**ADITYA AMIN** **ASSIGN : 15**

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).

sol. 60

>>> seconds\_in\_minute = 60

>>> minutes\_in\_hour = 60

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

>>> seconds\_in\_hour

3600

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

>>> seconds\_in\_minute = 60

>>> minutes\_in\_hour = 60

>>> seconds\_per\_hour = seconds\_in\_minute \* minutes\_in\_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.

>>> hours\_in\_day = 24

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

>>> seconds\_in\_day

86400

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

>>> seconds\_per\_hour = 60 \* 60

>>> minutes\_in\_hour = 60

>>> hours\_in\_day = 24

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

>>> seconds\_per\_day

86400

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

>>> seconds\_per\_day = 86400

>>> seconds\_per\_hour = 3600

>>> division\_result = seconds\_per\_day / seconds\_per\_hour

>>> division\_result

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 = 86400

>>> seconds\_per\_hour = 3600

>>> integer\_division\_result = seconds\_per\_day // seconds\_per\_hour

>>> integer\_division\_result

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, ...

def genPrimes():

"""Generator that yields prime numbers."""

primes = [] # List to store prime numbers

num = 2 # Start with the first prime number, 2

while True:

is\_prime = True

for prime in primes:

# Check if the current number is divisible by any prime number

if num % prime == 0:

is\_prime = False

break

if is\_prime:

primes.append(num)

yield num # Yield the prime number

num += 1 # Move on to the next number

# Create an instance of the genPrimes generator

prime\_generator = genPrimes()

# Generate the first 5 prime numbers

for \_ in range(5):

print(next(prime\_generator))