



DISCRETE(LAB₃)

AMIN MOHAMED AMIN EL-SAYED

21010310

Part(1) : Prime Number Checker

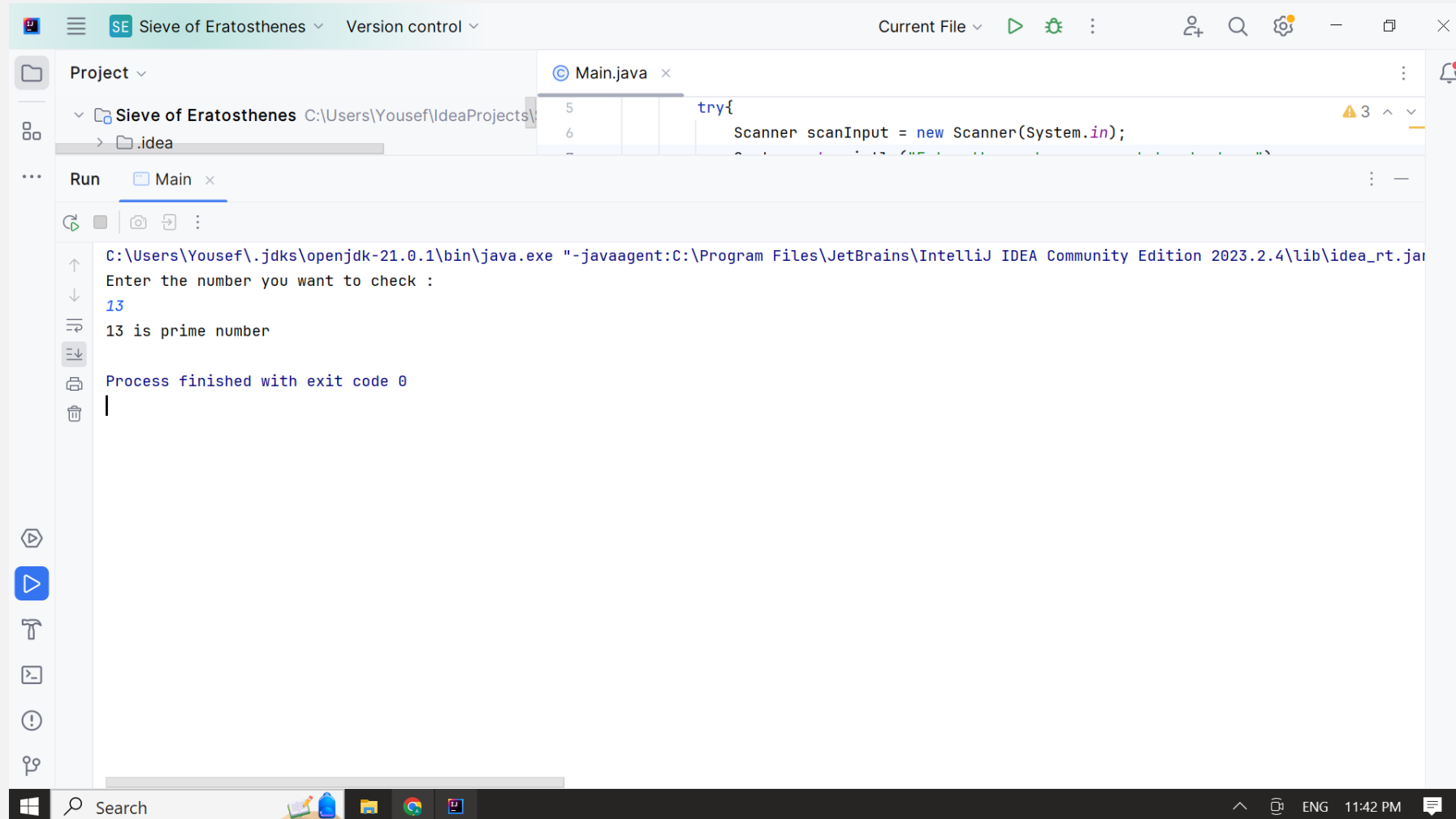
Problem statement

- ❑ Implement a function that determines whether a given positive integer is a prime number or not using Sieve of Eratosthenes

Used data structures. (no used data structures)

Part(1) : Prime Number Checker

Sample runs and test cases



The screenshot displays the IntelliJ IDEA IDE interface. The top toolbar shows the 'Run' button (a green play icon). The 'Project' view on the left shows the project structure: 'Sieve of Eratosthenes' and its subdirectory '.idea'. The 'Run' view at the bottom shows the execution of the 'Main' class. The console output is as follows:

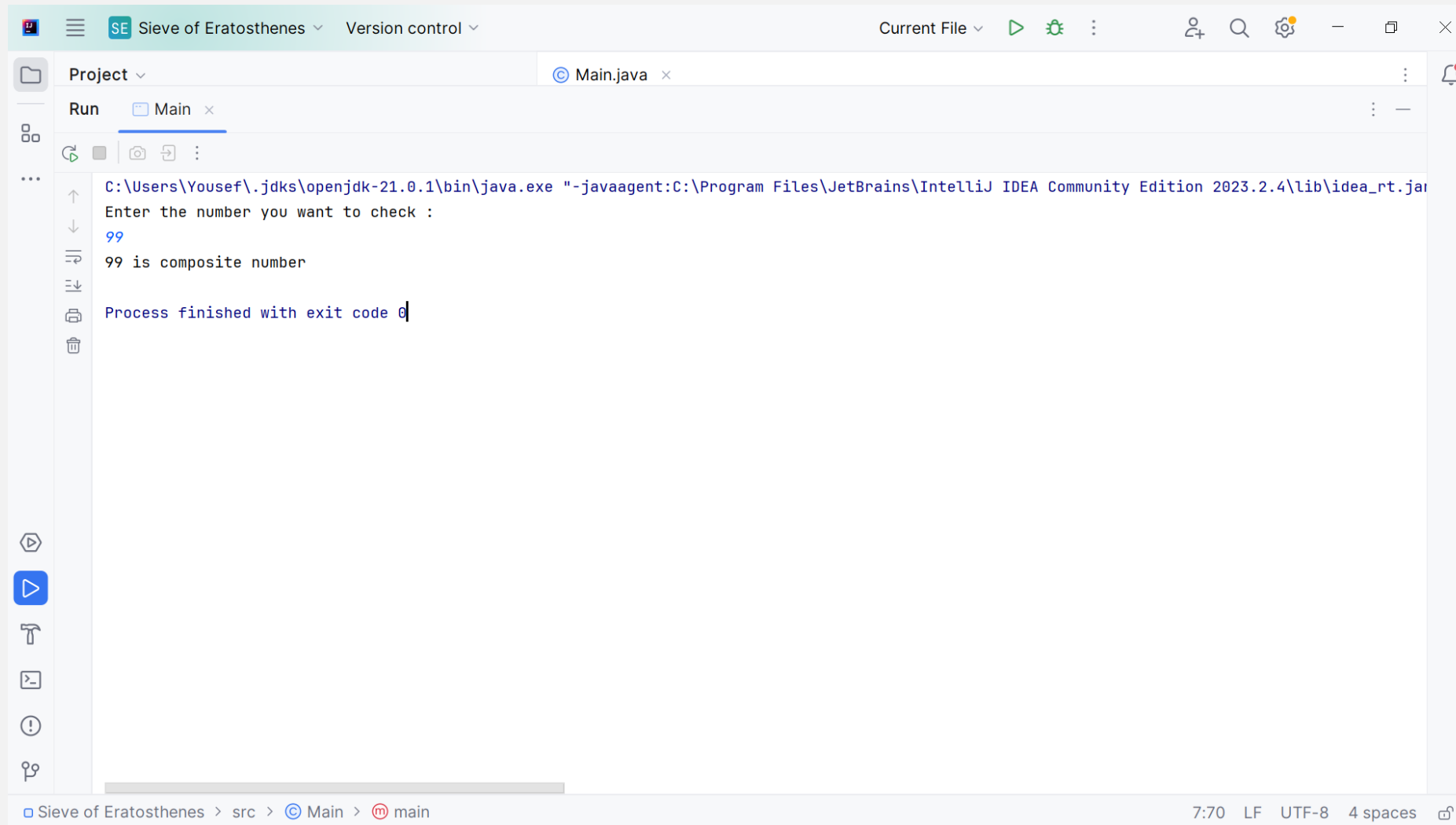
```
C:\Users\Yousef\.jdk\openjdk-21.0.1\bin\java.exe "-javaagent:C:\Program Files\JetBrains\IntelliJ IDEA Community Edition 2023.2.4\lib\idea_rt.jar"
Enter the number you want to check :
13
13 is prime number
Process finished with exit code 0
```

The code editor shows the following Java code in `Main.java`:

```
try{
    Scanner scanInput = new Scanner(System.in);
```

Part(1) : Prime Number Checker

Sample runs and test cases



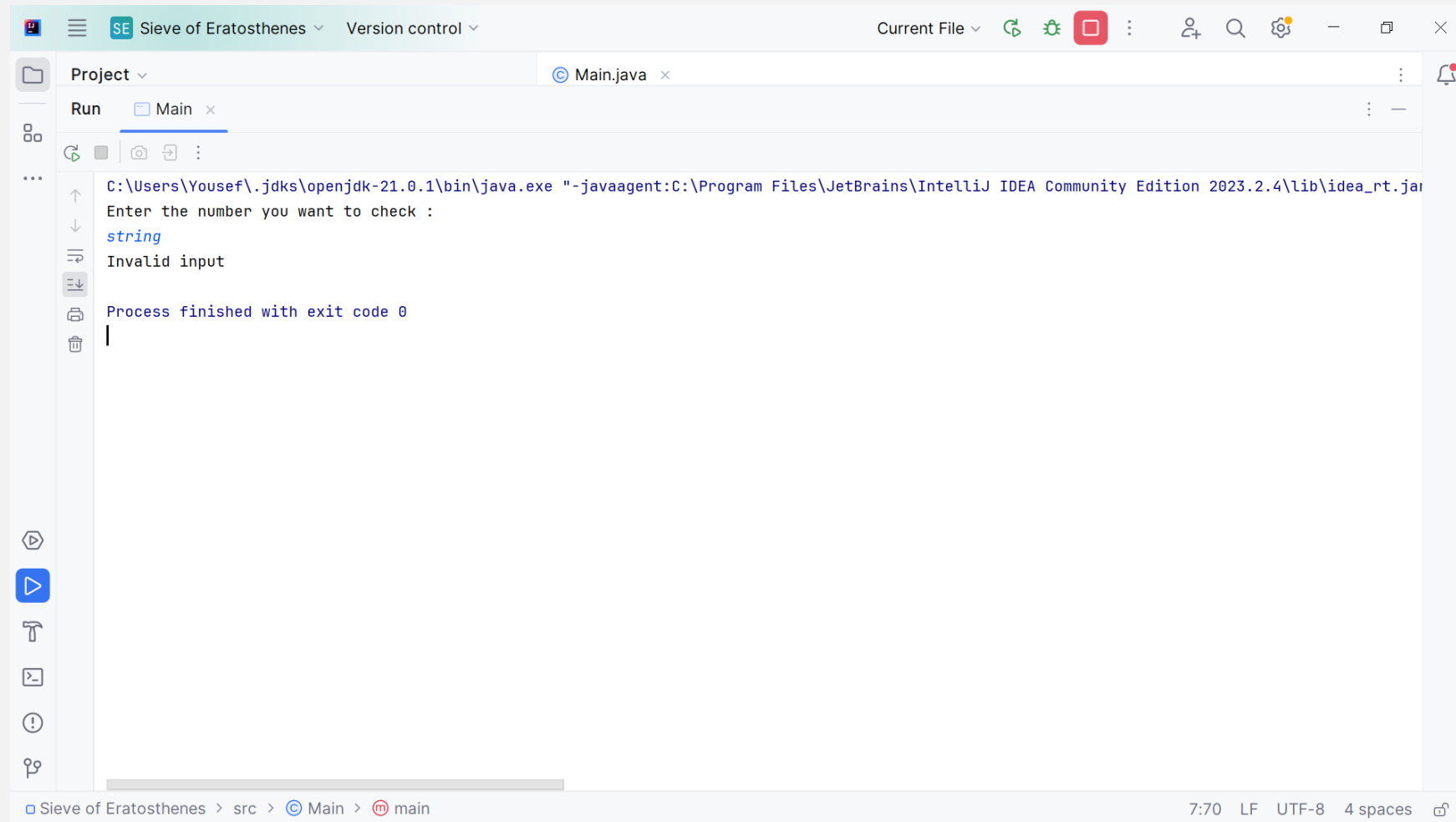
The screenshot displays the IntelliJ IDEA interface with a project named "Sieve of Eratosthenes". The "Run" tab is active, showing the execution of "Main.java". The console output is as follows:

```
C:\Users\Yousef\.jdk\openjdk-21.0.1\bin\java.exe "-javaagent:C:\Program Files\JetBrains\IntelliJ IDEA Community Edition 2023.2.4\lib\idea_rt.jar"
Enter the number you want to check :
99
99 is composite number
Process finished with exit code 0
```

