Homework 2

Due Monday, October 14, 2019

Math 206

1. Type in the function `isPrime` that we saw in lecture, and improve it if you want. Test it on a few examples.

2. Apart from 2 and 5, every prime has either a 1, 3, 7, or 9 in the last digit of its base-10 representation. We may wonder whether these four possibilities equally distributed, or whether one occur more frequently than another. Let's investigate.

   First, what does the following code do?

```
count3 = 0
for i in range(1,100):
  if isPrime(i) and i%10 == 3:
    count3 += 1
print(count3)
```

   Next, edit the code above and use it to fill in the following table:

last digit:	1	3	7	9
# of primes < 100 :				
# of primes < 1000 :				
# of primes < 10000 :				

   Challenge question: What about other bases than base 10?

(Continued on next page.)

3. An integer is called *square-free* if it is not divisible by the square of any number (other than 1). For example, $10 = 2 \cdot 5$ is square-free, although it is not prime; but $20 = 2^2 \cdot 5$ is not even square-free.

   Write a function `isSquareFree`, modeled on `isPrime`, that tests whether a number is or is not square-free, and outputs `True` or `False` as appropriate. Test it on a few examples.

4. Determine the number of square-free numbers that are less than 1000. Then do the same for 10,000 and 100,000. If you did things correctly, you should start to see a pattern. In each case, what is the approximate percentage of numbers that are square-free?

   The fraction of numbers betwee 1 and N that are square-free is known to converge to $6/\pi^2$ as $N \to \infty$. Does this agree with your findings?

5. Challenge question: This question question concerns "gaps" between prime. After 7, the next prime is 11, so that's a gap of 4. After 89, the next prime is 97, so that's a gap of 8. Find the largest gap between primes in the following cases:

   (a) Primes less than 500.

   (b) Primes less than 1,000.

   (c) Primes less than 10,000.

   (d) Primes less than 100,000.

   To solve this problem in a reasonable amount of time, you may need to write a smarter prime-finding algorithm than the one we've been using.