# Daily Coding Problem #66

## Problem

This problem was asked by Square.

Assume you have access to a function `toss_biased()` which returns 0 or 1 with a probability that's not 50-50 (but also not 0-100 or 100-0). You do not know the bias of the coin.

Write a function to simulate an unbiased coin toss.

## Solution

Since we don't know the bias of the coin, it sounds like we need to roll the coin more than once and do some calculations to find something with a 50-50 chance of occurring. Let's draw out the probability chart for tossing our coin twice. Let's say the probability of getting heads is p, so tails is `1 - p`:

- HH: p * p
- HT: p * (1 - p)
- TH: (1 - p) * p
- TT: (1 - p) * (1 - p)

Since multiplication is commutative, we find that flipping heads and then tails has the same probability of flipping tails, then heads! Then, our strategy looks like this:

- Toss our coin twice.
- If we get heads and then tails, return heads. (It doesn't really matter which as long as the inverse one is opposite)
- If we get heads and then tails, return tails.
- Otherwise if we get the same outcome for both coins, re-toss.

```
from random import random

BIAS = 0.66

def toss_biased():
    return random() > BIAS

def toss_fair():
    t1, t2 = toss_biased(), toss_biased()
    if t1 and not t2:
        return True
    elif not t1 and t2:
        return False
    else:
        return toss_fair()
```

Testing this seems to bear it out:

```
from collections import defaultdict
c = defaultdict(int)
for i in range(1000000):
    c[toss_fair()] += 1
print(c)
```

```
defaultdict(<class 'int'>, {False: 500104, True: 499896})
```

Because there's a possibility that we always roll the same two values, there is a possibility that this function never terminates.