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).

Ans: seconds\_in\_minute = 60

minutes\_in\_hour = 60

seconds\_in\_hour = seconds\_in\_minute \* minutes\_in\_hour

print(seconds\_in\_hour)

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

Ans: seconds\_in\_minute = 60

minutes\_in\_hour = 60

seconds\_per\_hour = seconds\_in\_minute \* minutes\_in\_hour

print(seconds\_per\_hour)

After executing this code, the value `3600` will be assigned to the variable `seconds\_per\_hour`.

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

Ans: There are 86400 seconds in a day.

seconds\_per\_hour = 60 \* 60

minutes\_in\_hour = 60

hours\_in\_day = 24

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

print(seconds\_in\_day)

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

Ans: seconds\_per\_hour = 60 \* 60

minutes\_in\_hour = 60

hours\_in\_day = 24

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

print(seconds\_per\_day)

After excuting this code the value 86400 will be assigned to the variable ‘second\_per\_day’.

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

Ans: seconds\_per\_hour = 60 \* 60

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

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

print(result)

after run this code, the output is

24.0

1. 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?

Ans: seconds\_per\_hour = 60 \* 60

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

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

print(result)

The result of the integer division is `24`, which is the same as the floating-point division result from the previous question, disregarding the final `.0`. The integer division discards any decimal places and returns the whole number portion of the division result. In this case, both the floating-point and integer division results indicate that there are 24 hours in a day.

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

Ans: def genPrimes():

# Start with the first prime number

yield 2

primes = [2] # Store the prime numbers found so far

num = 3 # Start checking from the next number after 2

while True:

is\_prime = True

# Check if the current number is divisible by any previously found prime numbers

for prime in primes:

if num % prime == 0:

is\_prime = False

break

if is\_prime:

# If the current number is prime, yield it and add it to the list of primes

yield num

primes.append(num)

num += 2 # Only check odd numbers for primality

# Test the genPrimes generator

primes\_generator = genPrimes()

# Print the first few prime numbers

for \_ in range(10):

print(next(primes\_generator))

Output:

2

3

5

7

11

13

17

19

23

29