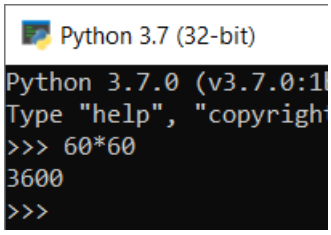
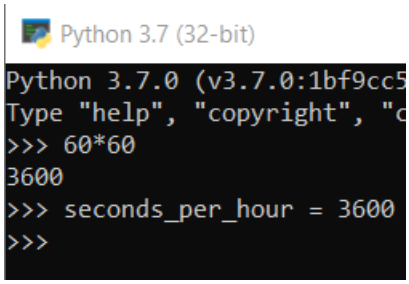
1.How many seconds are in an hour? Use the interactive interpreter as a calculator and multiply the number of seconds in a minute (60) by the number of minutes in an hour (also 60).

3600 seconds

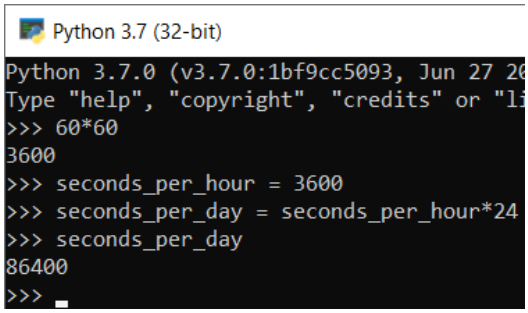


2. Assign the result from the previous task (seconds in an hour) to a variable called seconds_per_hour.

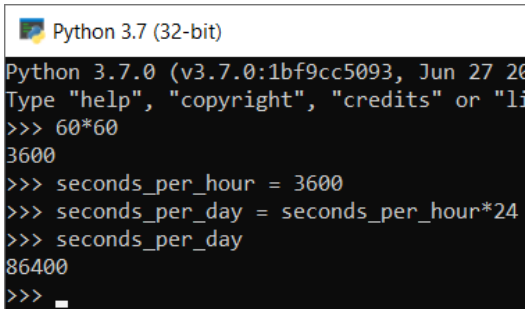


3. How many seconds do you think there are in a day? Make use of the variables seconds per hour and minutes per hour.

Ans: 86400 seconds



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



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

```
>>> seconds_per_day/seconds_per_hour
24.0
>>>
```

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?

```
>>> seconds_per_day/seconds_per_hour
24.0
>>> seconds_per_day//seconds_per_hour
24
>>>
```

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

Code:

```
#Generator to generate Prime Numbers

def genprimes():
    num = 0
    while num < 100:
        factors = []
        num+=1
        for i in (range(1,int(num/2+1))):
            if num%i == 0:
                factors.append(i)
        if len(factors) == 1:
            yield num #yielding prime numbers on calling next()

prime = genprimes()

print(next(prime))
print(next(prime))
print(next(prime))
print(next(prime))
print(next(prime))
```

```python
#Generator to generate Prime Numbers

def genprimes():
    num = 0
    while num < 100:
        factors = []
        num+=1
        for i in (range(1,int(num/2+1))):
            if num%i == 0:
                factors.append(i)
        if len(factors) == 1:
            yield num #yielding prime numbers on calling next()

prime = genprimes()

print(next(prime))
print(next(prime))
print(next(prime))
print(next(prime))
print(next(prime))
```

```
2
3
5
7
11
```