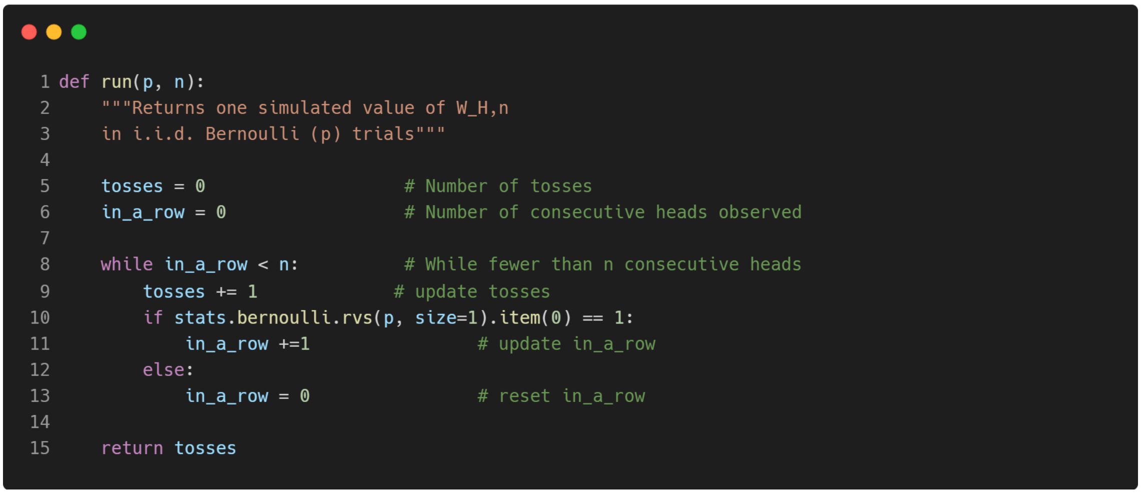


```python
import numpy as np
from scipy import stats
```

1.2 Function Definition

1.2.1 run(p,n)

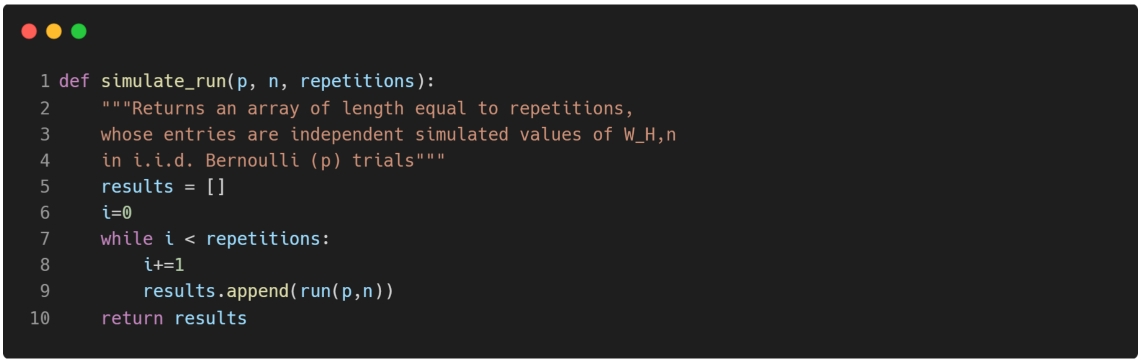
- returns one simulated value of time to see HH...H (n H, denoted by W_H,n)



```python
def run(p, n):
    """Returns one simulated value of W_H,n
    in i.i.d. Bernoulli (p) trials"""

    tosses = 0                   # Number of tosses
    in_a_row = 0                 # Number of consecutive heads observed

    while in_a_row < n:          # While fewer than n consecutive heads
        tosses += 1             # update tosses
        if stats.bernoulli.rvs(p, size=1).item(0) == 1:
            in_a_row +=1                # update in_a_row
        else:
            in_a_row = 0                # reset in_a_row

    return tosses
```

1.2.2 simulate_run(p, n, repetitions)

