**Assignment 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).**

**Ans 1-** 60**\***60

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

**seconds\_per\_hour.**

**Ans 2-** 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.**

**Ans 3-** seconds\_per\_hour**\***24

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

**Ans 4-** seconds\_per\_day **=** seconds\_per\_hour**\***24

seconds\_per\_day

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

**Ans 5-** seconds\_per\_day **/** seconds\_per\_hour

**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?**

**Ans 6-** seconds\_per\_day **//** seconds\_per\_hour

**7. Write a generator, genPrimes, that returns the sequence of prime numbers on successive calls to**

**its next() method: 2, 3, 5, 7, 11, ...**

**Ans 1-**

def genPrimes():

primes = [ 2, 3, 5, 7, 11 ]

def isPrimeNumber(n):

if n in primes:

return True

for elem in primes:

if n % elem == 0:

return False

primes.append(n)

return True

num = 1

while True:

num += 1

if isPrimeNumber(num):

next = num

yield next

num = next

primeNumber = genPrimes()

for i in range(189):

print(primeNumber.\_\_next\_\_())