The bottom status bar indicates the file path: Sieve of Eratosthenes > src > Main > main, with a cursor at line 7, column 70, using LF line endings, UTF-8 encoding, and 4 spaces for indentation.

Part(1) : Prime Number Checker

Sample runs and test cases

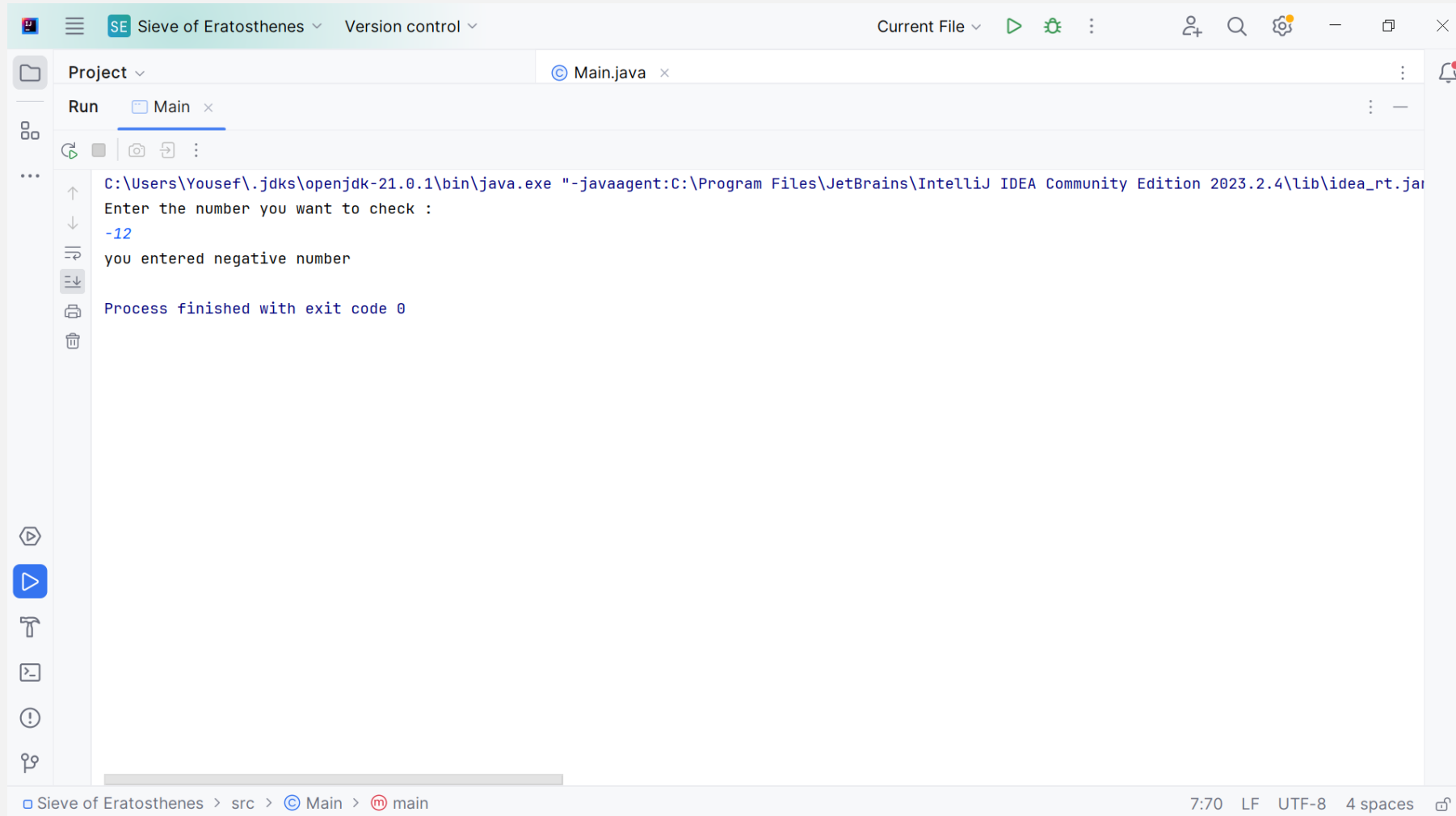


```
C:\Users\Yousef\.jdk\openjdk-21.0.1\bin\java.exe "-javaagent:C:\Program Files\JetBrains\IntelliJ IDEA Community Edition 2023.2.4\lib\idea_rt.jar"
Enter the number you want to check :
string
Invalid input
Process finished with exit code 0
```

The screenshot shows the IntelliJ IDEA interface. The top bar indicates the project is 'Sieve of Eratosthenes' and the current file is 'Main.java'. The Run window is open, showing the execution of the program. The command line shows the Java executable and the IntelliJ IDEA runtime agent. The user input 'string' is shown, followed by the output 'Invalid input'. The process finished with exit code 0. The bottom status bar shows the file path 'Sieve of Eratosthenes > src > Main > main' and the encoding 'UTF-8'.

Part(1) : Prime Number Checker

Sample runs and test cases



The screenshot shows the IntelliJ IDEA interface with a project named "Sieve of Eratosthenes". The "Run" tab is active, displaying the execution of "Main.java". The command line shows the Java executable path and the IntelliJ IDEA runtime jar. The input is "-12", and the output is "you entered negative number". The process finished with exit code 0.

```
C:\Users\Yousef\.jdk\openjdk-21.0.1\bin\java.exe "-javaagent:C:\Program Files\JetBrains\IntelliJ IDEA Community Edition 2023.2.4\lib\idea_rt.jar"
Enter the number you want to check :
-12
you entered negative number
Process finished with exit code 0
```

Part(1) : Prime Number checker

Assumption

- ☐ The user input number in range of int
- ☐ If the number is composite the program would say it

Part(2) : Prime Factorization

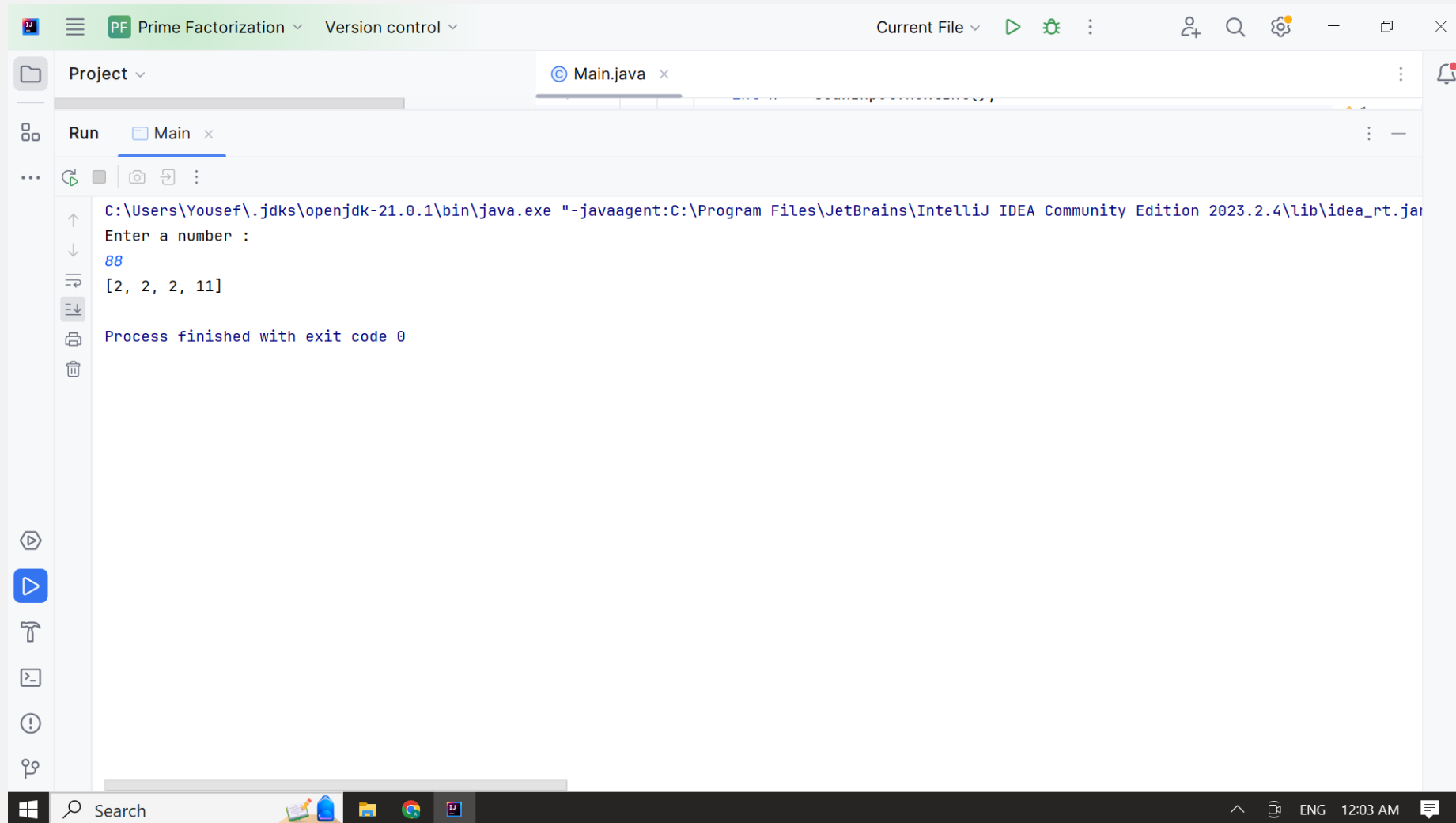
Problem statement

- ❑ Create a function that computes the prime factors of a given integer.

