Experimental Mathematics: Betting Games

Project supervisor: Kerchev George

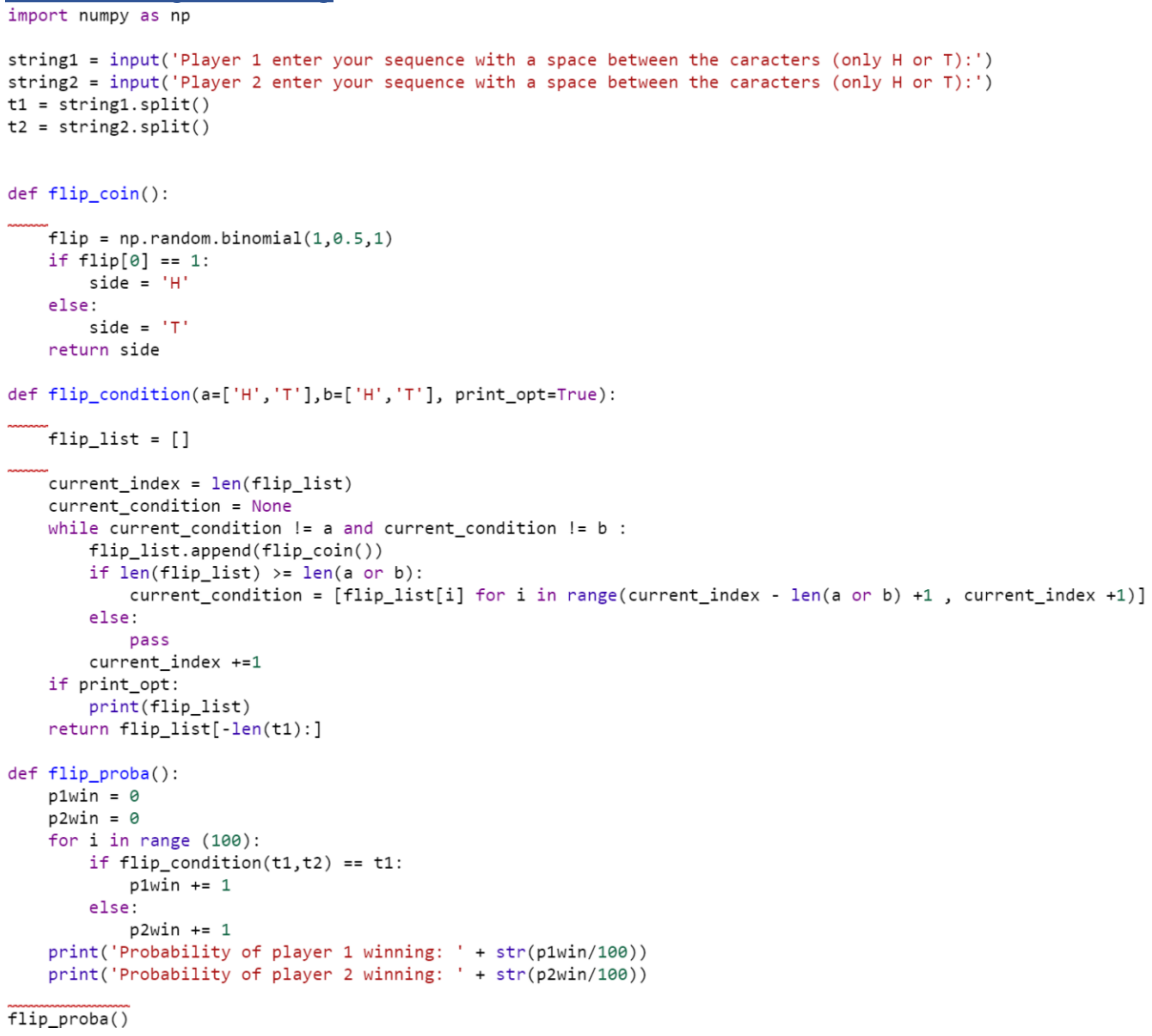
Table of contents

Project topic and goal 3
Part I: Programming 4
Part ll: Markov Chain 9
Sequences of two unbiased flips 9
Sequences of two biased flips 12
Sequences of three unbiased flips 13
Sequences of three biased flips 17
Part III: Hypothesis testing 22

Project topic and goal

The topic of the project is ´Betting games´. We assumed the following situation: two players were playing a coin flip game. Each player must choose a certain sequence of flips for example HHH or THH. If the sequence of Player 1 occurs first, he wins. Otherwise Player 2 wins. The goal of the project is to find the probabilities of each player winning, with a certain triplet that they choose, by using the Markov chain and comparing these results with the results given by a computer simulation. By that, one can later use the results to find the triplet with the highest probability of winning for Player 2 if Player 1 has already chosen a triplet.

Part I: Programming



```
import numpy as np

string1 = input('Player 1 enter your sequence with a space between the caracters (only H or T):')
string2 = input('Player 2 enter your sequence with a space between the caracters (only H or T):')
t1 = string1.split()
t2 = string2.split()


def flip_coin():

    flip = np.random.binomial(1,0.5,1)
    if flip[0] == 1:
        side = 'H'
    else:
        side = 'T'
    return side

def flip_condition(a=['H','T'],b=['H','T'], print_opt=True):

    flip_list = []

    current_index = len(flip_list)
    current_condition = None
    while current_condition != a and current_condition != b :
        flip_list.append(flip_coin())
        if len(flip_list) >= len(a or b):
            current_condition = [flip_list[i] for i in range(current_index - len(a or b) +1 , current_index +1)]
        else:
            pass
        current_index +=1
    if print_opt:
        print(flip_list)
    return flip_list[-len(t1):]

def flip_proba():
    p1win = 0
    p2win = 0
    for i in range (100):
        if flip_condition(t1,t2) == t1:
            p1win += 1
        else:
            p2win += 1
    print('Probability of player 1 winning: ' + str(p1win/100))
    print('Probability of player 2 winning: ' + str(p2win/100))

flip_proba()
```