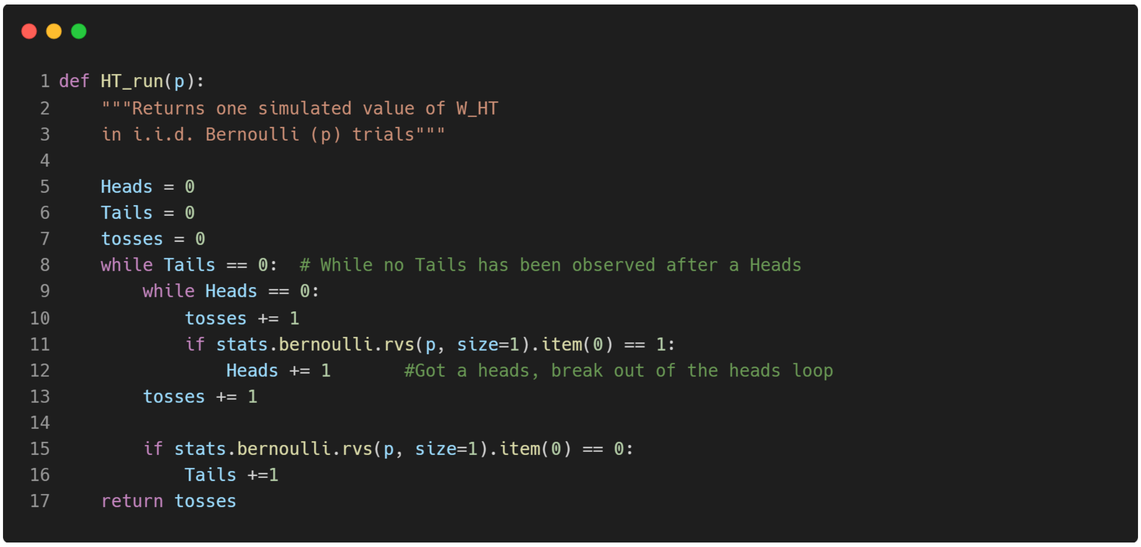
- Returns an array of length equal to repetitions, whose entries are independent simulated values of W_H,n in i.i.d. Bernoulli (p) trials



```python
def simulate_run(p, n, repetitions):
    """Returns an array of length equal to repetitions,
    whose entries are independent simulated values of W_H,n
    in i.i.d. Bernoulli (p) trials"""
    results = []
    i=0
    while i < repetitions:
        i+=1
        results.append(run(p,n))
    return results
```

1.2.3 HT_run(p)

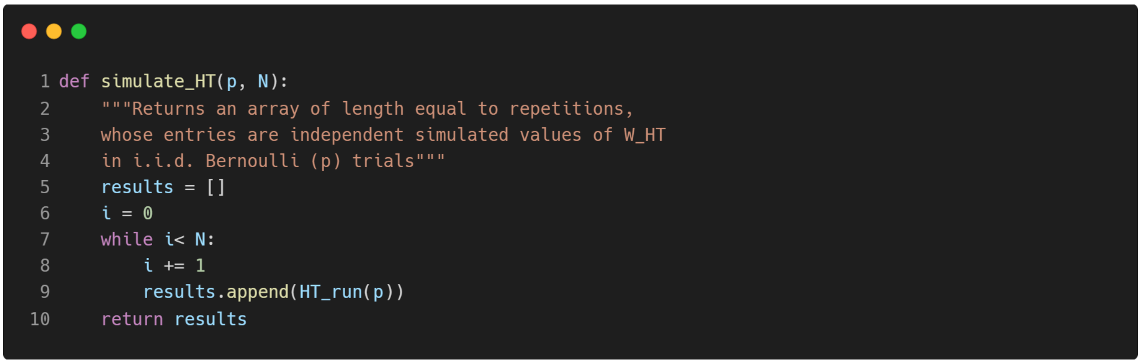
- Returns one simulated value of W_HT in i.i.d. Bernoulli (p) trials



```python
def HT_run(p):
    """Returns one simulated value of W_HT
    in i.i.d. Bernoulli (p) trials"""

    Heads = 0
    Tails = 0
    tosses = 0
    while Tails == 0:  # While no Tails has been observed after a Heads
        while Heads == 0:
            tosses += 1
            if stats.bernoulli.rvs(p, size=1).item(0) == 1:
                Heads += 1       #Got a heads, break out of the heads loop
        tosses += 1

        if stats.bernoulli.rvs(p, size=1).item(0) == 0:
            Tails +=1
    return tosses
```

