

Question 1.

How many seconds are in an hour? Use the interactive interpreter as a calculator and multiply the number of seconds in a minute (60) by the number of minutes in an hour (also 60).

Answer 1:

```
60*60
```

```
Output: 3600
```

Question 2.

Assign the result from the previous task (seconds in an hour) to a variable called `seconds_per_hour`.

Answer 2:

```
seconds_per_hour = 3600
```

Question 3.

How many seconds do you think there are in a day? Make use of the variables `seconds per hour` and `minutes per hour`.

Answer 3:

```
seconds_per_hour*24
```

```
output: 86400
```

Question 4.

Calculate seconds per day again, but this time save the result in a variable called `seconds_per_day`

Answer 4:

```
seconds_per_day = seconds_per_hour*24
```

```
seconds_per_day
```

```
Output: 86400
```

Question 5.

Divide `seconds_per_day` by `seconds_per_hour`. Use floating-point (/) division.

Answer 5:

```
seconds_per_day / seconds_per_hour
```

```
output: 24.0
```

Question 6.

Divide `seconds_per_day` by `seconds_per_hour`, using integer (//) division. Did this number agree with the floating-point value from the previous question, aside from the final .0?

Answer 6:

```
seconds_per_day // seconds_per_hour
```

```
output: 24
```

Question 7.

Write a generator, genPrimes, that returns the sequence of prime numbers on successive calls to its next() method: 2, 3, 5, 7, 11,

Answer 7:

```
def genPrimes():  
  
    primes = [ 2, 3, 5, 7, 11 ]  
  
    def isPrimeNumber(n):  
        if n in primes:  
            return True  
  
        for elem in primes:  
            if n % elem == 0:  
                return False  
  
        primes.append(n)  
        return True  
    num = 1  
    while True:  
        num += 1  
        if isPrimeNumber(num):  
            next = num  
            yield next  
            num = next  
primeNumber = genPrimes()  
  
for i in range(189):  
    print(primeNumber.__next__())
```