EX.No:1	Generation of prime numbers and computation of GCD for foundational cryptographic operations.
DATE:	

AIM:

To write a Python program to

a) Generate prime numbers within a given range.

b) Compute the Greatest Common Divisor (GCD) for multiple numbers

ALGORITHM:

Generating prime numbers:

- **Step 1:** Start.
- **Step 2:** Define a function prime(n) to generate prime numbers up to n.
- **Step 3:** Inside the function, use a for loop from 1 to n (inclusive).
- **Step 4:** For each number in the loop:
  - Check if the number is greater than 1.
  - If yes, use another loop from 2 to the number (exclusive).
  - Inside the inner loop, check if the number is divisible by any i:
    - If number % i == 0, the number is not prime; use break.
  - If the inner loop completes without breaking, print the number as prime.
- **Step 5:** Accept user input for n (upper limit).
- **Step 6:** Call the function prime(n) to display all prime numbers up to n.
- **Step 7:** End.

Greatest Common Divisor (GCD):

- **Step 1:** Start.
- **Step 2**: Define a function gcd_list(nums) to compute the GCD of a list of numbers.
- **Step 3:** Inside the function, find the smallest number in the list using min(nums) and store it in min_num.
- **Step 4:** Use a for loop to iterate from min_num down to 1 (i.e., in reverse order).
- **Step 5:** For each number i in the loop:
  - Use all() with a generator expression to check if every number n in the list satisfies n % i == 0.

- If true, this means i divides all numbers in the list.
- Return i as the GCD.

➢ **Step 6:** Outside the function, accept a list of integers from the user using:

PROGRAM:1 Checking the given number is prime or not

```
def is_prime(n):
    if n <= 1:
        return False
    for i in range(2, n):
        if n % i == 0:
            return False
    return True
number = int(input("Enter a number: "))
if is_prime(number):
    print(f"{number} is a prime number.")
else:
    print(f"{number} is not a prime number.")
```

PROGRAM:2 Generating prime numbers

```
def is_prime(n):
    if n <= 1:
        return False
    for i in range(2, n):
        if n % i == 0:
            return False
    return True
def generate_primes(start, end):
    primes = []
    for n in range(start, end + 1):
        if is_prime(n):
            primes.append(n)
```

```
    return primes
start = int(input("Enter the starting number of the range: "))
end = int(input("Enter the ending number of the range: "))
prime_numbers = generate_primes(start, end)
print(f"Prime numbers between {start} and {end} are: {prime_numbers}")
```

PROGRAM:3 GCD using Euclidean Algorithm

```
# Input two integers
a = int(input("Enter first number: "))
b = int(input("Enter second number: "))
# Ensure both are positive (optional, Euclidean algorithm works for non-negative
integers)
if a < 0:
    a = -a
if b < 0:
    b = -b
# Euclidean Algorithm using only loop and arithmetic
while b != 0:
    r = a % b
    a = b
    b = r
# Print the GCD
print("The GCD is", a)
```

PROGRAM:4 GCD using user built-in function

```
import math
# Input two numbers
a = int(input("Enter first number: "))
b = int(input("Enter second number: "))
# Using built-in gcd() function
gcd = math.gcd(a, b)
```

```
# Output the result
print("The GCD is", gcd)
```

PROGRAM:5 Computing the GCD for n numbers

```
def gcd_list(nums):
    min_num = min(nums)
    for i in range(min_num, 0, -1):
        if all(n % i == 0 for n in nums):
            return i
nums = list(map(int, input("Enter numbers separated by space: ").split()))
print("GCD is:", gcd_list(nums))
```

Result:

Hence the python code to implement first n prime numbers using function is obtained and also GCD of the multiple numbers is obtained by various techniques through the above programs.

EXERCISE:

1. Find the count of prime numbers between two given numbers.
2. Generate all prime numbers whose digits are also prime.
3. Write a Python program to find the GCD of 123456 and 7890 using Euclidean Algorithm.
4. Write a Python program to find prime numbers less than 100 such that the number is prime and the reverse of the number is also a prime
5. Write a Python program to check whether 35 and 64 are coprime.