1.2.4 simulate_HT(p, N)

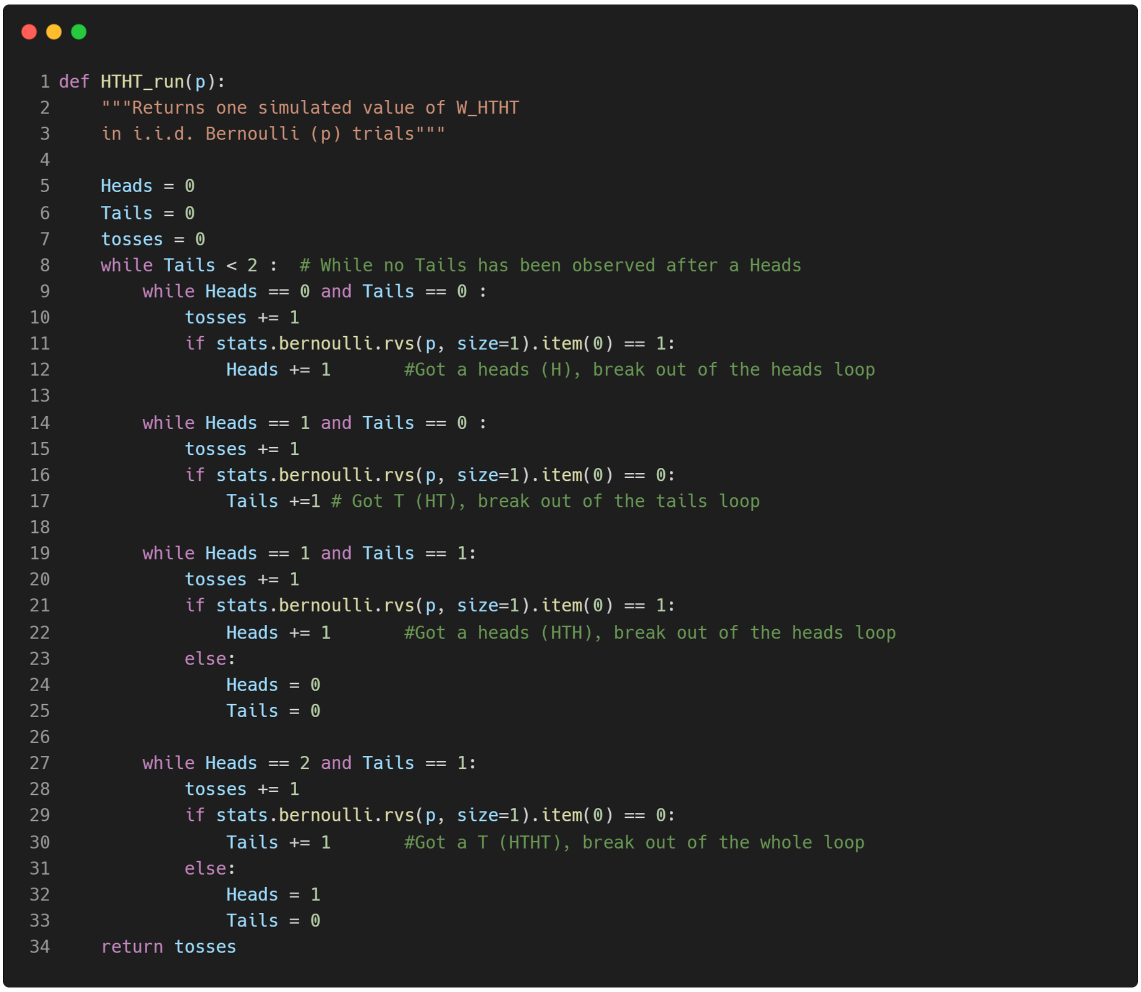
- Returns an array of length equal to repetitions, whose entries are independent simulated values of W_HT in i.i.d. Bernoulli (p) trials



```python
def simulate_HT(p, N):
    """Returns an array of length equal to repetitions,
    whose entries are independent simulated values of W_HT
    in i.i.d. Bernoulli (p) trials"""
    results = []
    i = 0
    while i< N:
        i += 1
        results.append(HT_run(p))
    return results
```