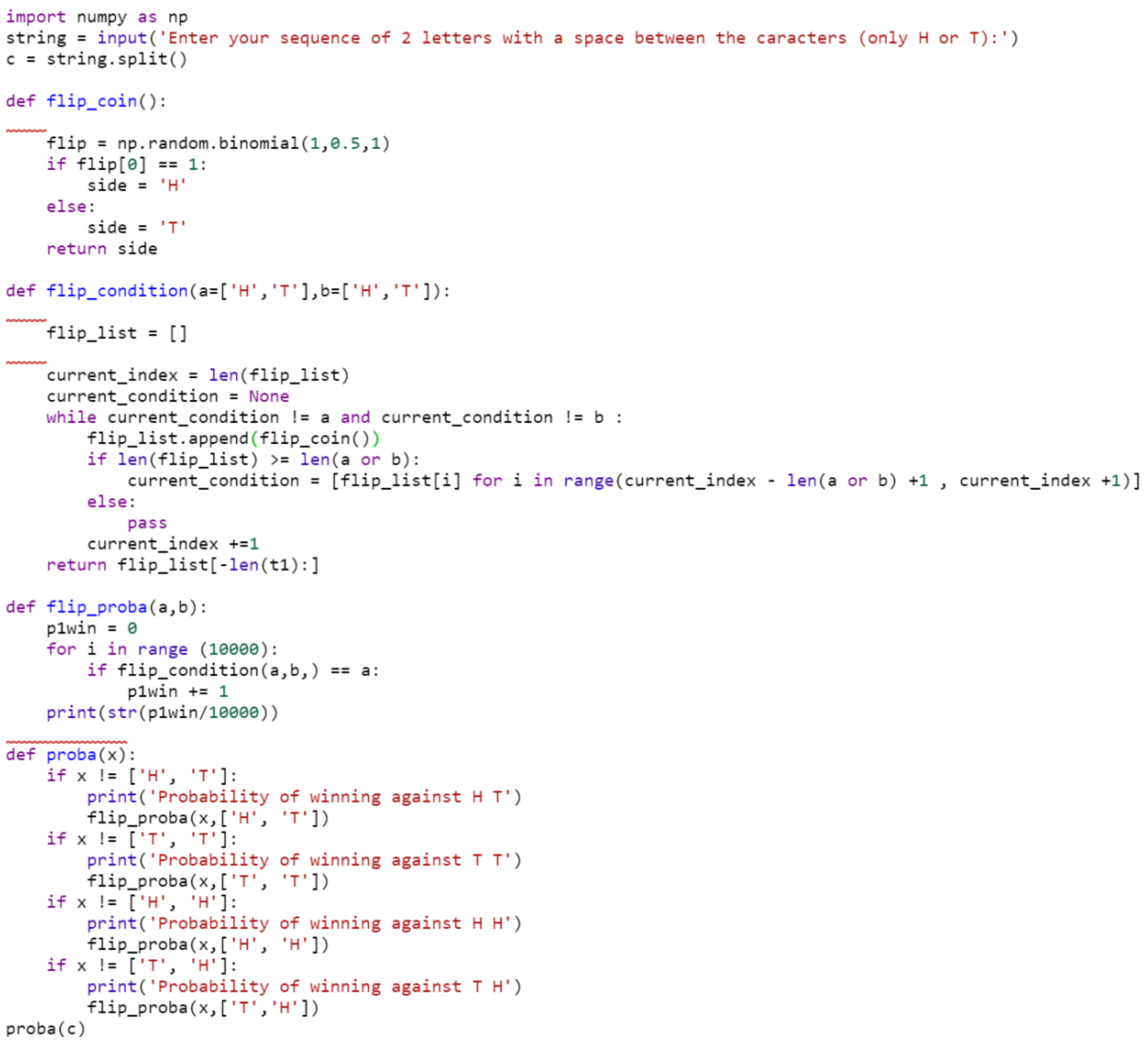
The first part of the program will simulate a series of coin flips and will stop when either the sequence of Player 1 or Player 2 occurs. It will repeat this process several times. The user can edit the amount of times that he wants to process to repeat itself. The bigger the number of simulations is, the more precise the results will be, and they will more likely match the theoretical results obtained by using the Markov chain.

The program has two inputs, the first one for Player 1 to enter his sequence and the second one for Player 2.

The first function we have, will simulate a regular coin flip with a probability of ½ for obtaining either heads or tails. The program will return a 1 for heads or a 0 for tails, which will then be returned as a 'H' or a 'T' for the user.

The second function gives the program the flip conditions and tells it when to stop the process of coin flipping. Both players can only use the letters 'H' or 'T' for their sequence. The function then tells the program to stop the simulation when either the first or the second sequence occurs. For that, the function always compares the last n elements of the list with the two sequences chosen by the players, where n stands for the length of these sequence and then returns the last n elements of the list. The elements of the list are the results obtained by the ‘flip_coin’ function, so ‘H’ or ‘T’. The function can also display a list of the simulation and the elements of the list if ‘print_opt’ is set as ‘True’.

The ‘flip_proba’ function calculates the probability of winning for each of the two players. It uses the ‘flip_condition’ function and runs it several amount of times. The user can change the number of simulations to a value he likes. The bigger this value, the more precise the results become. We have two local variables ‘p1win’ for Player 1 and ‘p2win’ for Player 2. Each time when Player 1 wins, the variable ‘p1win’ increases by 1 and if not ‘p2win’ increases by 1. At the end it calculates and prints the probability of the two players winning by dividing ‘p1win’ respectively ‘p2win’ by the number of simulations.

