What is a prime number?

A prime number is a number that is only evenly divisible by itself and 1.

How do we determine if a number is "even divisible" by another number in Python?

"n" is evenly divisible by "i" if the following expression (using the modulus operator) is True:

```
n%i == 0
```

This says:

The integer remainder of "n" divided by "i" equals 0

Remember that "range(2, n)" returns numbers between 2 and n-1 (inclusive), which excludes the numbers 1 and n themselves!

Python:

```
def isprime(n):
  for i in range(2, n):
    if n%i == 0:
      return False
  return True
```

To list prime numbers less than 1000...

```
def listprimes(max=1000):
  for i in range(1, max):
    if isprime(i):
      print(i, end=" ")
  print("")
```

```
>>> listprimes()
1 2 3 5 7 11 13 17 19 23 29 31 37 41 43 47 53 59 61 67 71 73 79 83 89 97 101 103 107 109
113 127 131 137 139 149 151 157 163 167 173 179 181 191 193 197 199 211 223 227 229 233
239 241 251 257 263 269 271 277 281 283 293 307 311 313 317 331 337 347 349 353 359 367
373 379 383 389 397 401 409 419 421 431 433 439 443 449 457 461 463 467 479 487 491 499
503 509 521 523 541 547 557 563 569 571 577 587 593 599 601 607 613 617 619 631 641 643
647 653 659 661 673 677 683 691 701 709 719 727 733 739 743 751 757 761 769 773 787 797
809 811 821 823 827 829 839 853 857 859 863 877 881 883 887 907 911 919 929 937 941 947
953 967 971 977 983 991 997
```

Or to list the first hundred prime numbers (instead of all prime numbers less than a thousand):

```
def listprimes(count=100):
  i = 1
  while True:
    if count <= 0:
      print("")
      return
    if isprime(i):
      print(i, end=" ")
      count -= 1
    i += 1
```

Which shows the first hundred prime numbers end at 523:

```
>>> listprimes()
1 2 3 5 7 11 13 17 19 23 29 31 37 41 43 47 53 59 61 67 71 73 79 83 89 97 101 103 107 109
113 127 131 137 139 149 151 157 163 167 173 179 181 191 193 197 199 211 223 227 229 233
239 241 251 257 263 269 271 277 281 283 293 307 311 313 317 331 337 347 349 353 359 367
373 379 383 389 397 401 409 419 421 431 433 439 443 449 457 461 463 467 479 487 491 499
503 509 521 523
```

To print the distance between prime numbers, we can remember the previous prime and subtract that from the next prime:

```
def listprimes(count=100):
  last = 0
  i = 1
  while True:
    if count <= 0:
      print("")
      return
    if isprime(i):
      print(i, i-last)  # print the prime and the distance from the previous prime
      last = i
      count -= 1
    i += 1
```

For the first hundred primes, on average there is a prime about every 5 numbers (523/100=5.23):

```
>>> listprimes()
1 1
2 1
3 1
5 2
...
503 4
509 6
521 12
523 2
```

What about up to the thousandth prime? On average, there is a prime about every 8 numbers (7907/1000=7.907)

```
>>> listprimes(1000)
...
7879 2
7883 4
7901 18
7907 6
```

Or up to the ten thousandth? On average, there is a prime about every 10 numbers (104723/10000=10.4723)

```
>>> listprimes(10000)
...
104707 6
104711 4
104717 6
104723 6
```

We can go (much!) faster by replacing:

```
  for i in range(2, n):
```

With:

```
  import math
  for i in range(2, int(math.sqrt(n))+1):
```

Do you see why? The square root of 10 is 3.16. As the divisor goes up, the quotient goes down!

```
10/2 = 5
10/3 = 3.33
10/4 = 2.5
10/5 = 2
10/6 = 1.67
10/7 = 1.43
```