Build-A-Code

Peter Keep

May 5, 2017

Moraine Valley Community College

The Problem

The Problem

If I start flipping a coin, which sequence of flips is more likely to occur first: HHT or HTT?

The Problem

If I start flipping a coin, which sequence of flips is more likely to occur first: HHT or HTT?

- Heads and Tails are equally likely

The Problem

If I start flipping a coin, which sequence of flips is more likely to occur first: HHT or HTT?

- Heads and Tails are equally likely
- The probability of flipping HH and HT is equally likely

The Problem

If I start flipping a coin, which sequence of flips is more likely to occur first: HHT or HTT?

- Heads and Tails are equally likely
- The probability of flipping HH and HT is equally likely
- The wording of the question is important

Test It!

Test It!

Flip your coin, and write down the results of each flip in order. Stop when you see one of these patterns:

Pattern 1: HHT
Pattern 2: HTT

Test It!

Flip your coin, and write down the results of each flip in order. Stop when you see one of these patterns:

Pattern 1: HHT
Pattern 2: HTT

BONUS: Count the number of coin flips it takes to see either pattern.

Which pattern comes up more often?

How many flips did it take?

Build a Program

What do we need the program to do?

Flip a Coin

```
import random
population=[0,1] (0=heads, 1=tails)
flip=random.choice(population)
```

Record Your Flip

We'll set up a list to write down our coin flips (`fliplist`) .

Record Your Flip

We'll set up a list to write down our coin flips (`fliplist`) .

This will be our initial setup BEFORE we flip.

```
fliplist=[]
```

Record Your Flip

We'll set up a list to write down our coin flips (`fliplist`) .

This will be our initial setup BEFORE we flip.

```
fliplist=[]
```

This is what we'll do AFTER we flip.

```
fliplist.append(flip)
```

`fliplist=fliplist[-3:]` (just look at the last 3 flips)

Check the Patterns

Initially, we'll define our patterns (`pattern1` and `pattern2`) as well as counters for when pattern 1 or pattern 2 wins (`p1wins` and `p2wins`).

Check the Patterns

Initially, we'll define our patterns (`pattern1` and `pattern2`) as well as counters for when pattern 1 or pattern 2 wins (`p1wins` and `p2wins`).

```
pattern1=[0,0,1]
pattern2=[0,1,1]
p1wins=0
p2wins=0
```

Check the Patterns

We'll use "if-then" statements to add wins and lengths to the appropriate counters when either pattern shows up.

Check the Patterns

We'll use "if-then" statements to add wins and lengths to the appropriate counters when either pattern shows up.

```
if fliplist==pattern1:
  p1wins=p1wins+1

if fliplist==pattern2:
  p2wins=p2wins+1
```

Keep Trying Until They Match

This is the most technical part. We're going to put most of the pieces we've build inside something called a "while loop." Since we'll stop flipping whenever we see either pattern, we'll keep flipping coins whenever our `fliplist` doesn't match `pattern1` or `pattern2`.

Keep Trying Until They Match

This is the most technical part. We're going to put most of the pieces we've build inside something called a "while loop." Since we'll stop flipping whenever we see either pattern, we'll keep flipping coins whenever our `fliplist` doesn't match `pattern1` or `pattern2`.

```
while fliplist != pattern1 and fliplist!=pattern2:
```

Bonus: Count the Flips!

Bonus: Count the Flips!

Each time we flip a coin, we should add a small counter. When we find the pattern, the counter should stop. We should store that number somewhere, and then when we're done, average the numbers for each pattern.

Bonus: Count the Flips!

Each time we flip a coin, we should add a small counter. When we find the pattern, the counter should stop. We should store that number somewhere, and then when we're done, average the numbers for each pattern.

Setup: `p1length=[]`, `p2length=[]`, `i=0`

Bonus: Count the Flips!

Each time we flip a coin, we should add a small counter. When we find the pattern, the counter should stop. We should store that number somewhere, and then when we're done, average the numbers for each pattern.

Setup: `p1length=[]`, `p2length=[]`, `i=0`

After each flip: `i=i+1`

Bonus: Count the Flips!

Each time we flip a coin, we should add a small counter. When we find the pattern, the counter should stop. We should store that number somewhere, and then when we're done, average the numbers for each pattern.

Setup: `p1length=[]`, `p2length=[]`, `i=0`

After each flip: `i=i+1`

When a pattern "wins:" `p1length.append(i)` or `p2length.append(i)`

Bonus: Count the Flips!

Each time we flip a coin, we should add a small counter. When we find the pattern, the counter should stop. We should store that number somewhere, and then when we're done, average the numbers for each pattern.

Setup: `p1length=[]`, `p2length=[]`, `i=0`

After each flip: `i=i+1`

When a pattern "wins:" `p1length.append(i)` or `p2length.append(i)`

At the end: Find the average of `p1length` and `p2length`.

The Program (link)

Results

Results

The probability of Pattern 1 showing up first is: 66.67%
Average flips until we see Pattern 1 is: 5.67

The probability of Pattern 2 showing up first is: 33.33%
Average flips until we see Pattern 2 is: 4.67

Why?