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. To calculate the number of seconds in an hour, you can multiply the number of seconds in a minute (60) by the number of minutes in an hour (60). Using the interactive interpreter, you can enter the following calculation:

60 \* 60

The result will be:

3600

Therefore, there are 3600 seconds in an hour.

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

Sol: To assign the result from the previous calculation (3600 seconds) to a variable called `seconds\_per\_hour`, you can use the following code:

seconds\_per\_hour = 3600

Now the variable `seconds\_per\_hour` holds the value 3600, which represents the number of seconds in an 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.

Sol: To calculate the number of seconds in a day, we need to multiply the number of seconds per hour by the number of hours in a day. Since there are 24 hours in a day, we can use the variables `seconds\_per\_hour` (which is already assigned as 3600) and `hours\_per\_day` (which is 24) to perform the calculation.

seconds\_per\_day = seconds\_per\_hour \* 24

Let's calculate it:

seconds\_per\_day = 3600 \* 24

The result will be:

86400

Therefore, there are 86,400 seconds in a day.

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

Sol: Let's calculate the number of seconds in a day and save the result in a variable called `seconds\_per\_day`. Since the value of `seconds\_per\_hour` is already assigned as 3600, we can use that along with the variable `hours\_per\_day` (which is 24) to calculate `seconds\_per\_day`. Here's the code:

hours\_per\_day = 24

seconds\_per\_day = seconds\_per\_hour \* hours\_per\_day

Now let's calculate it:

seconds\_per\_day = 3600 \* 24

The result will be:

86400

Therefore, there are 86,400 seconds in a day, and the value is stored in the variable `seconds\_per\_day`.

5. Divide seconds\_per\_day by seconds\_per\_hour. Use floating-point (/) division.

Sol: To divide `seconds\_per\_day` by `seconds\_per\_hour` using floating-point division, you can use the following code:

seconds\_per\_day = 86400

seconds\_per\_hour = 3600

division\_result = seconds\_per\_day / seconds\_per\_hour

Let's calculate it:

division\_result = 86400 / 3600

The result will be:

24.0

Therefore, `seconds\_per\_day` divided by `seconds\_per\_hour` using floating-point division is equal to 24.0. The value is stored in the variable `division\_result`.

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: Let's calculate the division of `seconds\_per\_day` by `seconds\_per\_hour` using integer division (`//`) and check if the result agrees with the floating-point value from the previous question. Here's the code:

seconds\_per\_day = 86400

seconds\_per\_hour = 3600

division\_result = seconds\_per\_day // seconds\_per\_hour

Let's calculate it:

division\_result = 86400 // 3600

The result will be:

24

Therefore, when using integer division (`//`), the result is 24 without the decimal point. This value agrees with the floating-point division result of 24.0 from the previous question, aside from the final `.0`.

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: implementation of the `genPrimes` generator that returns the sequence of prime numbers on successive calls to its `next()` method:

def genPrimes():

prime\_list = [2]

yield 2

num = 3

while True:

is\_prime = True

for prime in prime\_list:

if num % prime == 0:

is\_prime = False

break

if is\_prime:

prime\_list.append(num)

yield num

num += 2 # Increment by 2 to check only odd numbers

primes = genPrimes()

print(next(primes)) # Output: 2

print(next(primes)) # Output: 3

print(next(primes)) # Output: 5

print(next(primes)) # Output: 7

print(next(primes)) # Output: 11