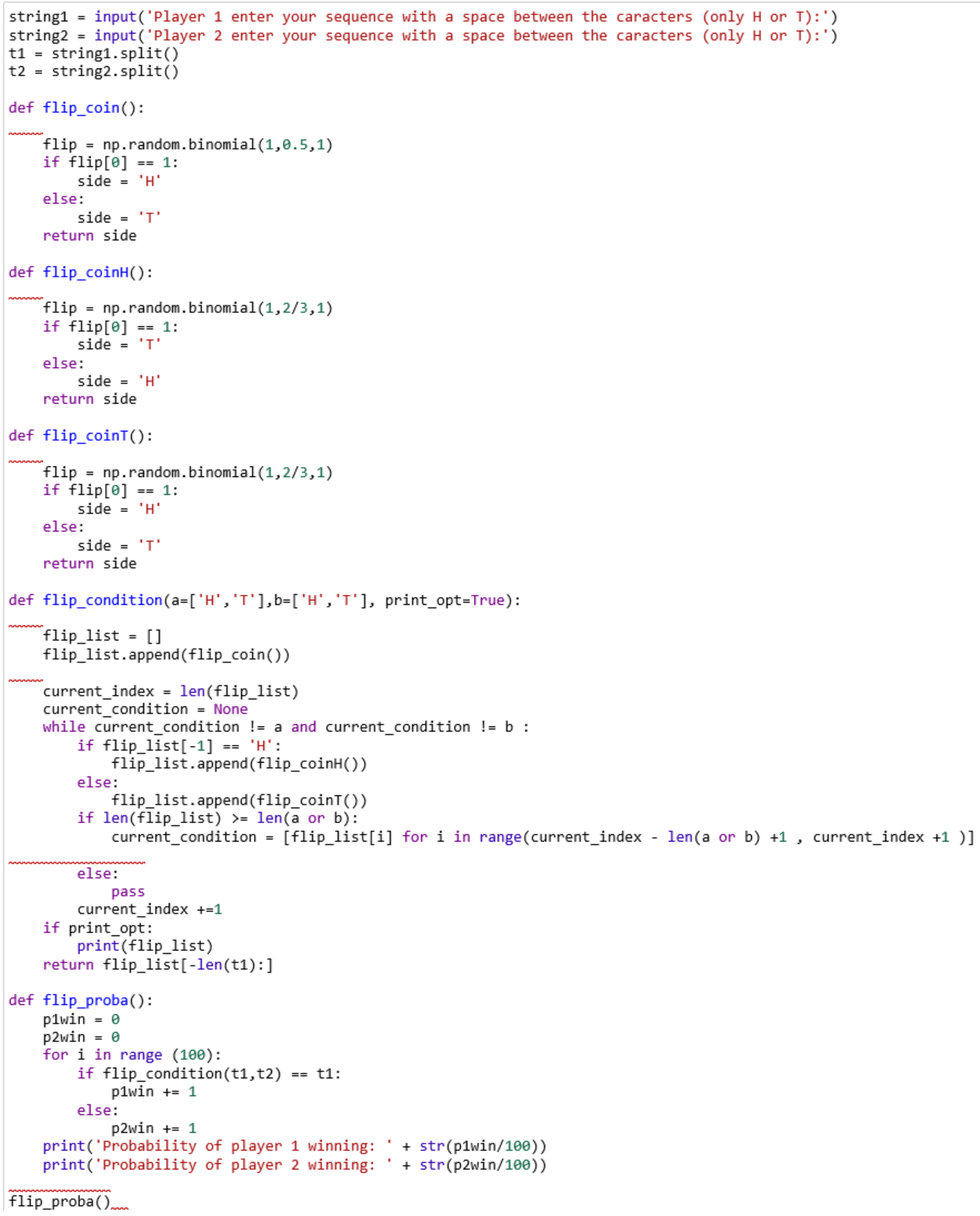


```
import numpy as np
string = input('Enter your sequence of 2 letters with a space between the caracters (only H or T):')
c = string.split()

def flip_coin():

    flip = np.random.binomial(1,0.5,1)
    if flip[0] == 1:
        side = 'H'
    else:
        side = 'T'
    return side

def flip_condition(a=['H','T'],b=['H','T']):

    flip_list = []

    current_index = len(flip_list)
    current_condition = None
    while current_condition != a and current_condition != b :
        flip_list.append(flip_coin())
        if len(flip_list) >= len(a or b):
            current_condition = [flip_list[i] for i in range(current_index - len(a or b) +1 , current_index +1)]
        else:
            pass
        current_index +=1
    return flip_list[-len(t1):]

def flip_proba(a,b):
    p1win = 0
    for i in range (10000):
        if flip_condition(a,b,) == a:
            p1win += 1
    print(str(p1win/10000))

def proba(x):
    if x != ['H', 'T']:
        print('Probability of winning against H T')
        flip_proba(x,['H', 'T'])
    if x != ['T', 'T']:
        print('Probability of winning against T T')
        flip_proba(x,['T', 'T'])
    if x != ['H', 'H']:
        print('Probability of winning against H H')
        flip_proba(x,['H', 'H'])
    if x != ['T', 'H']:
        print('Probability of winning against T H')
        flip_proba(x,['T','H'])
proba(c)
```

In the second part of the program, the user enters a sequence of only 2 letters, either ‘H’ or ‘T’ and the program then shows the probabilities of the sequence entered by the user against all the other sequences of only 2 letters.

To achieve this, we’ve got the same ‘flip_coin’ function as in the first part, in the function ‘flip_condition’ we only remove ‘if print_opt: print(flip_list)’ at the end, so that the program doesn’t show all the coin flips that were simulated. The function ‘flip_proba(a,b)’ this time has two inputs, ‘a’ and ‘b‘, the rest of the program stays the same as in the first part, with the difference that this time we’ve got a range of 10000 so that the results are more precise. Also, the function only prints the probability of ‘p1win’.

The last function ‘proba(x)’ takes the input from the user and compares the sequence to every possible sequence of 2 letters, which means: ‘H T’, ‘H H’, ‘T H’ and ‘T T’. Firstly, it checks if the input is the same as the sequence used in that line, if not, we use ‘flip_proba’ to print the probability of the input winning.



```
string1 = input('Player 1 enter your sequence with a space between the caracters (only H or T):')
string2 = input('Player 2 enter your sequence with a space between the caracters (only H or T):')
t1 = string1.split()
t2 = string2.split()

def flip_coin():

    flip = np.random.binomial(1,0.5,1)
    if flip[0] == 1:
        side = 'H'
    else:
        side = 'T'
    return side

def flip_coinH():

    flip = np.random.binomial(1,2/3,1)
    if flip[0] == 1:
        side = 'T'
    else:
        side = 'H'
    return side

def flip_coinT():

    flip = np.random.binomial(1,2/3,1)
    if flip[0] == 1:
        side = 'H'
    else:
        side = 'T'
    return side

def flip_condition(a=['H','T'],b=['H','T'], print_opt=True):

    flip_list = []
    flip_list.append(flip_coin())

    current_index = len(flip_list)
    current_condition = None
    while current_condition != a and current_condition != b :
        if flip_list[-1] == 'H':
            flip_list.append(flip_coinH())
        else:
            flip_list.append(flip_coinT())
        if len(flip_list) >= len(a or b):
            current_condition = [flip_list[i] for i in range(current_index - len(a or b) +1 , current_index +1 )]

        else:
            pass
        current_index +=1
    if print_opt:
        print(flip_list)
    return flip_list[-len(t1):]

def flip_proba():
    p1win = 0
    p2win = 0
    for i in range (100):
        if flip_condition(t1,t2) == t1:
            p1win += 1
        else:
            p2win += 1
    print('Probability of player 1 winning: ' + str(p1win/100))
    print('Probability of player 2 winning: ' + str(p2win/100))

flip_proba()
```

The third part of the program is about unbiased coin flips. The first coin flip has a probability of ½ for either ‘H’ or ‘T’, but if you get an ‘H’, the probability to get another ‘H’ in the next coin flip decreases to 1/3, the same applies for ‘T’. This means that the probability changes a lot during the 100 coin flips and the chance to get the same side more than once in a row is a bit lower. So, this part of the program doesn’t differ much from the first part. We just add two functions, ‘flip_coinH’, where the chance of getting an ‘H’ is 1/3, or as we wrote it, the chance of getting ‘T’ is 2/3, and ‘flip_coinT’, where the chance of getting an ‘H’ is 2/3.