1.2.5 HTHT_run(p)

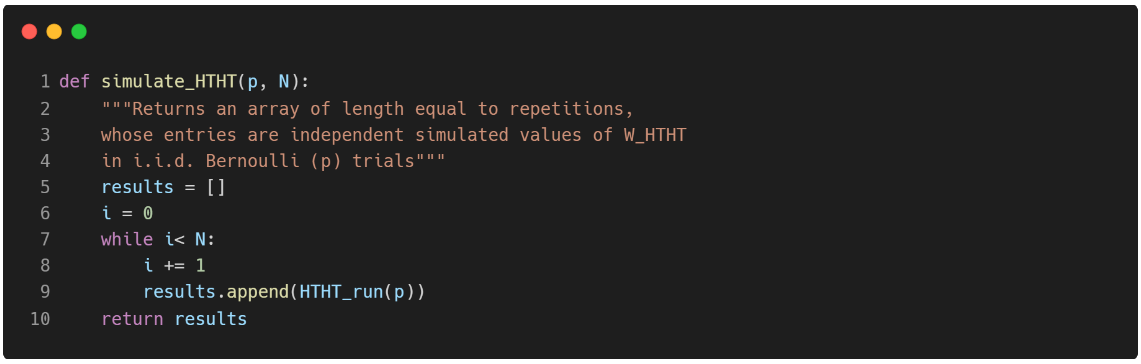
- Returns one simulated value of W_HTHT in i.i.d. Bernoulli (p) trials



```python
def HTHT_run(p):
    """Returns one simulated value of W_HTHT
    in i.i.d. Bernoulli (p) trials"""

    Heads = 0
    Tails = 0
    tosses = 0
    while Tails < 2 :  # While no Tails has been observed after a Heads
        while Heads == 0 and Tails == 0 :
            tosses += 1
            if stats.bernoulli.rvs(p, size=1).item(0) == 1:
                Heads += 1       #Got a heads (H), break out of the heads loop

        while Heads == 1 and Tails == 0 :
            tosses += 1
            if stats.bernoulli.rvs(p, size=1).item(0) == 0:
                Tails +=1 # Got T (HT), break out of the tails loop

        while Heads == 1 and Tails == 1:
            tosses += 1
            if stats.bernoulli.rvs(p, size=1).item(0) == 1:
                Heads += 1       #Got a heads (HTH), break out of the heads loop
            else:
                Heads = 0
                Tails = 0

        while Heads == 2 and Tails == 1:
            tosses += 1
            if stats.bernoulli.rvs(p, size=1).item(0) == 0:
                Tails += 1       #Got a T (HTHT), break out of the whole loop
            else:
                Heads = 1
                Tails = 0
    return tosses
```

1.2.6 simulate_HTHT(p, N)

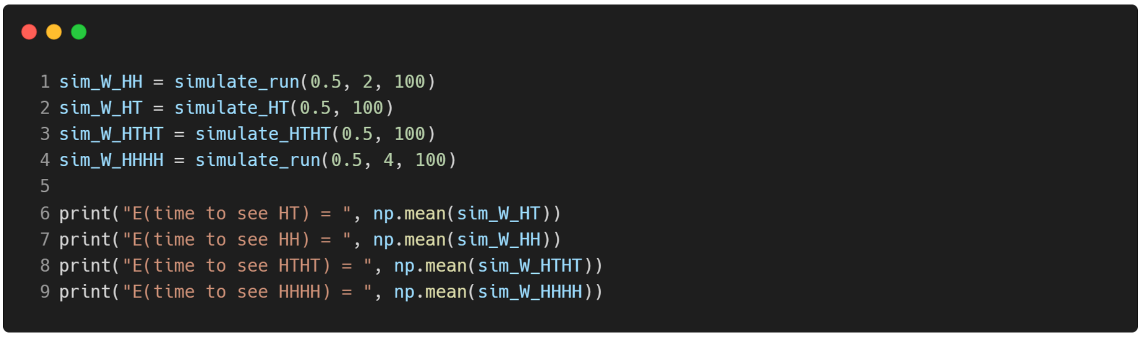
- Returns an array of length equal to repetitions, whose entries are independent simulated values of W_HTHT in i.i.d. Bernoulli (p) trials



```python
def simulate_HTHT(p, N):
    """Returns an array of length equal to repetitions,
    whose entries are independent simulated values of W_HTHT
    in i.i.d. Bernoulli (p) trials"""
    results = []
    i = 0
    while i< N:
        i += 1
        results.append(HTHT_run(p))
    return results
```

