Problem Statement

You have several cards, each with a number printed on them, given in int[] cards. You select two cards--with replacement--from the deck. You then concatenate the values together to form a single value. For example, if you first selected 5 and then 13, your concatenated value would be 513.

What is the probability that the value is prime?

Definition

Class: TwoCardsDraw

Method: primeChance

Parameters: int[]

Returns: double

Method signature: double primeChance(int[] cards)

(be sure your method is public)

Notes

- No leading zeroes are added when concatenating the numbers.

Constraints

- cards will contain between 1 and 50 elements, inclusive.
- Each element of cards will be between 1 and 99, inclusive.

Examples

0)

{ 1 }

Returns: 1.0

The only number you're going to get after concatenation is 11, so it's definitely prime.

1)

{ 1, 3 }

Returns: 0.75

11, 13, and 31 are prime, but 33 is not.

2)

{ 2, 4 }

Returns: 0.0

No number you get will be prime.

3)

{ 1, 1, 3 }

Returns: 0.8888888888888888

Notice that numbers can be repeated.

This problem was used for:

Rookie SRM 17 Rookie 17 - Division I, Level Two

```
def isPrime(n):
    for i in range(2,int(math.sqrt(n))+1):
        if n%i==0:
            return False
    return True


class TwoCardsDraw:
    def primeChance(self, cards):
        cnt=0
        cards=list(cards)
        for e1 in cards:
            for e2 in cards:
                strR=str(e1)+str(e2)
                #print(int(strR))
                if(isPrime(int(strR))):
                    cnt+=1


        return float(cnt)/float(len(cards))/float(len(cards))


obj = TwoCardsDraw()


t1=tuple([50,99,50,50,50,50,50,50,50,50,50,99,50,50,50,50,50,50,50,50,50,50,99,50,99,50,50,50,5
0,50,50,50,50,50,50,99,33])
x=obj.primeChance(t1)
print(x)
```