Number Theory

Team Members

- Abdel-Rahman Amgad Hassan (ID: 22010871)
- Mahmoud Hesham Mohamed (ID: 22011201)

1. Problem Statement

The goal of this project is to implement a system for solving numbers theory expressions. The application includes functionalities such as prime checking, prime factorization, GCD , and LCM calculation and algorithms such as the Sieve of Eratosthenes and the Euclidean algorithm.

2. Code Logic

Prime Number Checker Class

This class is responsible for identifying prime numbers:

- **Sieve of Eratosthenes:** Takes an input N and generates a list of all prime numbers up to N. Uses a boolean array to mark non-prime numbers.
- **isPrime:** Checks if a number n is prime by checking if it exists in the list returned by using the Sieve of Eratosthenes function.

Prime Factorization Class

This class decomposes numbers into their prime factors:

- **Prime Factorization:** Factors an input N by iterating up to $\sqrt{N}$ to identify its prime factors. Outputs a list of pairs [prime factor, exponent] for each factor.

GCD and LCM Computation Class

This class calculates the GCD and LCM of two numbers:

- **gcd:** - Implements the Euclidean algorithm for computation of the GCD for numbers a and b.
- **lcm:** - Computes the LCM using the property $\text{lcm}(a, b) \cdot \text{gcd}(a, b) = abs(a \cdot b))$.
- **gcdWithFactorization:** - Finds the GCD by comparing the prime factorization of two numbers and identifying common factors with the smallest powers.

3. Data Structures Used

- **Boolean Array:** Used in the Sieve of Eratosthenes to mark prime and non-prime numbers.
- **ArrayList:** Stores the list of prime numbers in Sieve algorithm. A 2D ArrayList is utilized for prime factorization.

4. Design Choices

- **Usability:** The separation of the sieve and isPrime functions ensures flexibility for different use cases.

4. Assumptions

- Used the property that GCD a , b is equal to the gcd of their absolute numbers to ensure correct output in all cases.
- Used the property that the prime factors of -x are the same as x, disregarding -1

5. Testing and Results

```
Enter the first integer: 5
Enter the second integer: 7
GCD: 1
GCD Using Factorization: 1
LCM: 35
```

Figure 1

```
Enter the first integer: 100
Enter the second integer: 5
GCD: 5
GCD Using Factorization: 5
LCM: 100
```

Figure 2

```
Enter the first integer: 0
Enter the second integer: 0
GCD of 0 and 0 is undefined.

Process finished with exit code 0
```

Figure 3

```
Enter the first integer: 0
Enter the second integer: 12
GCD: 12
GCD Using Factorization: 12
LCM of 0 and 0 is undefined.
```

Figure 4

```
Enter the first integer: -7
Enter the second integer: 7
GCD: 7
GCD Using Factorization: 7
LCM: 7
```

Figure 5

```
Enter the first integer: -7
Enter the second integer: 5
GCD: 1
GCD Using Factorization: 1
LCM: 35
```

Figure 6

```
Enter the first integer: -7
Enter the second integer: 7
is -7 prime: false
is 7 prime: true

Process finished with exit code 0
```

Figure 7

```
Enter the first integer: 35
Enter the second integer: 5
is 35 prime: false
is 5 prime: true

Process finished with exit code 0
```

Figure 8

```
Enter the first integer: 0
Enter the second integer: 6452
is 0 prime: false
is 6452 prime: false

Process finished with exit code 0
```

Figure 9

```
Enter the first integer: 125
Enter the second integer: 49
Prime factors of 125:
Prime: 5, Exponent: 3
Prime factors of 49:
Prime: 7, Exponent: 2
```

Figure 10

```
Enter the first integer: 35
Enter the second integer: 0
Prime factors of 35:
Prime: 5, Exponent: 1
Prime: 7, Exponent: 1
Prime factors of 0:
```

Figure 11

```
Enter the first integer: -49
Enter the second integer: 49
Prime factors of -49:
Prime: 7, Exponent: 2
Prime factors of 49:
Prime: 7, Exponent: 2
```

Figure 12