1.3 Testing Code (100 repetitions)

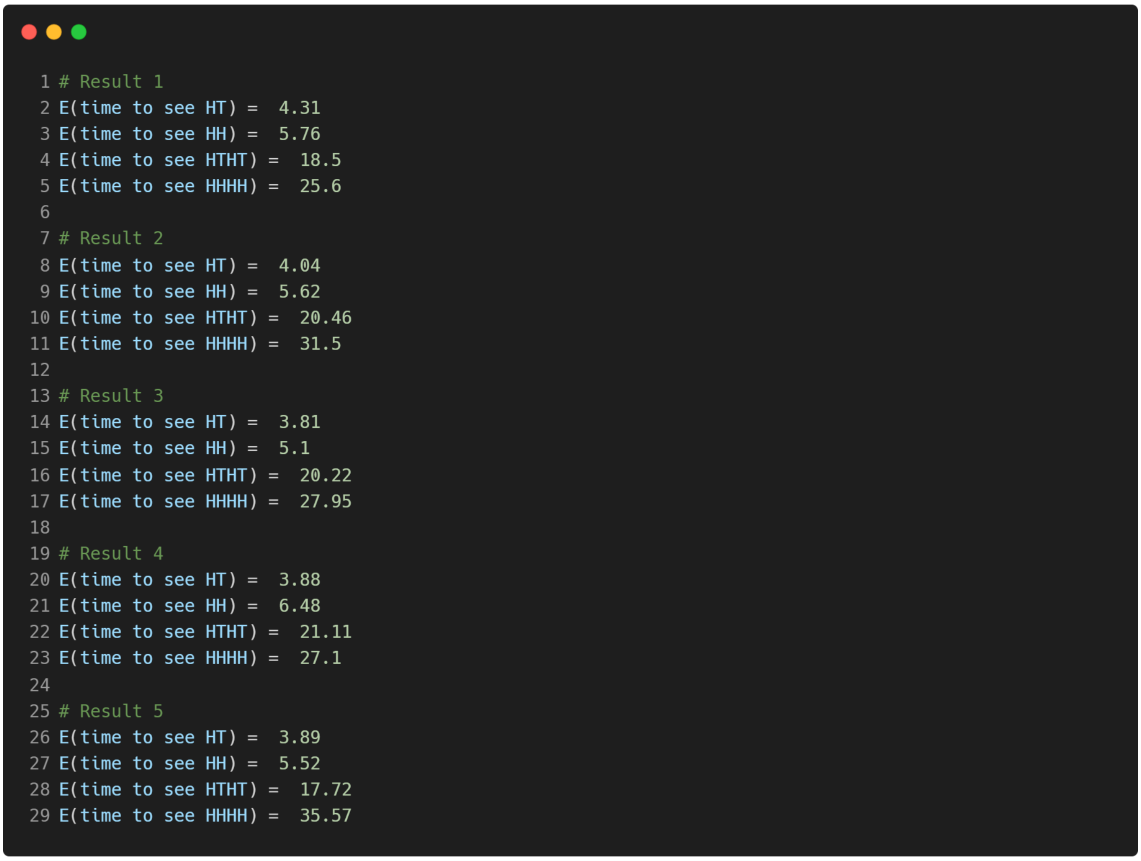
- calculates the mean value of 100 repetitions and prints them on the screen



```
sim_W_HH = simulate_run(0.5, 2, 100)
sim_W_HT = simulate_HT(0.5, 100)
sim_W_HTHT = simulate_HTHT(0.5, 100)
sim_W_HHHH = simulate_run(0.5, 4, 100)

print("E(time to see HT) = ", np.mean(sim_W_HT))
print("E(time to see HH) = ", np.mean(sim_W_HH))
print("E(time to see HTHT) = ", np.mean(sim_W_HTHT))
print("E(time to see HHHH) = ", np.mean(sim_W_HHHH))
```

1.4 Some Result of 100 repetitions test

- Some results of 100 repetitions test are shown below:

