1)Which keyword is used to create a function? Create a function to return a list of add two numbers in the range of 1 to 25.

Ans- def keyword

start, end = 1, 25

for num in range(start, end + 1):

# checking condition

if num % 2 != 0:

print(num, end = " ")

2)Why \*args and \*\*kwargs is used in some functions? Create a function each for \*args and \*\*kwargs to demonstrate their use.

Ans- When we are unsure about the number of arguments to pass in the functions.

def myFun(\*argv):

for arg in argv:

print(arg)

def myFun(\*\*kwargs):

for key, value in kwargs.items():

print("%s == %s" % (key, value))

# Driver code

myFun(first='Sayantan', mid='Learning', last='Data Science')

3)What is an iterator in python? Name the method used to initialize the iterator object and the method used for iteration.

Ans- An iterator is an object that contains a countable number of values.

An iterator is an object that can be iterated upon, meaning that you can traverse through all the values.

The iterator object is initialized using the iter() method. It uses the next() method for iteration

4)What is agenerator function in python? Why yield keyword is used? Give an example of a generator function.

Ans- In Python, a generator is a function that returns an iterator that produces a sequence of values when iterated over. Generators are useful when we want to produce a large sequence of values, but we don't want to store all of them in memory at once.

Yield keyword is used to create a generator function. A type of function that is memory efficient and can be used like an iterator object.

def my\_generator(n):

# initialize counter

value = 0

# loop until counter is less than n

while value < n:

# produce the current value of the counter

yield value

# increment the counter

value += 1

# iterate over the generator object produced by my\_generator

for value in my\_generator(3):

# print each value produced by generator

print(value)

5)Create a generator function for prime numbers less than 1000. Use the next() method to print the first 20 prime numbers.

Ans- lower\_value = int(input ("Please, Enter the Lowest Range Value: "))

upper\_value = int(input ("Please, Enter the Upper Range Value: "))

def my\_generator(n):

print ("The Prime Numbers in the range are: ")

for number in range (lower\_value, upper\_value + 1):

if number > 1:

for i in range (2, number):

if (number % i) == 0:

break

else:

print (number)