The last thing that changes is the function ‘flip_condition’. We take the same function from the first part of the program and just add the 4th line ‘flip_list.append(flip_coin())’.

This line adds an ‘H’ or a ‘T’ to the empty ‘flip_list’ with a probability of ½ for each of them. After this, the only other change begins in line 9, where we look at the last element of the list with ‘flip_list[-1] == 'H'’. So, if the last element is an ‘H’, we use ‘flip_coinH’ for the next coin flip, and if it’s not an ‘H’, we use ‘flip_coinT’.

Code link: https://1drv.ms/b/s!Ak4c-gsTBHG1qSTld6ZdjkVmyhV5?e=t8Iyoa

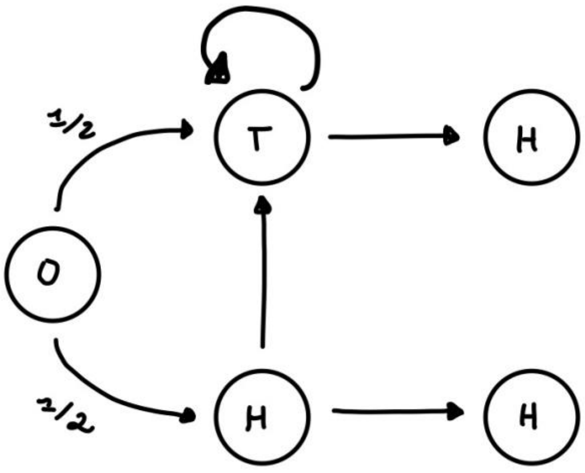
Part II: Markov Chain

Sequences of two unbiased flips

The table below shows the odds of the sequences on the left side winning against the sequences on the top. These probabilities are obtained by using the Markov Chain procedure, which we will show in hand of different examples.

	HH	HT	TH	TT
HH	▨	1/2	1/4	1/2
HT	1/2	▨	1/2	3/4
TH	3/4	1/2	▨	1/2
TT	1/2	1/4	1/2	▨

The first example is the sequence TH against the sequence HH.



1: TH

2: HH

p_0 : probability that player 1 wins against player 2 from the start

$$p_0 = \frac{1}{2}\,p_T + \frac{1}{2}\,p_H$$

If the first toss was a T, player 1 would win if the next toss would equal in an H else another T wouldn't benefit player 2 and player 1 would be at the same stage.

$$p_T = \frac{1}{2}\,p_T + \frac{1}{2}\cdot 1$$

However, if the first toss was an H, a second H would result in the win for player 2, however a T would be beneficial for player 1.

$$p_H = \frac{1}{2}\,p_T + \frac{1}{2}\cdot 0$$

$$\begin{cases} p_T = 2\,p_H \\ p_T = 1 \end{cases}$$

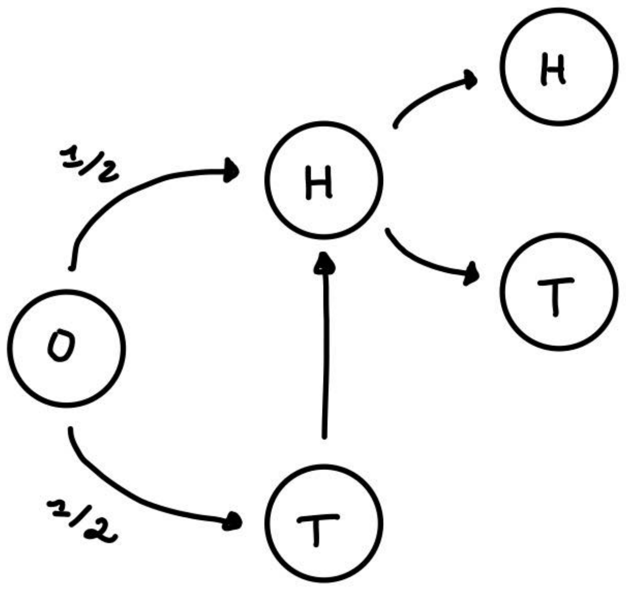
$$\Rightarrow p_H = \tfrac{1}{2}$$

It makes sense for p_T to be 1 as at that point it is impossible for player 2 to get two consecutive H's to win as player 1 just needs one H.

$$p_0 = \frac{1}{2}\cdot 1 + \frac{1}{2}\cdot\frac{1}{2} = \frac{3}{4}$$

In this case, we see that TH is more probable to win against HH.

For the second example, we will look at HH against HT.



1: HH

2: HT

p_0 : probability that player 1 wins against player 2 from the start

$$p_0 = \frac{1}{2}\, p_T + \frac{1}{2}\, p_H$$

An T after the first toss doesn't benefit any of the two players if we look at the sequences.

$$p_T = \frac{1}{2}\, p_T + \frac{1}{2}\, p_H$$

An H after the first toss is favorable for both, the next toss is decisive for both players.

$$p_H = \frac{1}{2} \cdot 1 + \frac{1}{2} \cdot 0$$

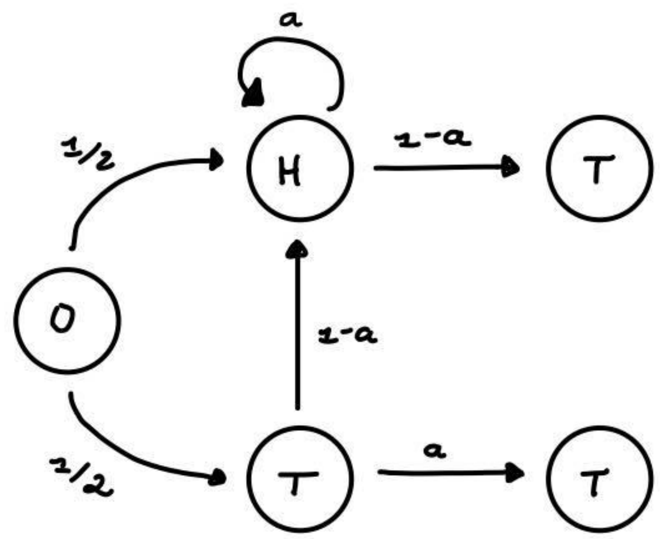
=> $p_H = p_T = \frac{1}{2}$

$$p_0 = \frac{1}{2} \cdot \frac{1}{2} + \frac{1}{2} \cdot \frac{1}{2} = \frac{1}{2}$$

This result is clear.

Sequences of two biased flips

In this section, we will take the probability as a variable. The probability of getting an H after having gotten an H equals to a and for getting T is 1-a. The same stands for T. We must consider a different of 1 and 0 else this experiment wouldn't make sense.