Used data structures (Array list<Integer>)

Part(2) : Prime Factorization

Sample runs and test cases



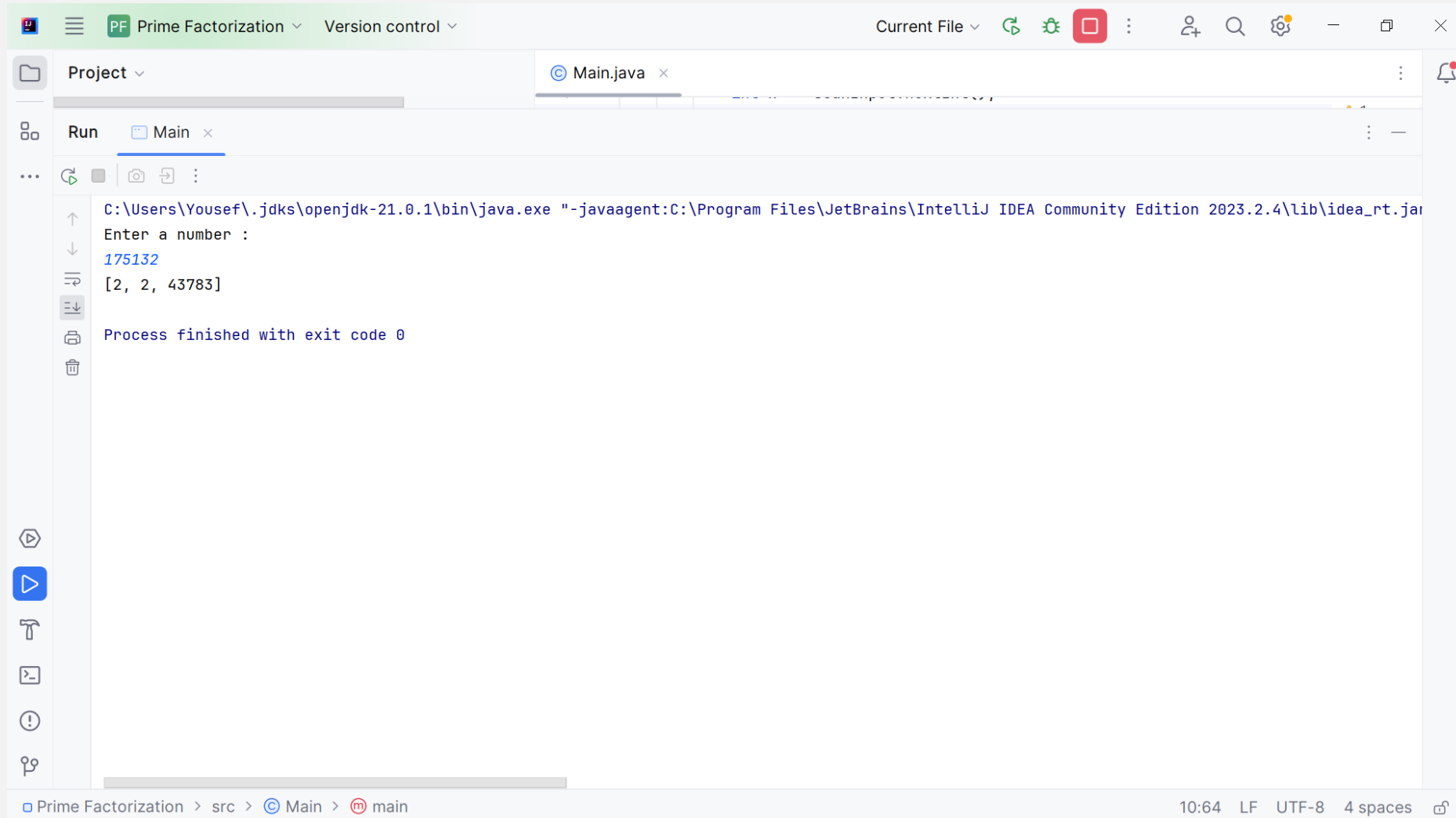
The screenshot displays the IntelliJ IDEA interface for a project named "Prime Factorization". The "Run" tab is active, showing the execution of the "Main" class. The command prompt window shows the following output:

```
C:\Users\Yousef\.jdk\openjdk-21.0.1\bin\java.exe "-javaagent:C:\Program Files\JetBrains\IntelliJ IDEA Community Edition 2023.2.4\lib\idea_rt.jar"
Enter a number :
88
[2, 2, 2, 11]
Process finished with exit code 0
```

The Windows taskbar at the bottom shows the system clock as 12:03 AM on ENG.

Part(2) : Prime Factorization

Sample runs and test cases



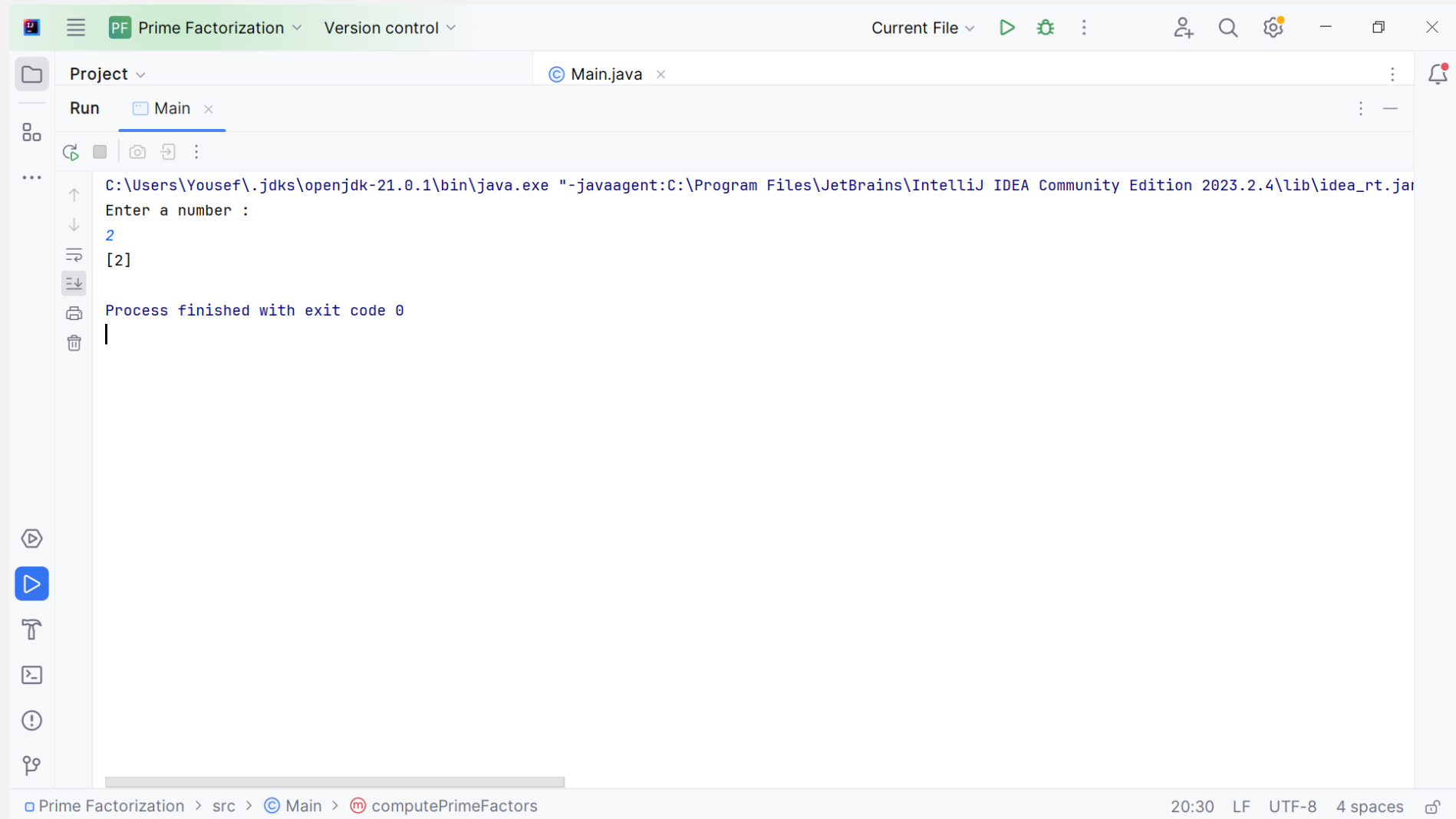
The screenshot shows the IntelliJ IDEA interface with the 'Run' tab active. The Run console displays the following output:

```
C:\Users\Yousef\.jdk\openjdk-21.0.1\bin\java.exe "-javaagent:C:\Program Files\JetBrains\IntelliJ IDEA Community Edition 2023.2.4\lib\idea_rt.jar"
Enter a number :
175132
[2, 2, 43783]
Process finished with exit code 0
```

The bottom status bar indicates the file path: `Prime Factorization > src > Main > main`, the time is 10:64, and the encoding is UTF-8 with 4 spaces.