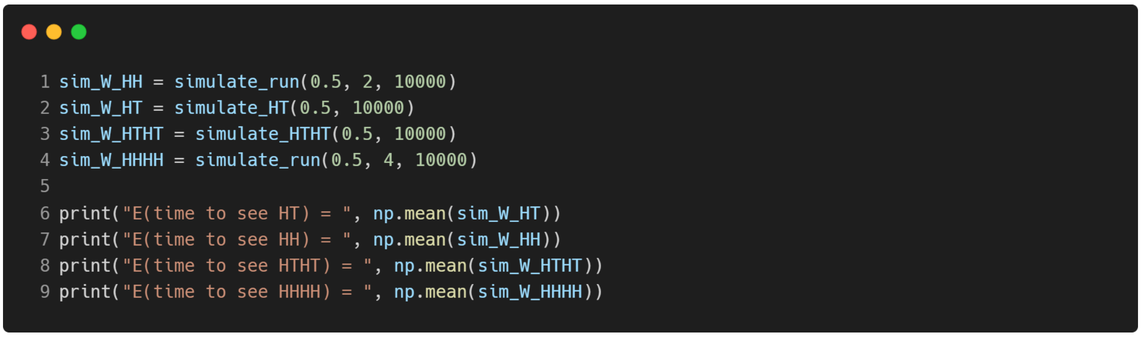


```
# Result 1
E(time to see HT) =  4.31
E(time to see HH) =  5.76
E(time to see HTHT) =  18.5
E(time to see HHHH) =  25.6

# Result 2
E(time to see HT) =  4.04
E(time to see HH) =  5.62
E(time to see HTHT) =  20.46
E(time to see HHHH) =  31.5

# Result 3
E(time to see HT) =  3.81
E(time to see HH) =  5.1
E(time to see HTHT) =  20.22
E(time to see HHHH) =  27.95

# Result 4
E(time to see HT) =  3.88
E(time to see HH) =  6.48
E(time to see HTHT) =  21.11
E(time to see HHHH) =  27.1

# Result 5
E(time to see HT) =  3.89
E(time to see HH) =  5.52
E(time to see HTHT) =  17.72
E(time to see HHHH) =  35.57
```

- Since the repetitions 100 is not large enough, it is hard to conclude and verify the Expectation to see a specific string. Then let's set a 10000 repetitions.

1.5 Testing Code (10000 repetitions)

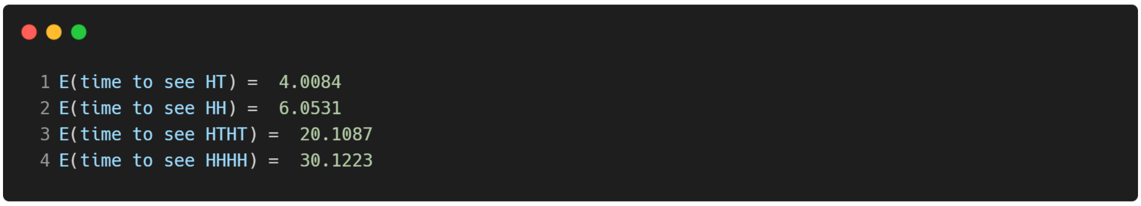
- calculates the mean value of 10000 repetitions and prints them on the screen



```
sim_W_HH = simulate_run(0.5, 2, 10000)
sim_W_HT = simulate_HT(0.5, 10000)
sim_W_HTHT = simulate_HTHT(0.5, 10000)
sim_W_HHHH = simulate_run(0.5, 4, 10000)

print("E(time to see HT) = ", np.mean(sim_W_HT))
print("E(time to see HH) = ", np.mean(sim_W_HH))
print("E(time to see HTHT) = ", np.mean(sim_W_HTHT))
print("E(time to see HHHH) = ", np.mean(sim_W_HHHH))
```

1.6 One Result of 10000 repetitions test

- One of the results of 10000 repetitions test is shown below



```
E(time to see HT) =  4.0084
E(time to see HH) =  6.0531
E(time to see HTHT) =  20.1087
E(time to see HHHH) =  30.1223
```

- According to the result, we can see that

$$E[\text{time to see } HT] \approx 4$$

$$E[\text{time to see } HH] \approx 6$$

$$E[\text{time to see } HTHT] \approx 20$$

$$E[\text{time to see } HHHH] \approx 30$$