KIET GROUP OF INSTITUTIONS

ARTIFICIAL INTELLIGENCE

MID SEMESTER EXAMINATION-1

PROBLEM STATEMENT:

PRIME NUMBER GENERATOR AND CHECKER

SUBMITTED BY:

NAME- DIVYA PAL

BRANCH- CSE (AI)

SECTION- B

ROLL NO- 29

UNIVERSITY ROL NO. - 202401100300103

INTRODUCTION

A **prime number** is a natural number greater than **1** that has exactly **two factors**: **1 and itself**. This means that a prime number cannot be divided evenly by any other number except **1** and itself.

PROBLEM STAEMENT EXPLANATION:

The problem consists of two tasks: **(1) Checking if a number is prime** and **(2) Generating a list of prime numbers up to a given limit**.

- **Prime Check**: A prime number is greater than 1 and has only two factors (1 and itself). The function is_prime(n) checks if n is prime by testing divisibility from 2 to √n. If any number divides n evenly, it's not prime.
- **Prime Generation**: The function generate_primes(limit) finds all prime numbers up to limit. It loops through numbers from 2 to limit, uses is_prime() to check if each number is prime, and adds it to a list.

Methodology to Solve the Prime Number Problem

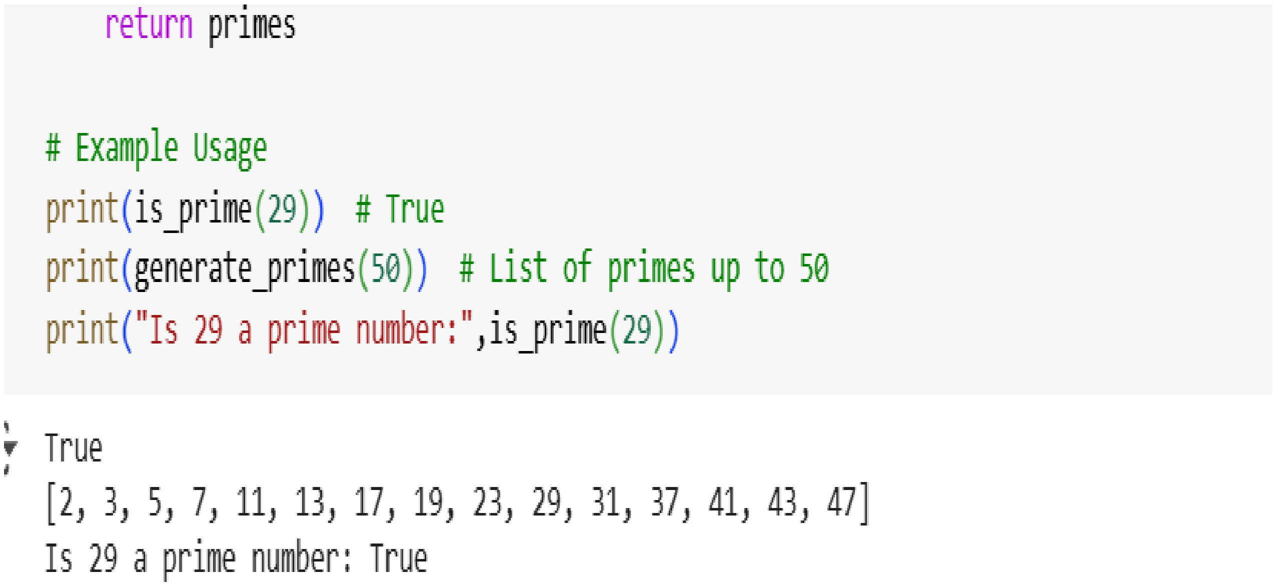
To efficiently determine prime numbers and generate a list of primes up to a given limit, we follow these steps:

1. **Understanding the Properties of Prime Numbers**
   - A prime number has exactly **two factors**: 1 and itself.
   - The number **2** is the smallest and only even prime.
2. **Approach for Prime Checking (is_prime(n))**
   - If n < 2, return **False** (0 and 1 are not prime).
   - Check divisibility from 2 to √n (since factors repeat after √n).
   - If any number divides n evenly, return **False**; otherwise, return **True**.
3. **Approach for Prime Generation (generate_primes(limit))**
   - Initialize an empty list to store prime numbers.
   - Loop through numbers from **2 to limit**.
   - Use is_prime(n) to check if each number is prime.
   - If prime, add it to the list.

CODE TO THE GIVEN PROBLEM:

```
import math
def is_prime(n):
    if n < 2:
        return False
    for i in range(2, int(math.sqrt(n)) + 1):
        if n % i == 0:
            return False
    return True
def generate_primes(limit):
    primes = []
    for num in range(2, limit + 1):
        if is_prime(num):
            primes.append(num)
    return primes
# Example Usage
print(is_prime(29))  # True
print(generate_primes(50))  # List of primes up to 50
print("Is 29 a prime number:",is_prime(29))
```

OUTPUT:



```
    return primes


# Example Usage
print(is_prime(29))  # True
print(generate_primes(50))  # List of primes up to 50
print("Is 29 a prime number:",is_prime(29))
```

```
True
[2, 3, 5, 7, 11, 13, 17, 19, 23, 29, 31, 37, 41, 43, 47]
Is 29 a prime number: True
```