Part(2) : Prime Factorization

Sample runs and test cases



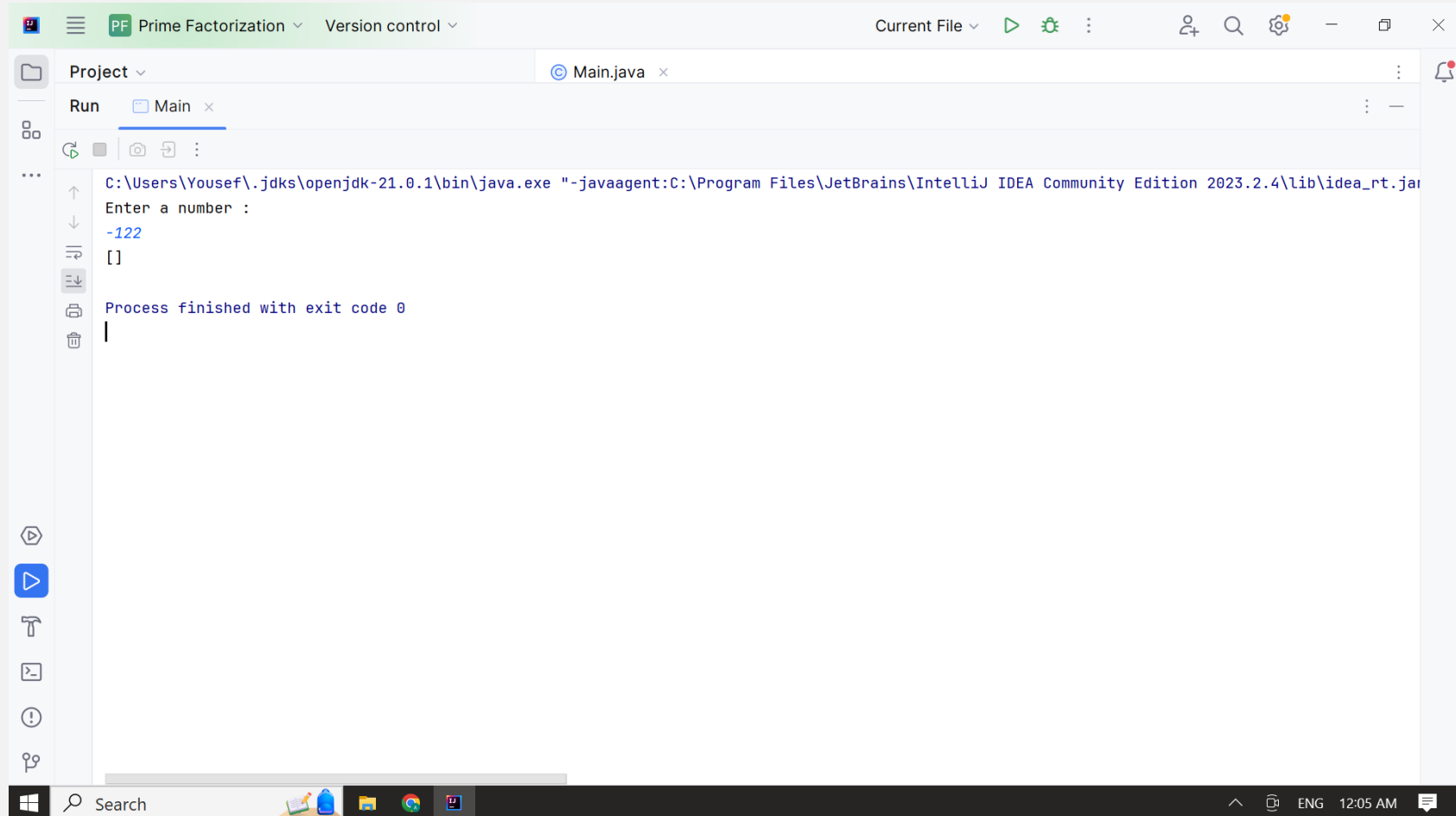
The screenshot displays the IntelliJ IDEA interface. The top toolbar shows the 'Run' button (a green play icon). The 'Run' window is open, showing the execution of 'Main'. The console output is as follows:

```
C:\Users\Yousef\.jdk\openjdk-21.0.1\bin\java.exe "-javaagent:C:\Program Files\JetBrains\IntelliJ IDEA Community Edition 2023.2.4\lib\idea_rt.jar"
Enter a number :
2
[2]
Process finished with exit code 0
```

The bottom status bar indicates the current file is 'Main.java' in the 'computePrimeFactors' method, with a cursor at line 20:30. The encoding is UTF-8 and there are 4 spaces.

Part(2) : Prime Factorization

Sample runs and test cases



```
C:\Users\Yousef\.jdk\openjdk-21.0.1\bin\java.exe "-javaagent:C:\Program Files\JetBrains\IntelliJ IDEA Community Edition 2023.2.4\lib\idea_rt.jar"
Enter a number :
-122
[]
Process finished with exit code 0
```

Part(2) : Prime Factorization

Assumption

- ☐ I assume the the user input integer values that are in the range of int data type.
- ☐ When the number is smaller than 2 , I display [] as these numbers are not in range of prime numbers.

Part(3) : GCD and LCM Computation

Problem statement

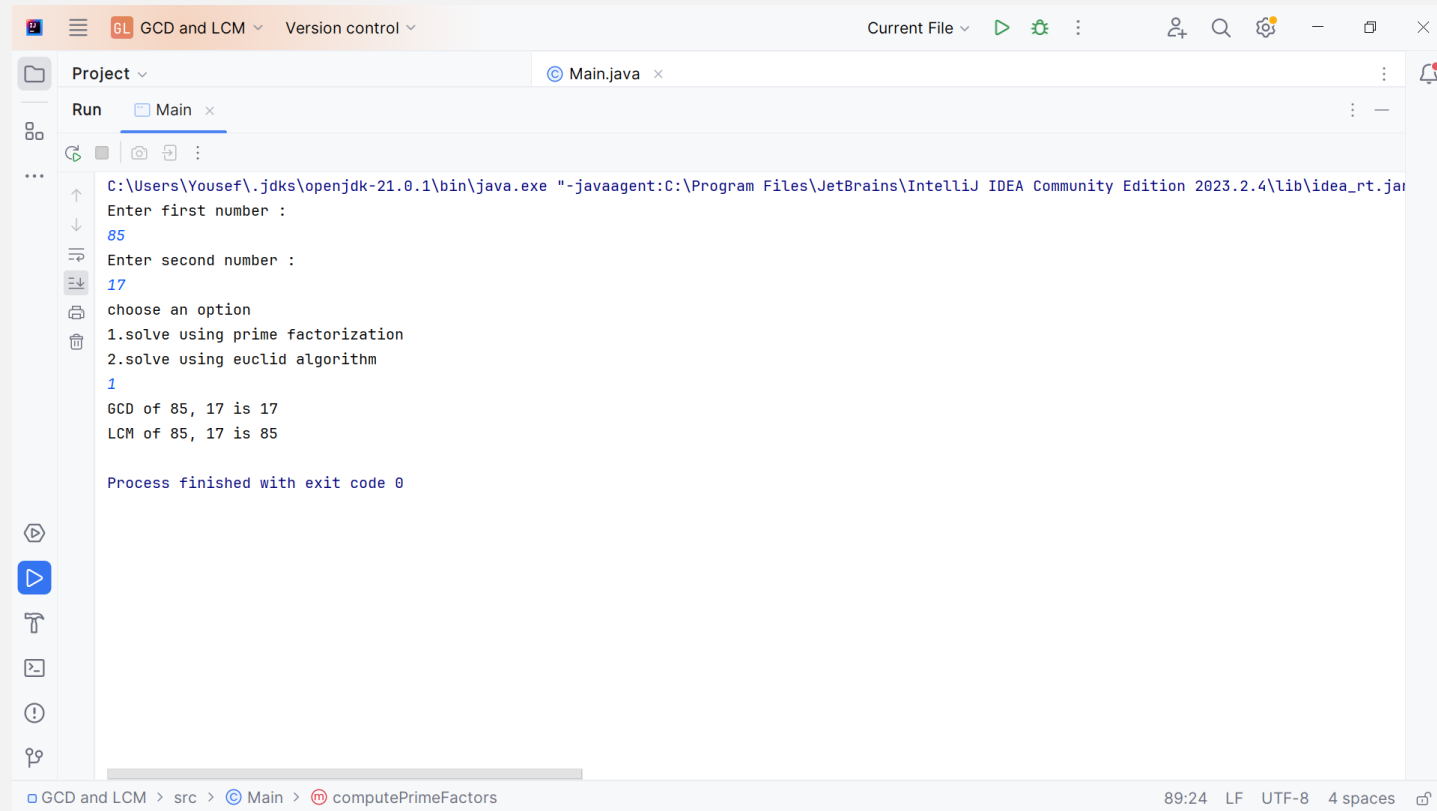
Implement functions to calculate the GCD and LCM of two positive integers.

- a) Using the Euclidean algorithm for GCD computation and the relationship between GCD and LCM to find the LCM.
- b) Using prime factorization.

Used data structures.(Array list<Integer>)

Part(3) : GCD and LCM Computation

Sample runs and test cases



The screenshot displays the IntelliJ IDEA interface with a project named "GCD and LCM". The "Run" tab is active, showing the execution of "Main.java". The command prompt shows the Java command being executed, followed by user input for two numbers (85 and 17) and a choice of algorithm (1 for prime factorization). The output displays the GCD and LCM results for the input numbers.

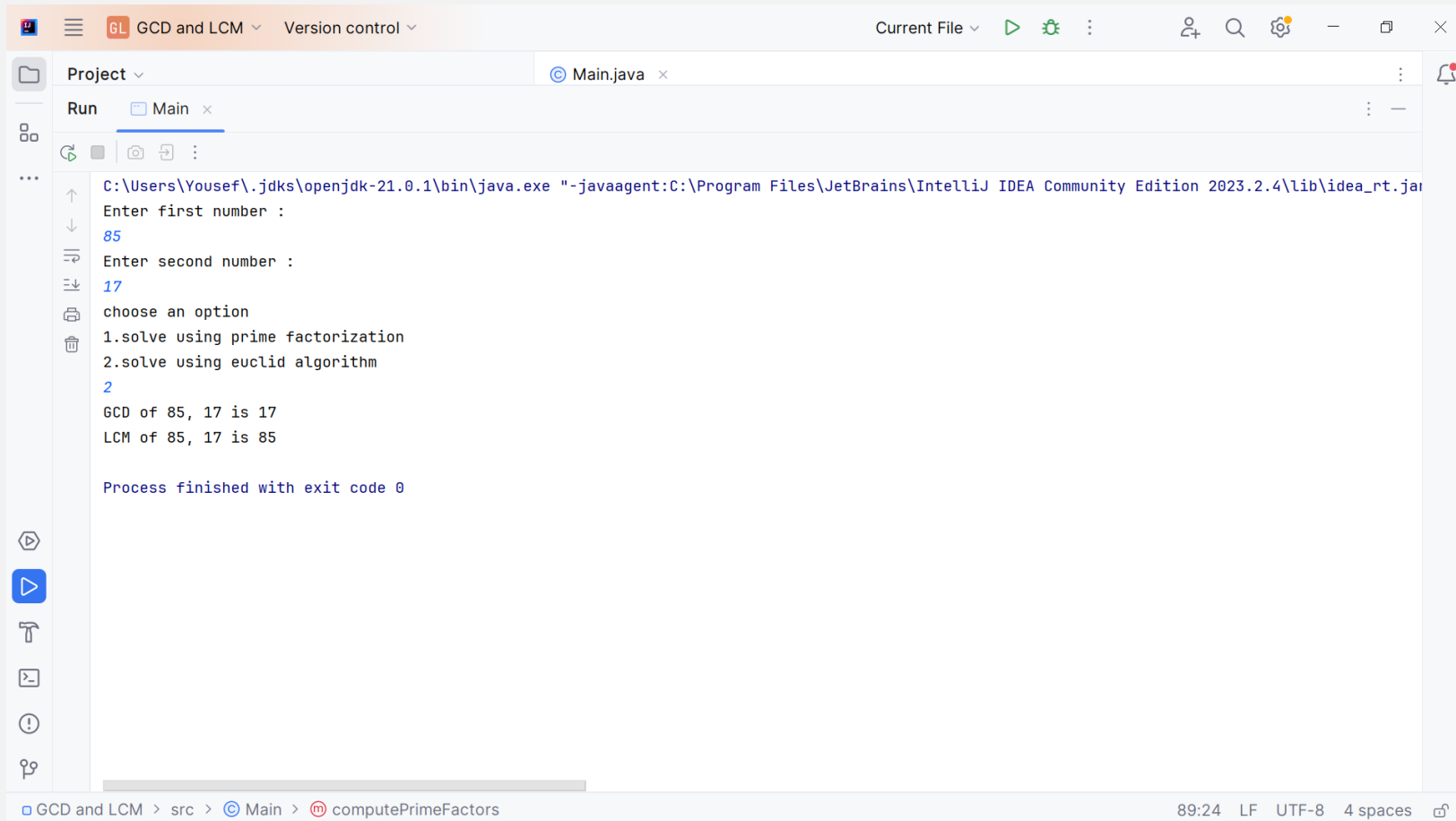
```
C:\Users\Yousef\.jdk\openjdk-21.0.1\bin\java.exe "-javaagent:C:\Program Files\JetBrains\IntelliJ IDEA Community Edition 2023.2.4\lib\idea_rt.jar"
Enter first number :
85
Enter second number :
17
choose an option
1.solve using prime factorization
2.solve using euclid algorithm
1
GCD of 85, 17 is 17
LCM of 85, 17 is 85

Process finished with exit code 0
```

The bottom status bar indicates the file path: "GCD and LCM > src > Main > computePrimeFactors", the time "89:24", and encoding "UTF-8" with "4 spaces".

