**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).**

seconds\_per\_hour = 60 \* 60

print(seconds\_per\_hour) # Output: 3600

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

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.**

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

print(seconds\_per\_day) # Output: 86400

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

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

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

result = seconds\_per\_day / seconds\_per\_hour

print(result) # Output: 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?**

result = seconds\_per\_day // seconds\_per\_hour

print(result) # Output: 24

Yes, the integer division result (24) agrees with the floating-point result (24.0), aside from the .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, ...**

def genPrimes():

primes = [] # List to store prime numbers

num = 2 # Start checking from 2

while True:

is\_prime = True

for prime in primes:

if num % prime == 0:

is\_prime = False

break

if is\_prime:

primes.append(num)

yield num

num += 1

# Example usage:

prime\_gen = genPrimes()

print(next(prime\_gen)) # Output: 2

print(next(prime\_gen)) # Output: 3

print(next(prime\_gen)) # Output: 5

print(next(prime\_gen)) # Output: 7

print(next(prime\_gen)) # Output: 11