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).

Ans1

seconds\_in\_minute = 60

minutes\_in\_hour = 60

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

print(seconds\_in\_hour)

output 3600

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

Ans2

seconds\_per\_hour = 60 \* 60

print(seconds\_per\_hour)

output 3600

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

Ans3

seconds\_per\_hour = 60 \* 60

minutes\_per\_hour = 60

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

Ans4

seconds\_per\_hour = 60 \* 60

minutes\_per\_hour = 60

hours\_per\_day = 24

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

print(seconds\_per\_day)

output 86400

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

Ans5

seconds\_per\_hour = 60 \* 60

minutes\_per\_hour = 60

hours\_per\_day = 24

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

seconds\_per\_hour\_in\_day = seconds\_per\_day / seconds\_per\_hour

print(seconds\_per\_hour\_in\_day)

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?

Ans6

seconds\_per\_hour = 60 \* 60

minutes\_per\_hour = 60

hours\_per\_day = 24

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

seconds\_per\_hour\_in\_day = seconds\_per\_day // seconds\_per\_hour

print(seconds\_per\_hour\_in\_day)

output 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, ...

Ans7

def genPrimes():

# First prime number

yield 2

primes = [2] # List of prime numbers found so far

num = 3 # Next number to test for primality

while True:

# Check if num is prime

is\_prime = True

for p in primes:

if num % p == 0:

is\_prime = False

break

if is\_prime:

# num is prime, so yield it and add it to the list of primes

yield num

primes.append(num)

num += 2 # Move on to the next odd number

gen = genPrimes()

print(next(gen)) # Output: 2

print(next(gen)) # Output: 3

print(next(gen)) # Output: 5

print(next(gen)) # Output: 7

print(next(gen)) # Output: 11