Part(3) : GCD and LCM Computation

Sample runs and test cases



The screenshot displays the IntelliJ IDEA interface with a project named "GCD and LCM". The "Run" tab is active, showing the execution of "Main.java". The console output is as follows:

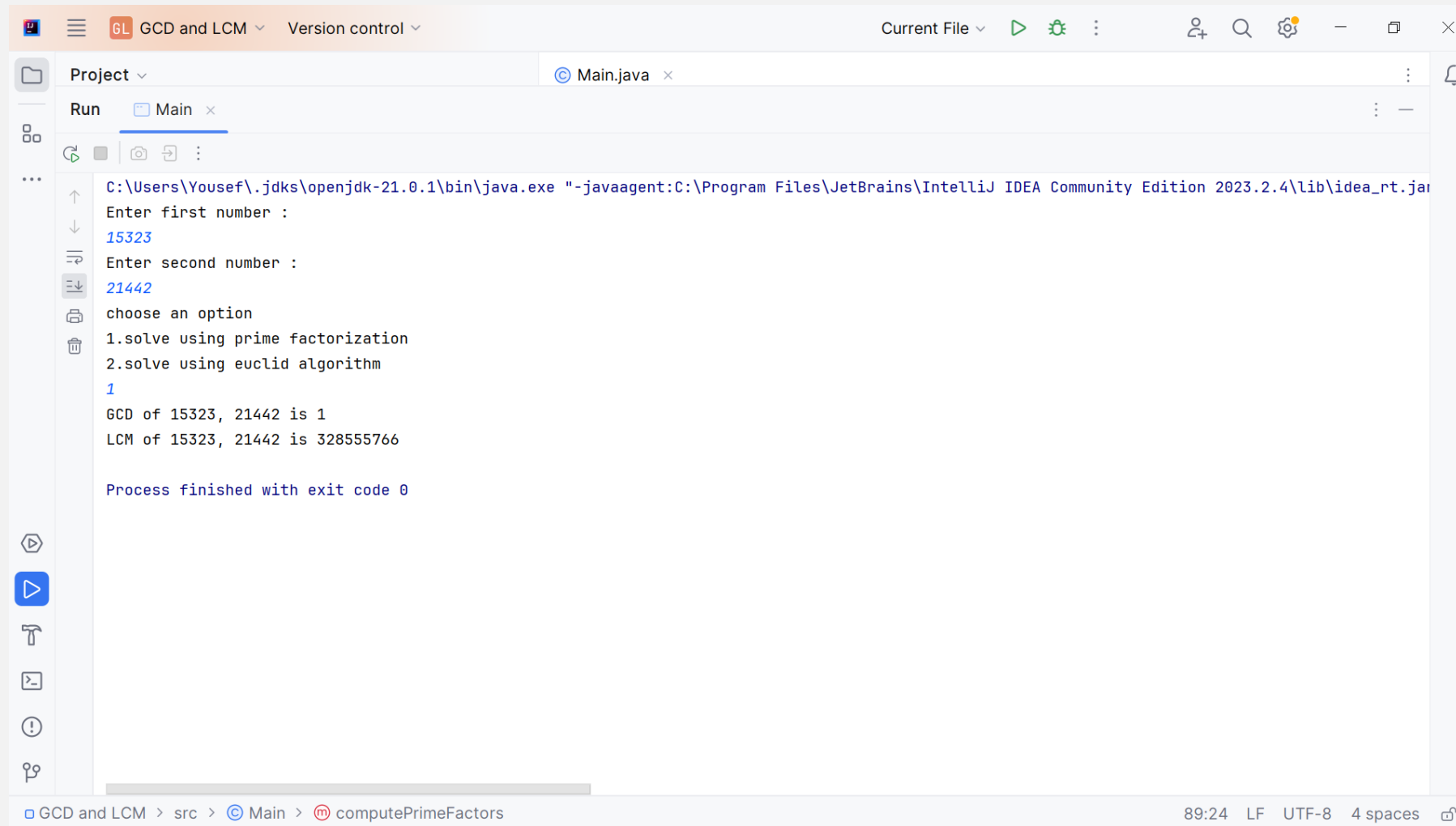
```
C:\Users\Yousef\.jdk\openjdk-21.0.1\bin\java.exe "-javaagent:C:\Program Files\JetBrains\IntelliJ IDEA Community Edition 2023.2.4\lib\idea_rt.jar"
Enter first number :
85
Enter second number :
17
choose an option
1.solve using prime factorization
2.solve using euclid algorithm
2
GCD of 85, 17 is 17
LCM of 85, 17 is 85

Process finished with exit code 0
```

The bottom status bar indicates the file path: "GCD and LCM > src > Main > computePrimeFactors", the cursor position "89:24", and the encoding "LF UTF-8 4 spaces".

Part(3) : GCD and LCM Computation

Sample runs and test cases



The screenshot displays the IntelliJ IDEA interface with a project named "GCD and LCM". The "Run" tab is active, showing the execution of "Main.java". The console output is as follows:

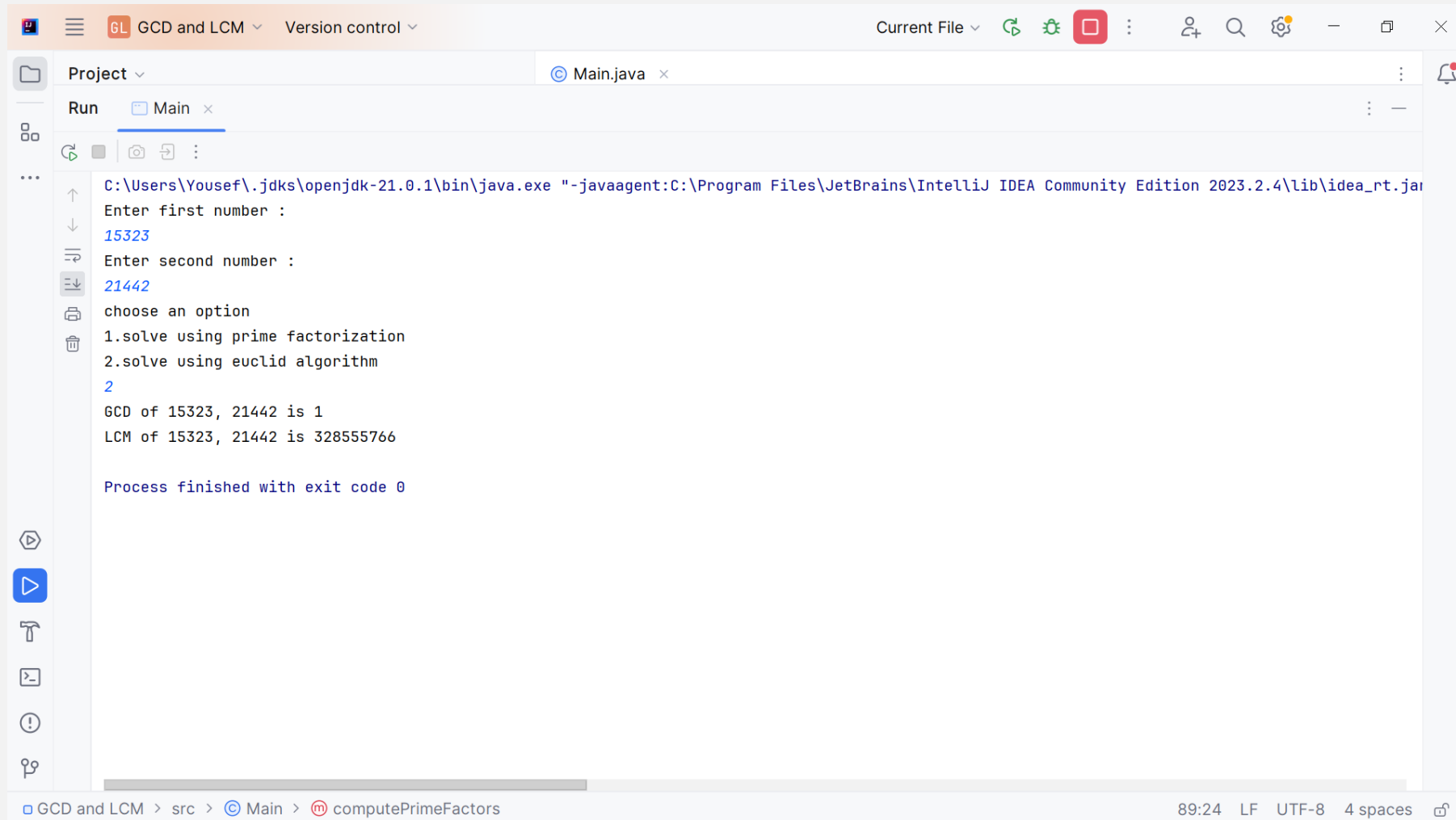
```
C:\Users\Yousef\.jdk\openjdk-21.0.1\bin\java.exe "-javaagent:C:\Program Files\JetBrains\IntelliJ IDEA Community Edition 2023.2.4\lib\idea_rt.jar"
Enter first number :
15323
Enter second number :
21442
choose an option
1.solve using prime factorization
2.solve using euclid algorithm
1
GCD of 15323, 21442 is 1
LCM of 15323, 21442 is 328555766

Process finished with exit code 0
```

The bottom status bar indicates the file path: "GCD and LCM > src > Main > computePrimeFactors", along with encoding details: "89:24 LF UTF-8 4 spaces".

Part(3) :GCD and LCM Computation

Sample runs and test cases



The screenshot displays the IntelliJ IDEA interface with a project named "GCD and LCM". The "Run" tab is active, showing the execution of "Main.java". The console output shows the program prompting for two numbers, 15323 and 21442, and then asking for an option to solve using prime factorization or Euclid's algorithm. Option 2 is chosen, resulting in a GCD of 1 and an LCM of 328555766. The process finishes with exit code 0.

