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).

sol.

```
>>> 60*60
```

3600

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

Sol. seconds_per_hour = 60*60

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

Sol.

```
>>> seconds_per_hour = 60*60
>>> 24*seconds_per_hour
```

86400

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

Sol. seconds_per_day = 24*seconds_per_hour

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

Sol.

```
>>> seconds_per_day  = 24*seconds_per_hour
>>> 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?

Sol.

```
>>> 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, ...

Sol.

```
[ ]:

[72]: def genPrimes(num :int):
          for i in range(2,num):
              isPrime = True
              for j in range(2,i//2+1):
                  if(i%j == 0.0):
                      isPrime = False
                      break
              if(isPrime):
                  yield i

[85]: p = genPrimes(100)

[86]: next(p)

[86]: 2

[87]: next(p)

[87]: 3

[88]: for i in iter(p):
          print(i, end = ', ')

      5, 7, 11, 13, 17, 19, 23, 29, 31, 37, 41, 43, 47, 53, 59, 61, 67, 71, 73, 79, 83, 89, 97,

[ ]:
```