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).

:- 3600.

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

:- seconds\_per\_hour = 3600

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\_hour \* 24\* 7

604800

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

86400

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

:- 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?

:- seconds\_per\_day // seconds\_per\_hour

24

integer division always rounds down to the nearest whole number, so the result of the division is an integer rather than a floating-point number.

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 = []

num = 2

while True:

is\_prime = True

for p in primes:

if num % p == 0:

is\_prime = False

break

if is\_prime:

primes.append(num)

yield num

num += 1

primes = genPrimes()

print(next(primes)) # prints 2

print(next(primes)) # prints 3

print(next(primes)) # prints 5

print(next(primes)) # prints 7

print(next(primes)) # prints 11