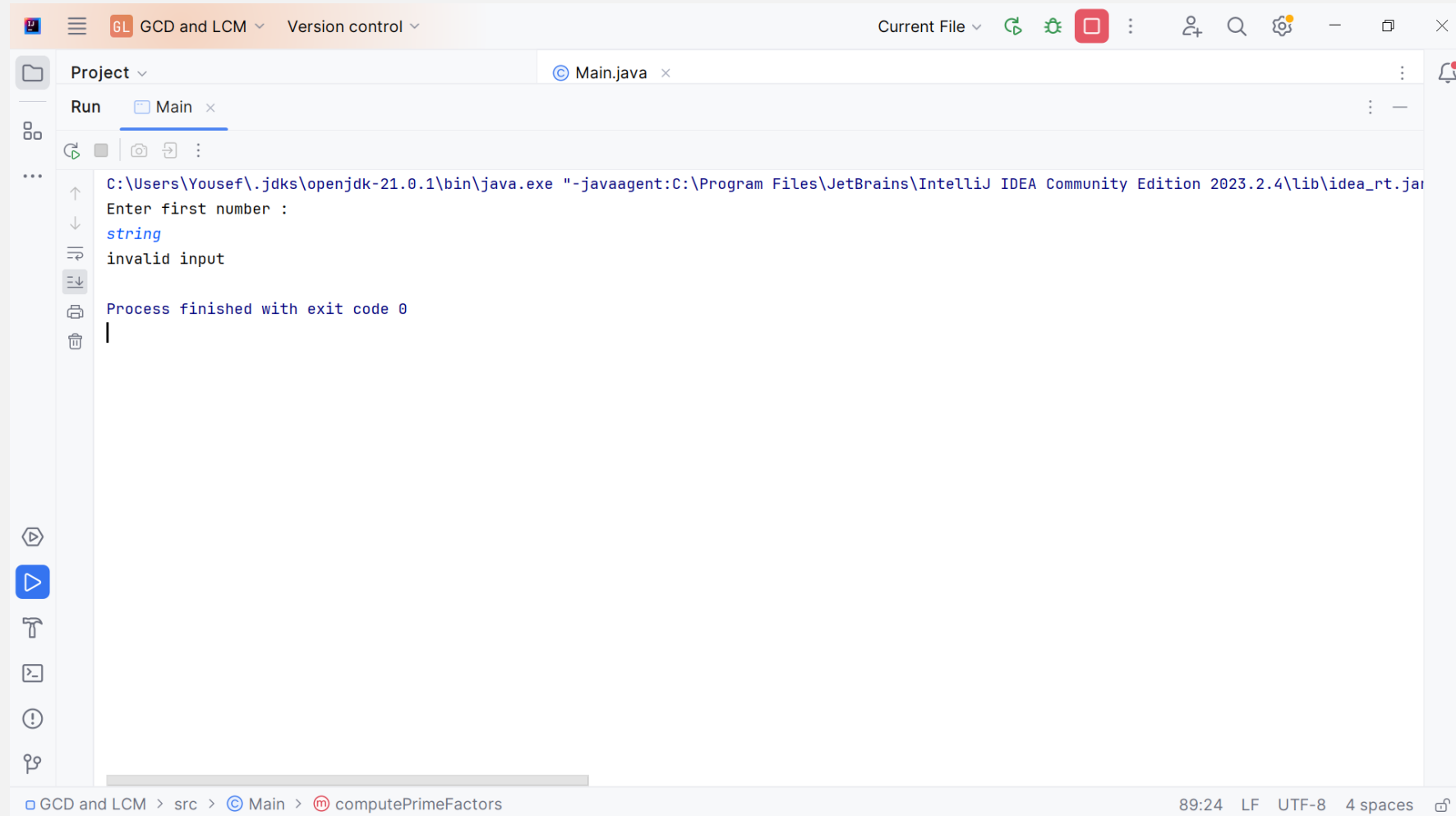
```
C:\Users\Yousef\.jdk\openjdk-21.0.1\bin\java.exe "-javaagent:C:\Program Files\JetBrains\IntelliJ IDEA Community Edition 2023.2.4\lib\idea_rt.jar"
Enter first number :
15323
Enter second number :
21442
choose an option
1.solve using prime factorization
2.solve using euclid algorithm
2
GCD of 15323, 21442 is 1
LCM of 15323, 21442 is 328555766

Process finished with exit code 0
```

Bottom status bar: GCD and LCM > src > Main > computePrimeFactors | 89:24 LF UTF-8 4 spaces

Part(3) : GCD and LCM Computation

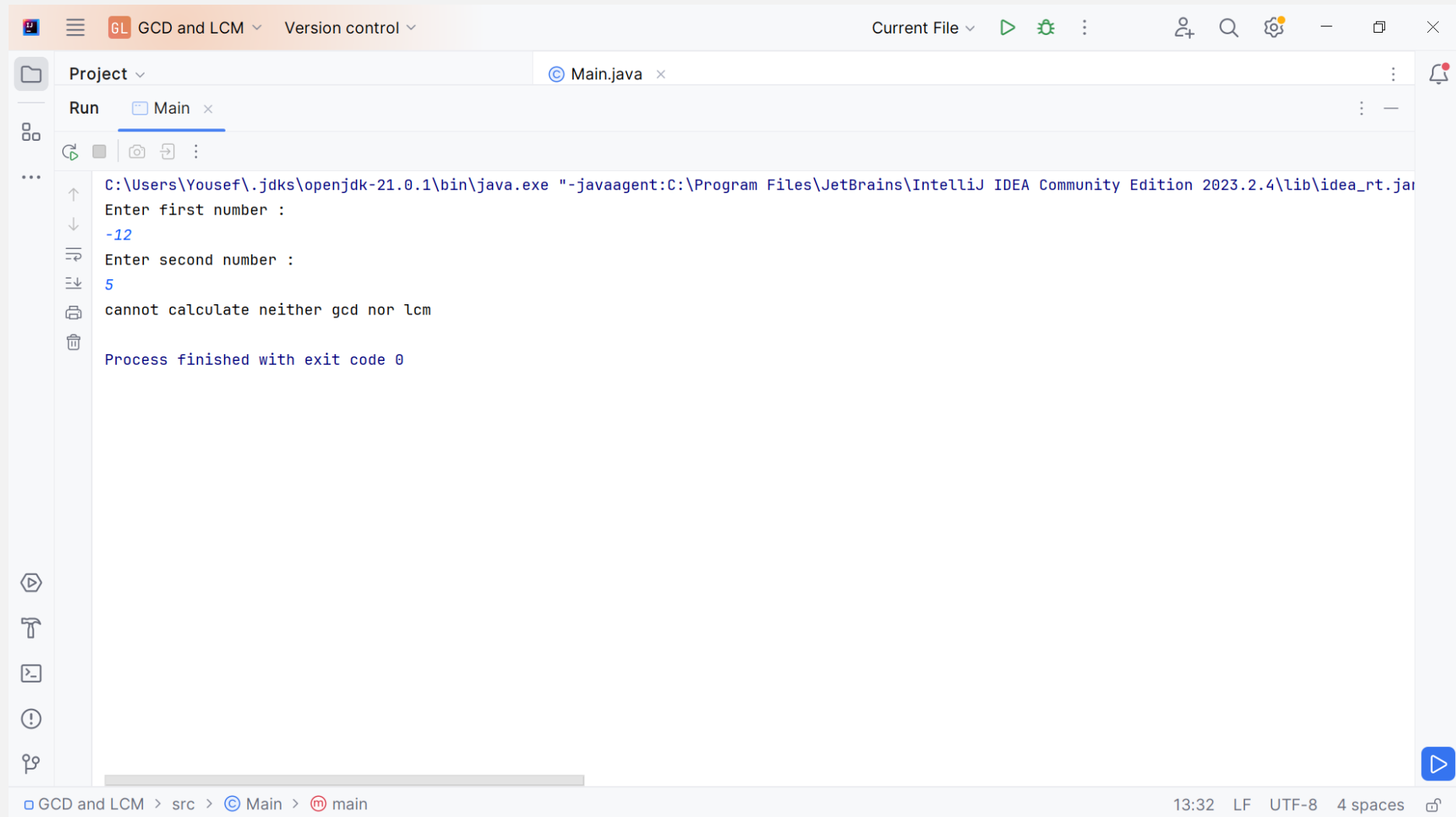
Sample runs and test cases



```
C:\Users\Yousef\.jdk\openjdk-21.0.1\bin\java.exe "-javaagent:C:\Program Files\JetBrains\IntelliJ IDEA Community Edition 2023.2.4\lib\idea_rt.jar"
Enter first number :
string
invalid input
Process finished with exit code 0
```

Part(3) : GCD and LCM Computation

Sample runs and test cases



The screenshot displays the IntelliJ IDEA interface with a project named 'GCD and LCM'. The 'Run' tab is active, showing the execution of 'Main.java'. The command prompt shows the Java command being executed, followed by user input for two numbers: -12 and 5. The program's output indicates that it cannot calculate the GCD or LCM for these inputs, and it finishes with an exit code of 0.

```
C:\Users\Yousef\.jdk\openjdk-21.0.1\bin\java.exe "-javaagent:C:\Program Files\JetBrains\IntelliJ IDEA Community Edition 2023.2.4\lib\idea_rt.jar"
Enter first number :
-12
Enter second number :
5
cannot calculate neither gcd nor lcm
Process finished with exit code 0
```

Bottom status bar: GCD and LCM > src > Main > main | 13:32 | LF | UTF-8 | 4 spaces

Part(3) : GCD and LCM Computation

Assumption

- ❑ I assume the the user input integer values that are in the range of int data type.

Part(4) : Chinese Remainder Theorem

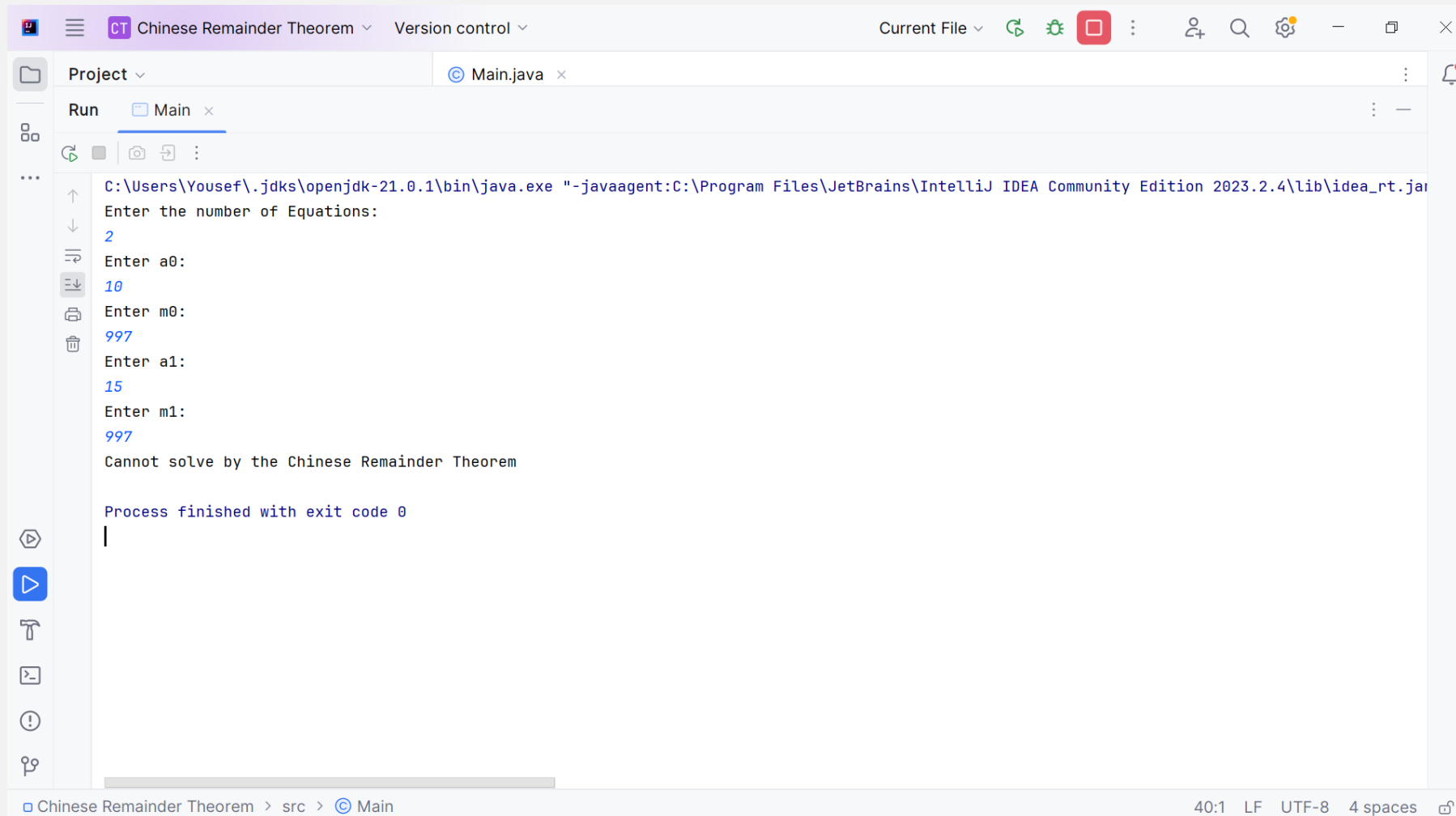
Problem statement

- Implement Chinese remainder theorem that takes as input $m_1, m_2, m_3, \dots, m_n$ that are pairwise relatively prime and (a_1, a_2, \dots, a_n) and calculates x such that $x = a_1 \pmod{m_1}$, $x = a_2 \pmod{m_2}$, ..., $x = a_n \pmod{m_n}$

