**ANSWERS**

**Q1.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. 60**

**Ans.** 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):

seconds\_in\_an\_hour = 60 \* 60

print(seconds\_in\_an\_hour)

The output will be: 3600

**Q2. Assign the result from the previous task (seconds in an hour) to a variable called seconds\_per\_hour.**

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

seconds\_per\_hour = 3600

**Q3. How many seconds do you think there are in a day? Make use of the variables seconds per hour and minutes per hour.**

**Ans.** To calculate the number of seconds in a day, you can multiply the seconds\_per\_hour variable by the number of hours in a day (24):

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

print(seconds\_in\_a\_day)

The output will be: 86400

**Q4. Calculate seconds per day again, but this time save the result in a variable called seconds\_per\_day.**

**Ans.** 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

print(seconds\_per\_day)

The output will be: 86400

**Q5. Divide seconds\_per\_day by seconds\_per\_hour. Use floating-point (/) division.**

**Ans.** To calculate the ratio of seconds\_per\_day to seconds\_per\_hour using floating-point division, you can divide seconds\_per\_day by seconds\_per\_hour:

ratio = seconds\_per\_day / seconds\_per\_hour

print(ratio)

The output will be: 24.0

**Q6. 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.** To calculate the ratio of seconds\_per\_day to seconds\_per\_hour using integer division, you can use the // operator:

ratio = seconds\_per\_day // seconds\_per\_hour

print(ratio)

The output will be: 24. Yes, the integer division result agrees with the floating-point value from the previous question, aside from the final .0.

**Q7. Write a generator, genPrimes, that returns the sequence of prime numbers on successive calls to its next() method: 2, 3, 5, 7, 11, ...**

**Ans.** Here's an example implementation of the genPrimes generator, which returns the sequence of prime numbers on successive calls to its next() method:

def genPrimes():

primes = []

num = 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\_generator = genPrimes()

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

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

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

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

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

# ... and so on

The genPrimes generator generates prime numbers on each successive call to its next() method. It uses a list to store previously generated primes and checks if the current number is divisible by any of the primes to determine if it is prime.