1: TT

2: HT

$$p_0 = \frac{1}{2}\,p_H + \frac{1}{2}\,p_T$$

$$p_H = a \cdot p_H + (1-a)\cdot 0 \;\Rightarrow\; p_H = 0$$

As soon as we get an H, player 2 is certain to win.

$$p_T = a + (1-a)\,p_H \Rightarrow\; p_T = a$$

The probability of player 1 winning after getting an T depends on the probability a.

$$p_0 = \frac{1}{2}\,a$$

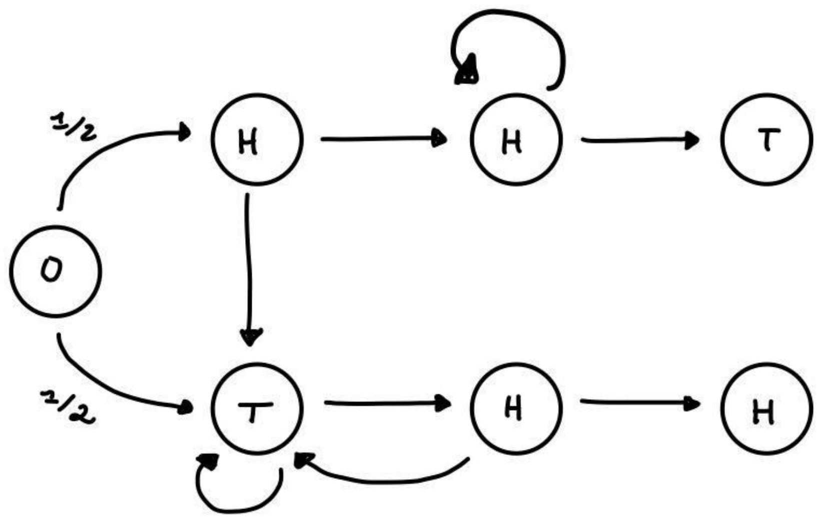
No matter the probability a, player 1 is more likely to win because getting an H after the first or second toss instantly makes it impossible for player 2 to win. Player 2 needs to get two consecutive T's at the beginning else it is not possible.

Sequences of three unbiased flips

The table below is to be read the same way as the table before. However, here we consider sequences of three flips. We will calculate some of these probabilities below using the Markov Chain procedure.

	HHH	HHT	HTH	HTT	THH	THT	TTH	TTT
HHH	▨	1/2	2/5	2/5	1/8	5/12	3/10	1/2
HHT	1/2	▨	2/3	2/3	1/4	5/8	1/2	7/10
HTH	3/5	1/3	▨	1/2	1/2	1/2	3/8	7/12
HTT	3/5	1/3	1/2	▨	1/2	1/2	3/4	7/8
THH	7/8	3/4	1/2	1/2	▨	1/2	1/3	3/5
THT	7/12	3/8	1/2	1/2	1/2	▨	1/3	3/5
TTH	7/10	1/2	5/8	1/4	2/3	2/3	▨	1/2
TTT	1/2	3/10	5/12	1/8	2/5	2/5	1/2	▨

The first example is HHT against THH.



1: HHT

2: THH

$$p_0 = \frac{1}{2}\, p_H + \frac{1}{2}\, p_T$$

$$p_H = \frac{1}{2}\, p_{HH} + \frac{1}{2}\, p_T \quad (1)$$

$$p_T = \frac{1}{2}\, p_{TH} + \frac{1}{2}\, p_T \quad (2)$$

$$p_{HH} = \frac{1}{2}\, p_{HH} + \frac{1}{2} \cdot 1 \;\Rightarrow\; p_{HH} = 1 \quad (3)$$

If player one finds himself at this stage, he is sure to win.

$$p_{TH} = \frac{1}{2}\, p_T + \frac{1}{2} \cdot 0 = \frac{1}{2}\, p_T \quad (4)$$

(4) in (2) :

$$p_T = \frac{1}{2} \cdot \frac{1}{2}\, p_T + \frac{1}{2}\, p_T \;\Rightarrow\; p_T = 0 \quad (5)$$

This result is clear, as player two wins as soon as he gets a T because player 1 cannot get their sequence without player 2 winning.

(5) and (3) in (1) :

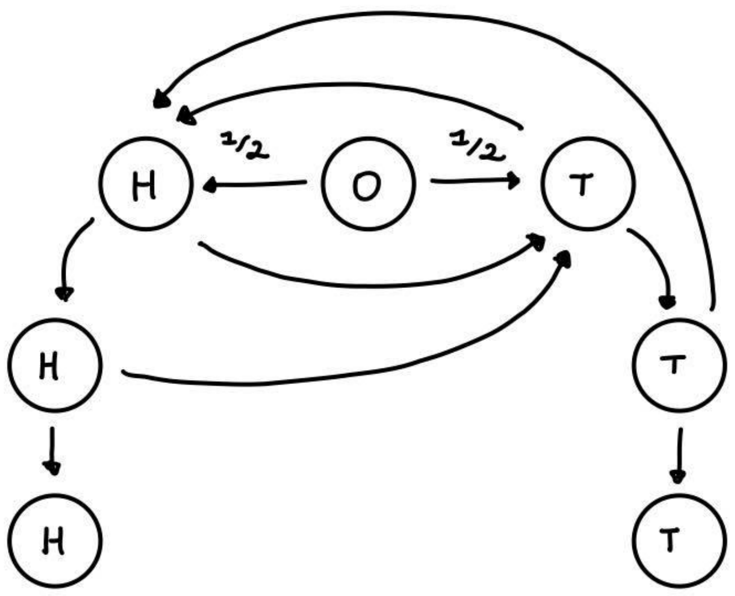
$$p_H = \frac{1}{2}$$

We can explain this simply by saying that if at this stage the next toss equals in an H player 1 wins, else getting an T means player 2 wins.

$$p_0 = \frac{1}{4}$$

We see that THH in this case is more favorable to win against HHT.

The second example is HHH against TTT.