Used data structures. (Arrays)

Part(4) : Chinese Remainder Theorem

Sample runs and test cases



The screenshot shows the IntelliJ IDEA interface with a project named "Chinese Remainder Theorem". The "Run" tab is active, displaying the output of the program. The program prompts the user to enter the number of equations, followed by pairs of (a, m) values. In this run, the user entered 2 equations with (a0=10, m0=997) and (a1=15, m1=997). The program output indicates it cannot solve the system by the Chinese Remainder Theorem and finishes with exit code 0.

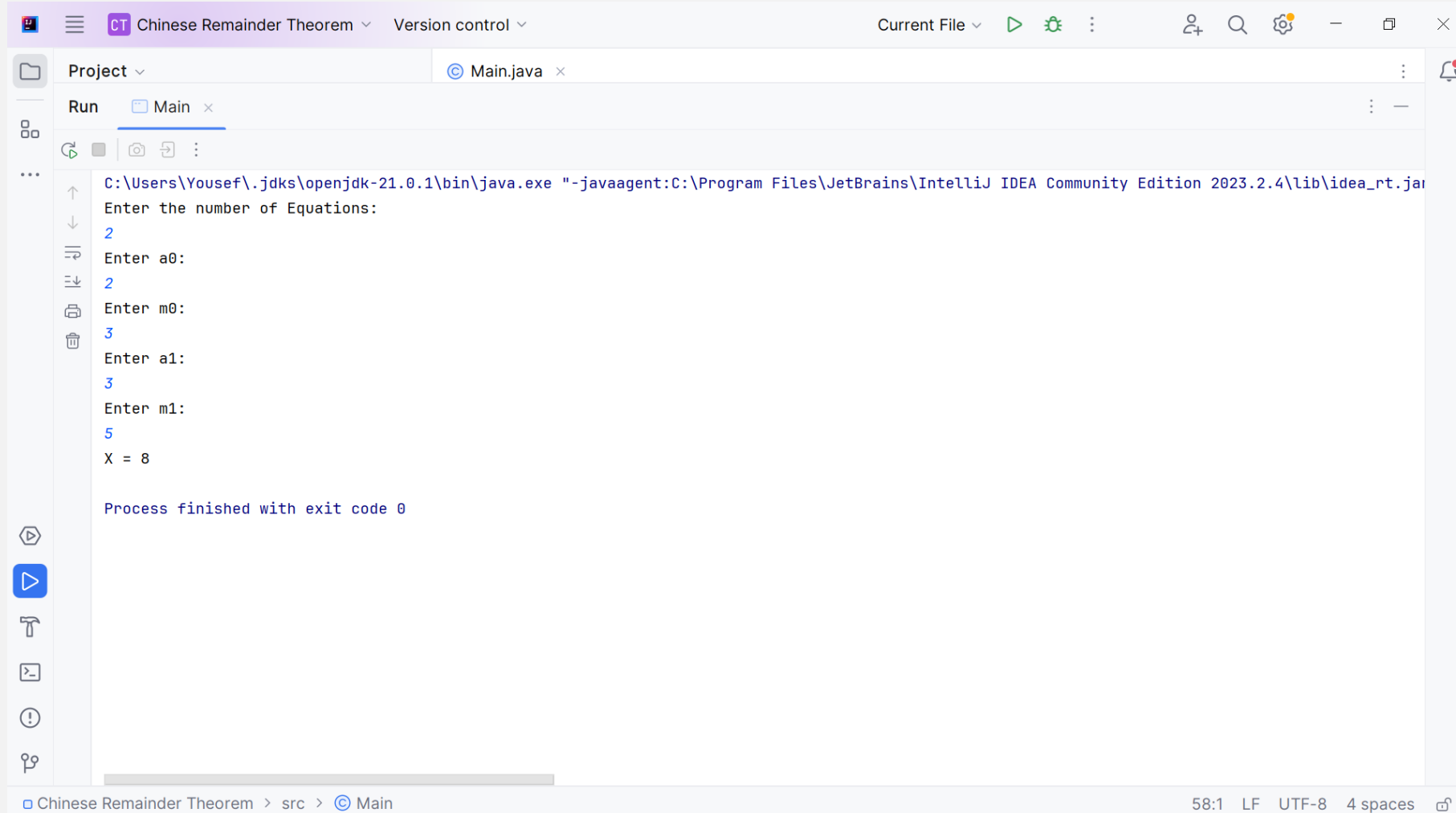
```
C:\Users\Yousef\.jdk\openjdk-21.0.1\bin\java.exe "-javaagent:C:\Program Files\JetBrains\IntelliJ IDEA Community Edition 2023.2.4\lib\idea_rt.jar"
Enter the number of Equations:
2
Enter a0:
10
Enter m0:
997
Enter a1:
15
Enter m1:
997
Cannot solve by the Chinese Remainder Theorem

Process finished with exit code 0
|
```

Chinese Remainder Theorem > src > Main 40:1 LF UTF-8 4 spaces

Part(2) : Prime Factorization

Sample runs and test cases



The screenshot shows the IntelliJ IDEA interface with a project named "Chinese Remainder Theorem". The "Run" tab is active, displaying the execution of "Main.java". The command prompt shows the Java command being executed, followed by user input for the number of equations (2), and coefficients (a0=2, m0=3, a1=3, m1=5). The output shows the result X = 8 and a successful exit code.

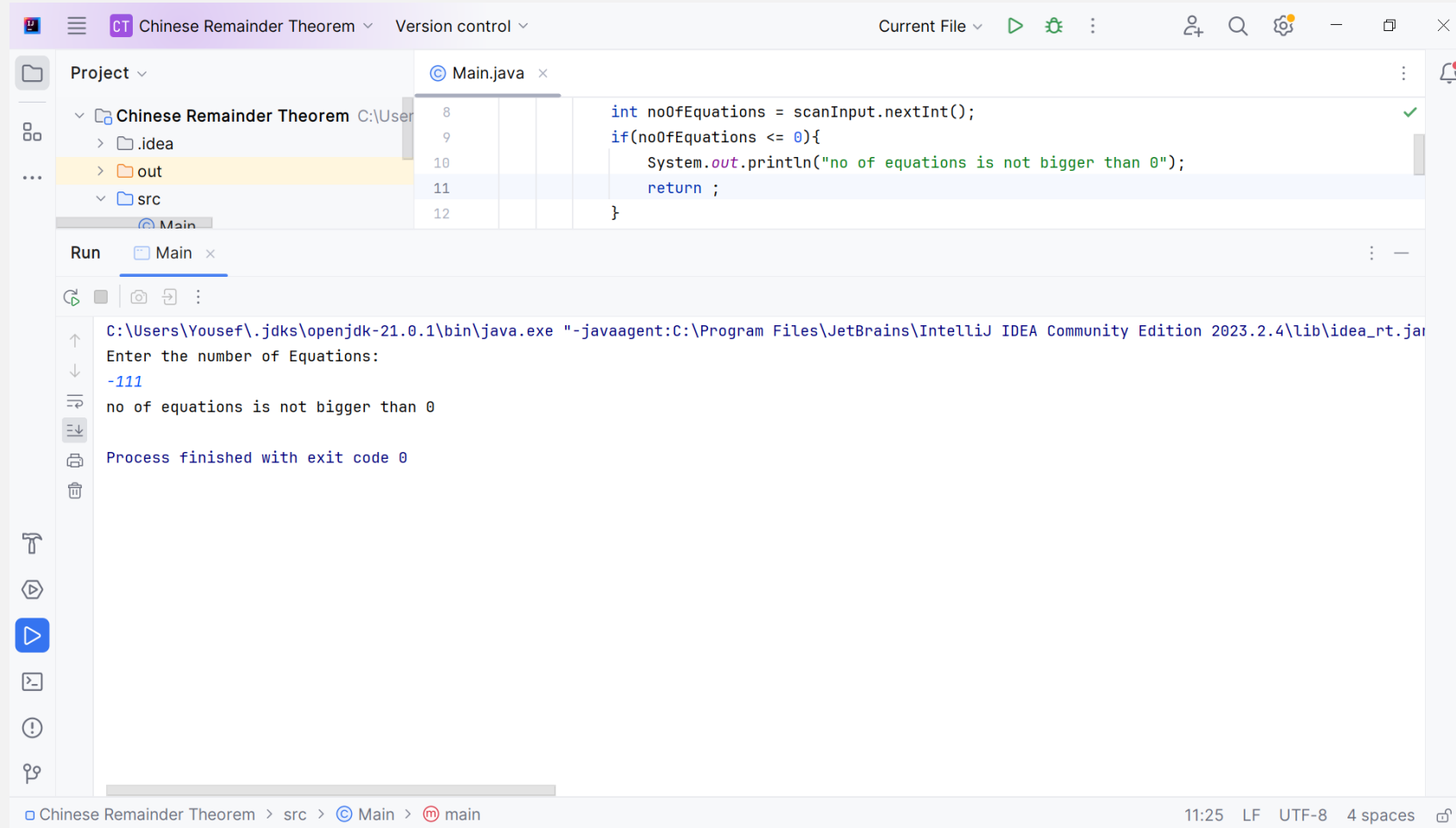
```
C:\Users\Yousef\.jdk\openjdk-21.0.1\bin\java.exe "-javaagent:C:\Program Files\JetBrains\IntelliJ IDEA Community Edition 2023.2.4\lib\idea_rt.jar"
Enter the number of Equations:
2
Enter a0:
2
Enter m0:
3
Enter a1:
3
Enter m1:
5
X = 8

Process finished with exit code 0
```

