Daily Coding Problem #45

Problem

This problem was asked by Two Sigma.

Using a function `rand5()` that returns an integer from 1 to 5 (inclusive) with uniform probability, implement a function `rand7()` that returns an integer from 1 to 7 (inclusive).

Solution

We can solve this by computing `rand5()` twice. This gives us more than 7 options to choose from. However, we must be careful not to take the sum or product of the results -- this can skew the probability distribution. Consider that there's only one way to make 2 from two `rand5`s but two ways to make 3.

So we must consider each distinct pair of `rand5()` results. This gives us 5^2 = 25 different ways to pick from, each uniformly distributed. Ideally, we would divide these by 7, but no power of 5 is also a multiple of 7 (consider the prime factorization of 5^N), so we will have to make do. For our solution, we'll make a table of results:

	1	**2**	**3**	**4**	**5**
1	1	1	1	6	7
2	2	2	2	6	7
3	3	3	3	6	7
4	4	4	4	R	R
5	5	5	5	R	R

R means we need to reroll.

```
def rand7():
    r1, r2 = rand5(), rand5()
```

```
    if r2 <= 3:
        return r1
    elif r2 == 4:
        if r1 <= 3:
            return 6
        else:
            return rand7()
    else: # r2 == 5
        if r1 <= 3:
            return 7
        else:
            return rand7()
```

This method has a potentially infinite runtime, since it's possible that we always roll the cases where we need to reroll.