1: HHH

2: TTT

$$p_0 = \frac{1}{2}\, p_H + \frac{1}{2}\, p_T$$

$$p_T = \frac{1}{2}\, p_{TT} + \frac{1}{2}\, p_H \quad (1)$$

$$p_H = \frac{1}{2}\, p_{HH} + \frac{1}{2}\, p_T \quad (2)$$

$$p_{HH} = \frac{1}{2} \cdot 1 + \frac{1}{2}\, p_T \quad (3)$$

$$p_{TT} = \frac{1}{2} \cdot 0 + \frac{1}{2}\, p_H = \frac{1}{2} p_H \quad (4)$$

(3) in (2) :

$$p_H = \frac{1}{2}\left(\frac{1}{2} + \frac{1}{2}p_T\right) + \frac{1}{2}\,p_T = \frac{1}{4} + \frac{3}{4}p_T \quad (5)$$

(4) in (1) :

$$p_T = \frac{1}{2}\cdot\frac{1}{2}p_H + \frac{1}{2}\,p_H = \frac{3}{4}p_H \quad (6)$$

(6) in (5) :

$$p_H = \frac{1}{4} + \frac{3}{4}\cdot\frac{3}{4}p_H = \frac{1}{4} + \frac{9}{16}p_H$$

$$\Leftrightarrow \frac{7}{16}p_H = \frac{1}{4}$$

$$\Leftrightarrow p_H = \frac{4}{7} \quad (7)$$

(7) in (6) :

$$p_T = \frac{3}{4}\cdot\frac{4}{7} = \frac{3}{7}$$

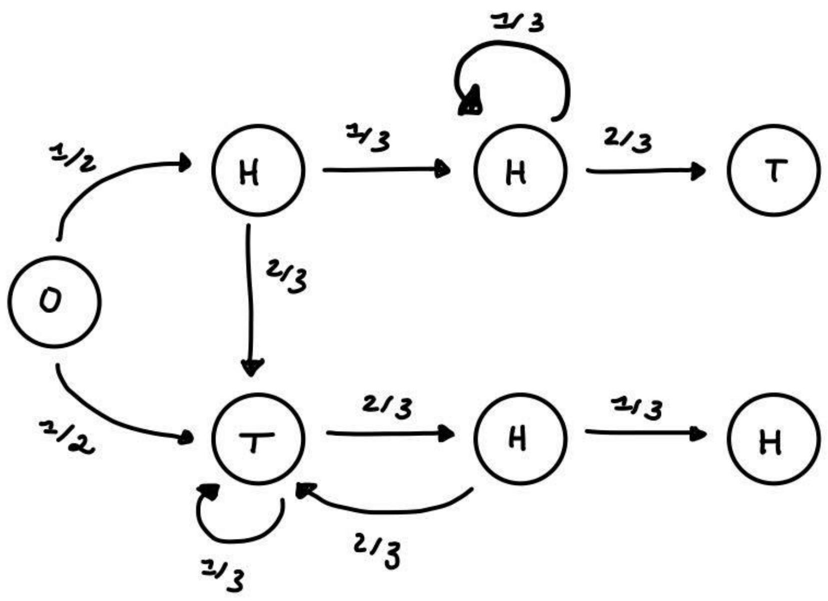
$$p_0 = \frac{1}{2}\cdot\frac{4}{7} + \frac{1}{2}\cdot\frac{3}{7} = \frac{1}{2}$$

For this example, not much explanation is needed. The result is clear.

Sequences of three biased flips

In this section, we consider a biased coin toss. For the first coin toss, the probability still corresponds to a unbiased coin toss. However, for the second toss the probability changes. If the player received an H in the first toss, for the second toss, the probability of getting H again goes down to $\frac{1}{3}$. It's the same for getting T after the first toss.

The first example is HHT against THH.



1: HHT

2: THH

$$p_0 = \frac{1}{2}\, p_H + \frac{1}{2}\, p_T$$

$$p_H = \frac{1}{3}\, p_{HH} + \frac{2}{3}\, p_T \quad (1)$$

After getting H after the first, the player either gets a T or an H which would be better for player 1 as it would correspond to his sequence.

$$p_T = \frac{1}{3}\, p_T + \frac{2}{3}\, p_{TH} \quad (2)$$

Getting a T in the second toss wouldn't change anything, however, an H would correspond to the sequence of player 2 and is also more probable.

$$p_{HH} = \frac{2}{3} \cdot 1 + \frac{1}{3}\, p_{HH} \;\Rightarrow\; p_{HH} = 1 \quad (3)$$

If the last two tosses were two consecutive H, the player 1 would win by getting an T with probability of 2/3. By getting an H, it would still benefit player 1. We see that if player 1 achieves two consecutive H's, he is certain to win, which is clear as the no matter what the result is it won't benefit player 2.

$$p_{TH} = \frac{2}{3}\, p_T + \frac{1}{3} \cdot 0 = \frac{2}{3}\, p_T \quad (4)$$

In this case, player 2 would win if he gets an H. If he gets a T, he would go back to the start of his sequence. By solving the equations, we get that as soon as the coin toss equals a T and player 1 hasn't won, player 2 is certain to win as there is no way for player one to win because as soon as two consecutive H's are tossed player 2 wins knowing an T was tossed already.

(4) in (2):

$$p_T = \frac{1}{3}\, p_T + \frac{4}{9}\, p_T$$

$$\Rightarrow\; p_T = 0 \quad (5)$$

$$\Rightarrow\; p_{TH} = 0$$

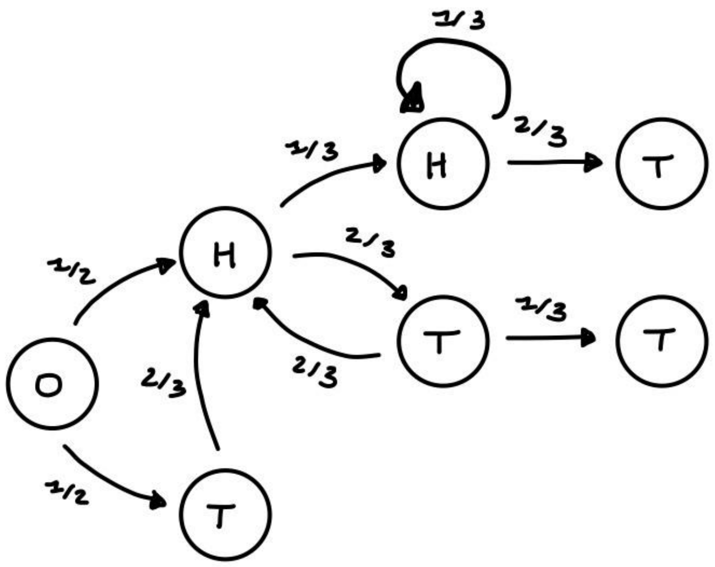
(3) and (5) in (1) :

$$p_H = \frac{1}{3} \cdot 1 \,+\, \frac{2}{3} \cdot 0 = \frac{1}{3}$$

$$p_0 = \frac{1}{2} \cdot \frac{1}{3} + \frac{1}{2} \cdot 0 = \frac{1}{6}$$

The probability makes sense, as player two is more likely to win because the only way player 1 wins is if the very first two tosses are two consecutive H's, else getting a T equals in the certain victory of player two. This result does correspond to the result obtained by using our program.