Chinese Remainder Theorem > src > Main 58:1 LF UTF-8 4 spaces

Part(4) : Chinese Remainder Theorem

Sample runs and test cases



The screenshot displays the IntelliJ IDEA interface for a project named "Chinese Remainder Theorem". The "Project" view on the left shows the directory structure: `Chinese Remainder Theorem` (C:\User) containing `.idea`, `out`, and `src` folders. The `Main.java` file is open in the editor, showing the following code:

```
8  
9  
10 int noOfEquations = scanInput.nextInt();  
11 if(noOfEquations <= 0){  
12     System.out.println("no of equations is not bigger than 0");  
13     return ;  
14 }
```

The "Run" tab at the bottom shows the execution output for the "Main" class. The command executed is:

```
C:\Users\Yousef\.jdk\openjdk-21.0.1\bin\java.exe "-javaagent:C:\Program Files\JetBrains\IntelliJ IDEA Community Edition 2023.2.4\lib\idea_rt.jar"
```

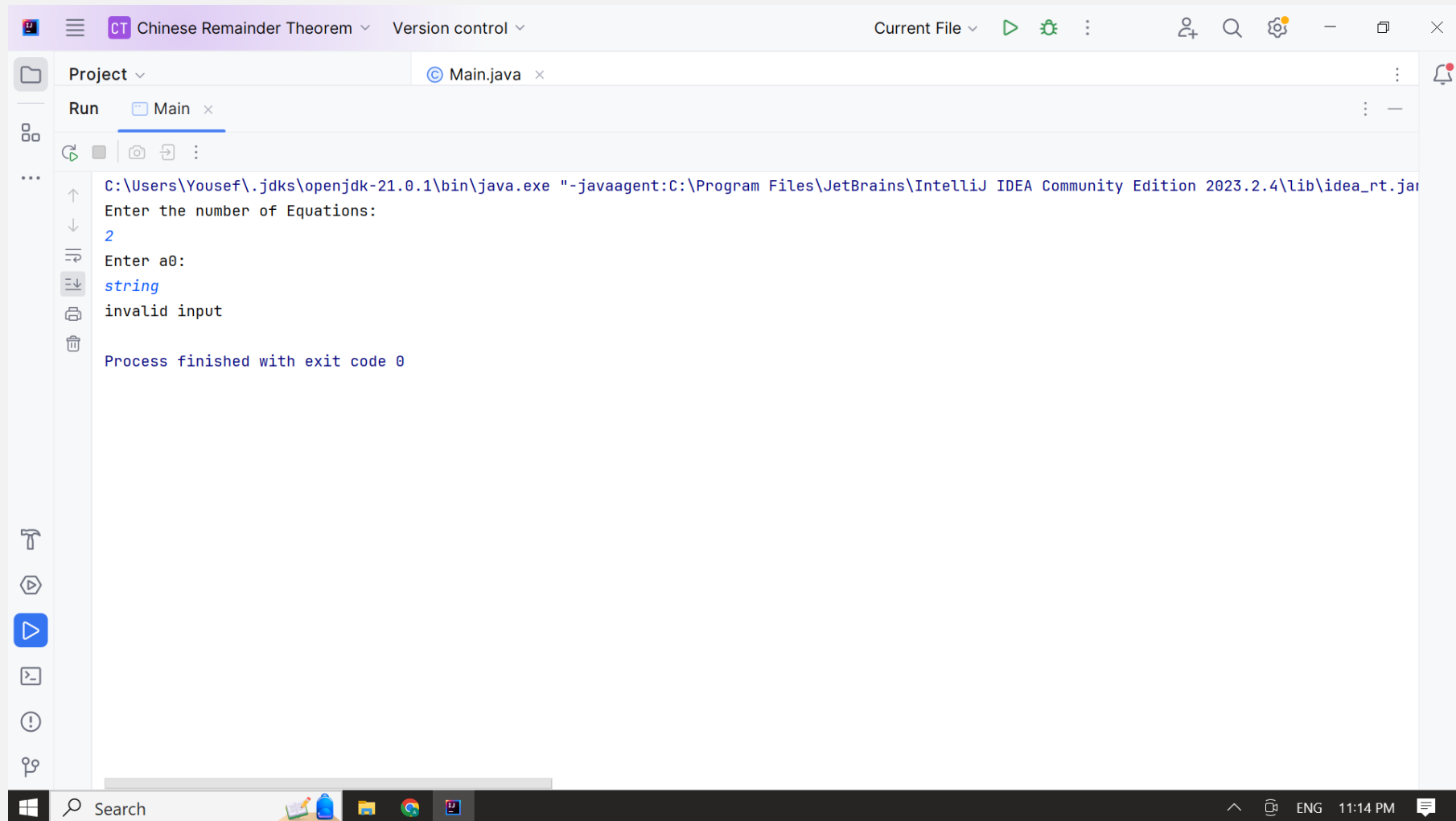
The input provided was `-111`, and the output was:

```
Enter the number of Equations:  
-111  
no of equations is not bigger than 0  
Process finished with exit code 0
```

The status bar at the bottom indicates the current file is `main` in the `src` directory of the `Chinese Remainder Theorem` project, with a timestamp of 11:25, LF line endings, UTF-8 encoding, and 4 spaces for indentation.

Part(4) : Chinese Remainder Theorem

Sample runs and test cases



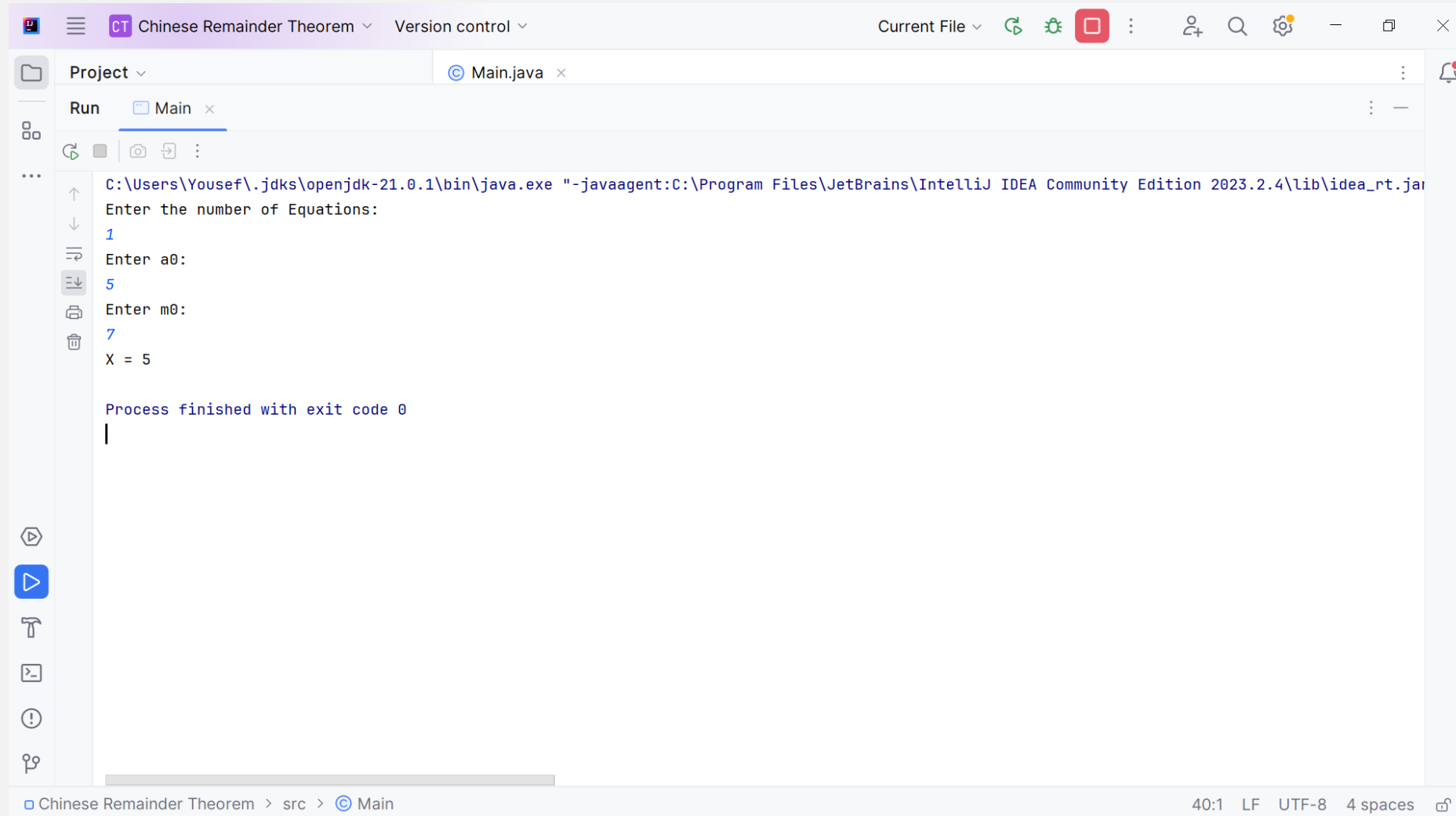
The screenshot displays the IntelliJ IDEA interface with a project named "Chinese Remainder Theorem". The "Run" tab is active, showing the execution of "Main.java". The command prompt shows the following sequence of events:

```
C:\Users\Yousef\.jdk\openjdk-21.0.1\bin\java.exe "-javaagent:C:\Program Files\JetBrains\IntelliJ IDEA Community Edition 2023.2.4\lib\idea_rt.jar"
Enter the number of Equations:
2
Enter a0:
string
invalid input
Process finished with exit code 0
```

The Windows taskbar at the bottom shows the system clock at 11:14 PM and the language set to ENG.

Part(4) : Chinese Remainder Theorem

Sample runs and test cases



The screenshot shows the IntelliJ IDEA interface with a project named "Chinese Remainder Theorem". The "Run" tab is active, displaying the execution of the "Main" class. The command prompt shows the following input and output:

```
C:\Users\Yousef\.jdk\openjdk-21.0.1\bin\java.exe "-javaagent:C:\Program Files\JetBrains\IntelliJ IDEA Community Edition 2023.2.4\lib\idea_rt.jar"
Enter the number of Equations:
1
Enter a0:
5
Enter m0:
7
X = 5

Process finished with exit code 0
```

The bottom status bar indicates the file encoding is UTF-8, with 4 spaces and a line length of 40:1.

Part(4) : Chinese Remainder Theorem

Sample runs and test cases

```
CT Chinese Remainder Theorem Version control
Current File
Main.java
System.out.println("Cannot solve by the Chinese Remainder Theorem");
} catch (TestMismatchException e) {
}

Run Main
C:\Users\Yousef\.jdk\openjdk-21.0.1\bin\java.exe "-javaagent:C:\Program Files\JetBrains\IntelliJ IDEA Community Edition 2023.2.4\lib\idea_rt.jar"
Enter the number of Equations:
3
Enter a0: |
2
Enter m0:
3
Enter a1:
-2
Enter m1:
7
Enter a2:
3
Enter m2:
5
X = 68

Process finished with exit code 0

Chinese Remainder Theorem > src > Main > main 37:42 (49 chars, 1 line break) LF UTF-8 4 spaces
```

Part(4) : Chinese Remainder Theorem

Assumption

- ❑ I assume the the user input integer values that are in the range of int data type.
- ❑ The program handles the following Exceptions
NumberFormatException , ArithmeticException (in case some value has no modular inverse), InputMismatchException

THANKYOU

Amin Mohamed Amin El-Sayed 

es-amin.mohamedamin2026@alexu.edu.eg 