The second example is HHT against HTT.



1: HHT

2: HTT

$$p_0 = \frac{1}{2}\,p_H + \frac{1}{2}\,p_T$$

$$p_H = \frac{1}{3}\,p_{HH} + \frac{2}{3}\,p_{HT} \quad (1)$$

Getting an H after the first toss benefits both players. However, the next toss is more favorable for player two as the probability of getting T after an H is higher.

$$p_T = \frac{1}{3}\,p_T + \frac{2}{3}\,p_H \;\Leftrightarrow\; p_T = p_H$$

Getting a T after the first toss does not benefit any of the two players. The equality then does make sense.

$$p_{HH} = \frac{2}{3}\cdot 1 + \frac{1}{3}\,p_{HH} \;\Rightarrow\; p_{HH} = 1$$

Two consecutive H's equals in the certain victory for player 1 which is clear.

$$p_{HT} = \frac{2}{3}\,p_H + \frac{1}{3}\cdot 0 = \frac{2}{3}\,p_H$$

Player two wins in this case if he receives a T which is less likely. However, getting an H cancels his sequence and we find ourselves back at the stage p_H for which both can still win.

Using the information, we know in (1) we get:

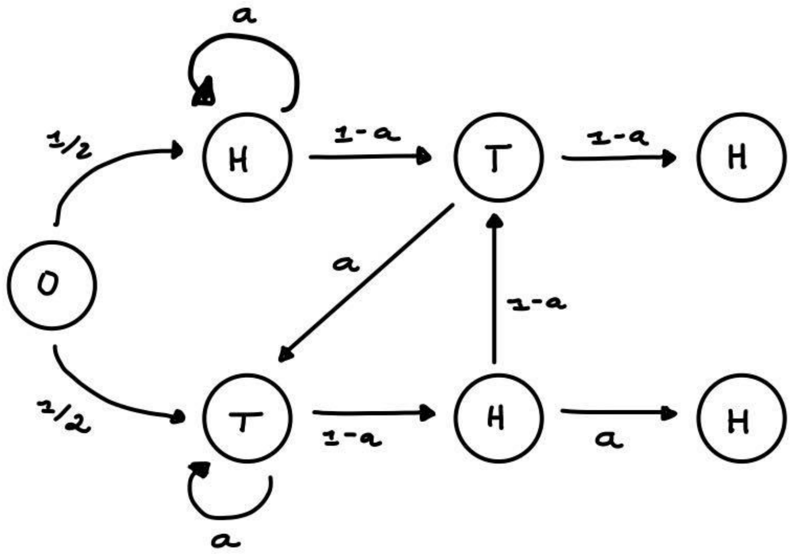
$$p_H = \frac{1}{3}\cdot 1 + \frac{2}{3}\cdot\frac{2}{3}\,p_H = \frac{1}{3} + \frac{4}{9}\cdot p_H$$

$$\Rightarrow \quad p_H = p_T = \frac{3}{5}$$

$$p_0 = \frac{1}{2}\cdot\frac{3}{5} + \frac{1}{2}\cdot\frac{3}{5} = \frac{3}{5}$$

This result equals the result gotten by our program. The result makes sense as player 1 is certain to win if he gets two consecutive H's, but the second H is less likely to drop. For player 2 having gotten the sequence HT, it's more likely to get an H which would cancel his sequence however from there it is more likely to get a T. So, it makes sense that player is more likely to win but that not by much.

Now, we will do the same experiment, however, the probability we choose is a variable a. Here a must be different of 0 and 1.



1: HTH

2: THH

$$p_0 = \frac{1}{2}\,p_H + \frac{1}{2}\,p_T$$

$$p_H = a\cdot p_H + (1-a)\,p_{HT} \Leftrightarrow (1-a)\,p_H = (1-a)\,p_{HT} \Leftrightarrow p_H = p_{HT} \quad (1)$$

$$p_T = a\cdot p_T + (1-a)\,p_{TH} \Leftrightarrow (1-a)\,p_T = (1-a)\,p_{TH} \Leftrightarrow p_T = p_{TH} \quad (2)$$

$$p_{TH} = (1-a)\,p_{HT} + a\cdot 0 \Leftrightarrow p_{TH} = (1-a)\,p_{HT} \quad (3)$$

$$p_{HT} = a\cdot p_T + (1-a)\cdot 1 = a\cdot p_T + (1-a) \quad (4)$$

(2) in (4) :

$$p_{HT} = a\,(1-a)\,p_{HT} + (1-a)$$

$$\Leftrightarrow\ p_{HT}\,(a^2 - a + 1) = 1 - a$$

$$\Leftrightarrow\ p_{HT} = \frac{1-a}{a^2 - a + 1} = p_H$$

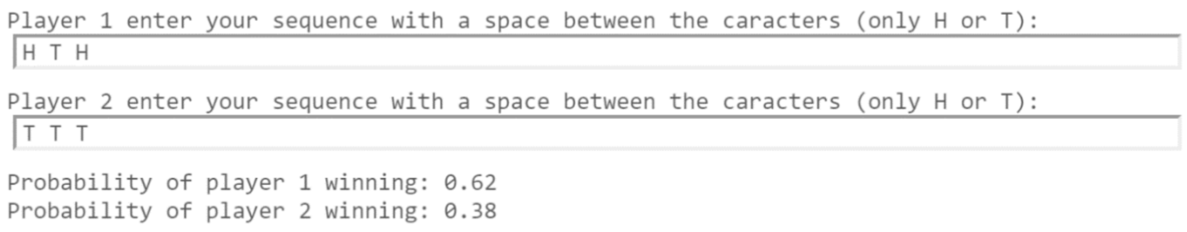
$$\Rightarrow\ p_{TH} = \frac{(1-a)^2}{a^2 - a + 1} = p_T$$

$$p_0 = \frac{1-a}{2(a^2 - a + 1)} + \frac{(1-a)^2}{2(a^2 - a + 1)} = \frac{a^2 - 3a + 2}{2(a^2 - a + 1)}$$

We checked this result for $a = \frac{1}{3}$ by comparing it to the result obtained by our program, and they are the same.

Part III: Hypothesis testing

We pick two patterns; in our example it is 'HTH' and 'TTT', and compare the theoretical results obtained by the Markov chain to the results obtained by the simulations.



```
Player 1 enter your sequence with a space between the caracters (only H or T):
H T H

Player 2 enter your sequence with a space between the caracters (only H or T):
T T T

Probability of player 1 winning: 0.62
Probability of player 2 winning: 0.38
```

According to the Markov chain theory, 'HTH' should have a 7/12 probability of winning against 'TTT'. Now we simulate the game 100 times and for every time that 'HTH' wins, the outcome is $X_i = 1$ and if 'TTT' wins the outcome is $X_i = 0$ for $i = 1, \dots, 100$. After 100 simulations 'HTH' has won 62 times while 'TTT' has won 38 times. So, 'HTH' has a 0.62 probability of winning.

Our null hypothesis H_0 is now that the data obtained by the simulation comes from a Bernoulli distribution with parameter $p = 1/2$. So, we assume that there's a 50/50 chance of winning.

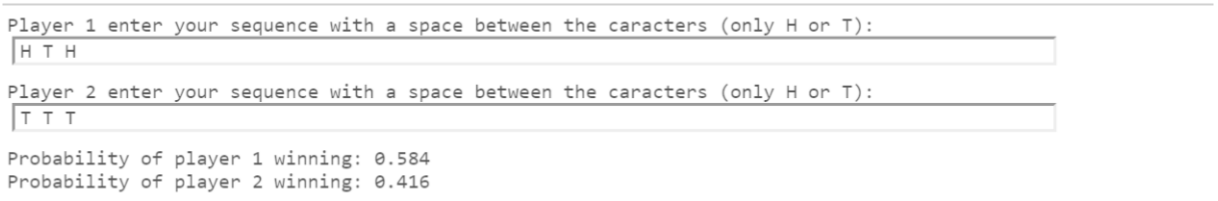
We now evaluate the quantity $Z = \frac{\sum_{i=1}^{100} X_i - 100 \cdot \frac{1}{2}}{\sqrt{100(1/2)(1-1/2)}}$.

Since 'HTH' has won 62 times, we have got $\sum_{i=1}^{100} X_i = 62$.

Therefore: $Z = \frac{62-50}{\sqrt{25}} = \frac{12}{5} = 2.4$.

If we look at the standard normal table, we can say something like $P(Z \geq 1.65) = 0.05$.

Since $Z = 2.4$, we can reject the null hypothesis H_0 at the 95% confidence level. In other words, we are 95% confident that $p > 1/2$.



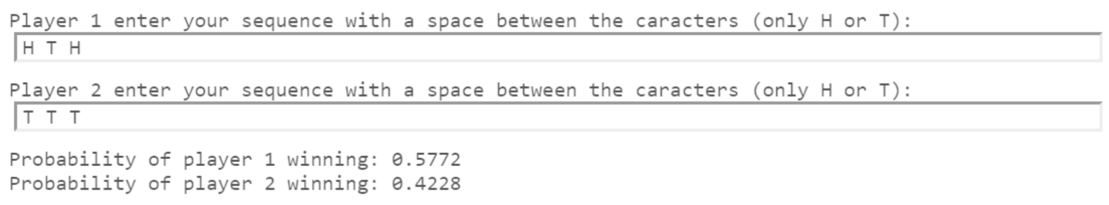
```
Player 1 enter your sequence with a space between the caracters (only H or T):
H T H

Player 2 enter your sequence with a space between the caracters (only H or T):
T T T

Probability of player 1 winning: 0.584
Probability of player 2 winning: 0.416
```

After 1000 simulations 'HTH' has won 584 times while 'TTT' has won 416 times. So, 'HTH' has a 0.584 probability of winning. We now repeat the same process. We evaluate the quantity $Z = \frac{\sum_{i=1}^{1000} X_i - 1000 \cdot \frac{1}{2}}{\sqrt{1000(1/2)(1-1/2)}}$

Since 'HTH' has won 584 times, we have got $\sum_{i=1}^{1000} X_i = 584$.

So: $Z = \frac{584-500}{\sqrt{250}} = \frac{84}{\sqrt{250}} \cong 5.31$.

Since $Z \cong 5.31$, we can reject the null hypothesis H_0 at the 95% confidence level.



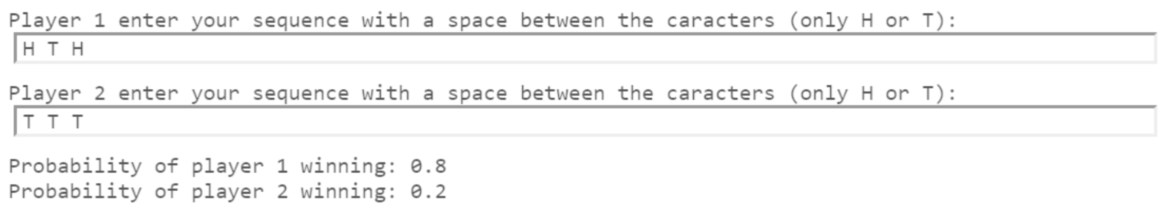
```
Player 1 enter your sequence with a space between the caracters (only H or T):
H T H
Player 2 enter your sequence with a space between the caracters (only H or T):
T T T
Probability of player 1 winning: 0.5772
Probability of player 2 winning: 0.4228
```

After 10000 simulations 'HTH' has won 5772 times while 'TTT' has won 4228 times. So, 'HTH' has a 0.5772 probability of winning. We now repeat the same process. We evaluate the quantity $Z = \frac{\sum_{i=1}^{10000} X_i - 10000 \cdot \frac{1}{2}}{\sqrt{10000(1/2)(1-1/2)}}$

Since 'HTH' has won 584 times, we've got $\sum_{i=1}^{10000} X_i = 5772$.

So: $Z = \frac{5772-5000}{\sqrt{2500}} = \frac{772}{50} = 15.44$.

Since $Z = 15.44$, we can reject the null hypothesis H_0 at the 95% confidence level.



```
Player 1 enter your sequence with a space between the caracters (only H or T):
H T H
Player 2 enter your sequence with a space between the caracters (only H or T):
T T T
Probability of player 1 winning: 0.8
Probability of player 2 winning: 0.2
```

After 10 simulations 'HTH' has won 8 times while 'TTT' has won 2 times. So, 'HTH' has a 0.8 probability of winning. We now repeat the same process. We evaluate the quantity $Z = \frac{\sum_{i=1}^{10} X_i - 10 \cdot \frac{1}{2}}{\sqrt{10(1/2)(1-1/2)}}$

Since 'HTH' has won 8 times, we've got $\sum_{i=1}^{10} X_i = 8$.

So: $Z = \frac{8-5}{\sqrt{2.5}} = \frac{3}{\sqrt{2.5}} \cong 1.90.$

Since $Z = 1.90$, we can reject the null hypothesis H_0 at the 95% confidence level.

But if we simulate the game only 10 times, the results can vary a lot, so that we can't really say anything about the probabilities or the quantity Z and the parameter p.