자바 프로그래밍 1 분반

Lab 1

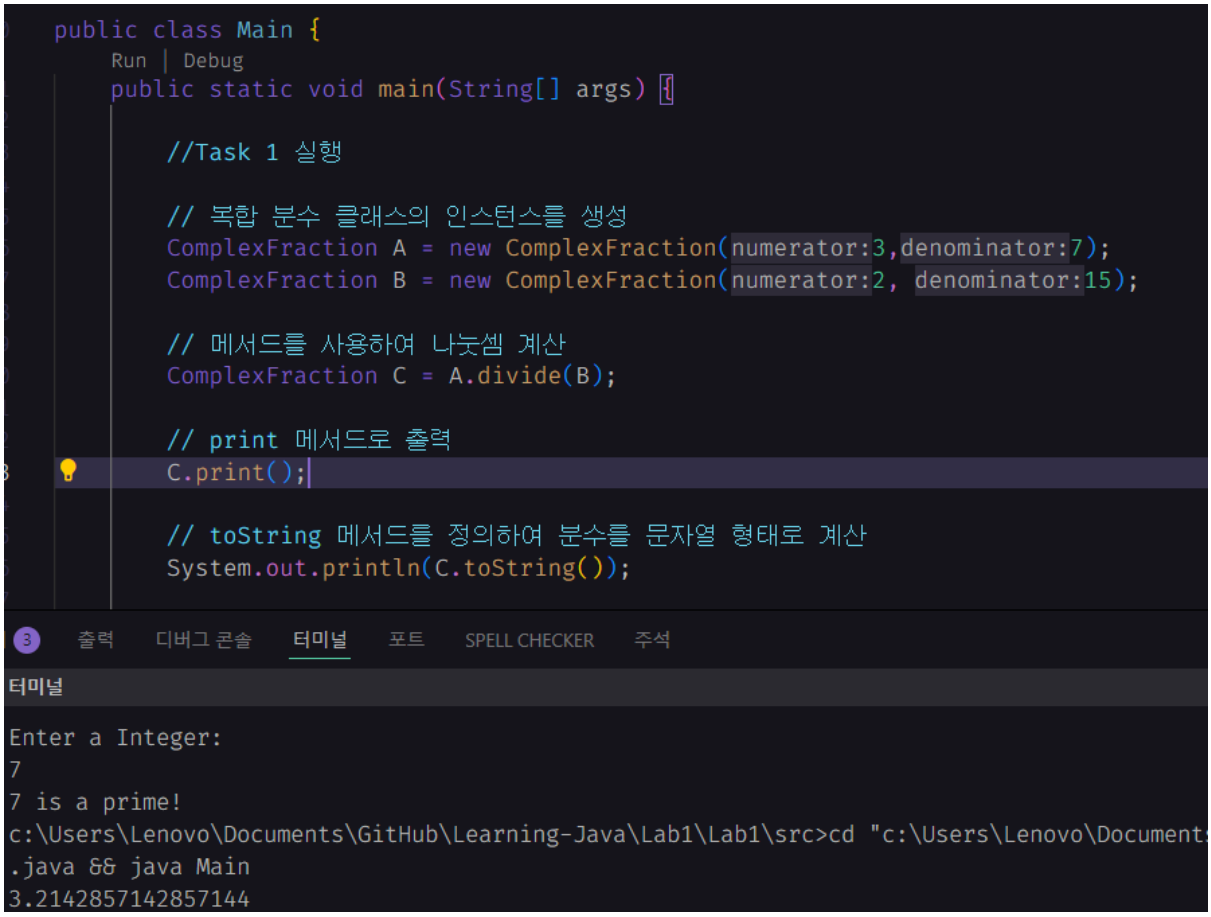
32241484 류지성

2025-03-24

Task 1.

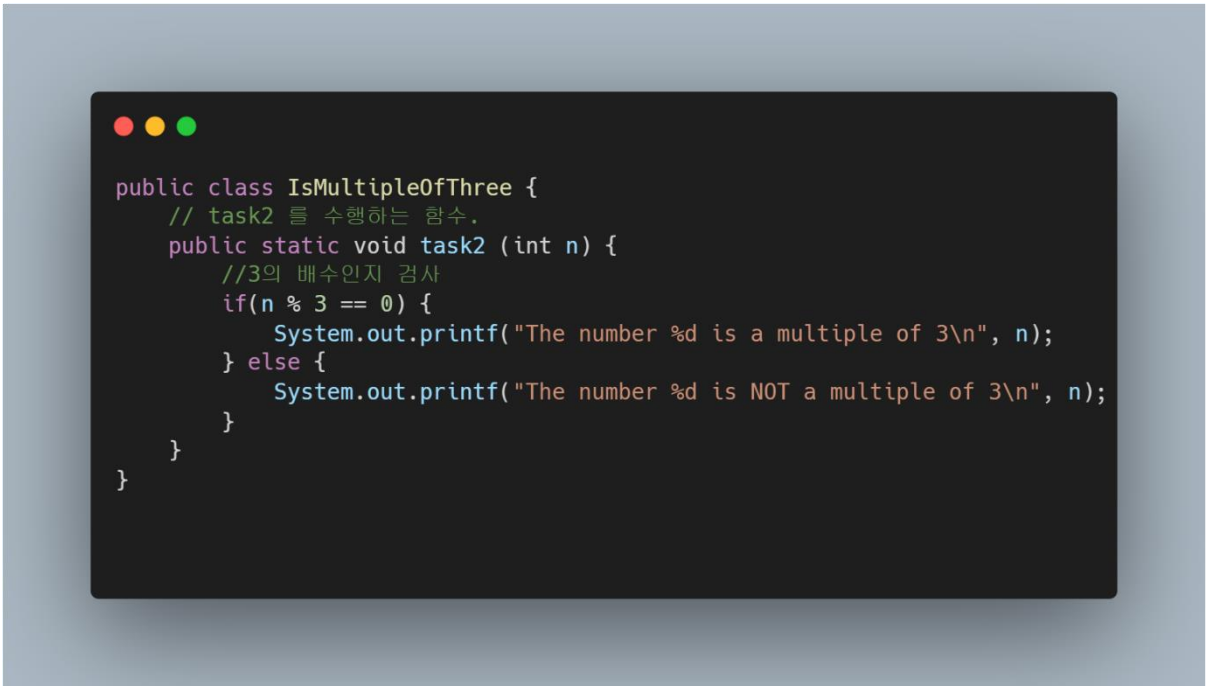
```java
public class ComplexFraction {
    // 분자와 분모를 가집니다.
    private int numerator;
    private int denominator;

    //생성자
    public ComplexFraction(int numerator, int denominator) {
        // 0인 분수는 정의될 수 없습니다.
        if(denominator == 0) {
            throw new IllegalArgumentException("분수는 0이 될 수 없습니다.");
        }
        this.numerator = numerator;
        this.denominator = denominator;
    }

    //나누기 메서드
    public ComplexFraction divide (ComplexFraction other) {
        int returnNumerator = this.numerator * other.denominator;
        int returnDenominator = other.numerator * this.denominator;
        return new ComplexFraction(returnNumerator, returnDenominator);
    }
    //분수를 string format으로 반환합니다.
    public String toString() {
        return numerator + " / " + denominator;
    }
    //분수의 대소비교
    public boolean greaterThan(ComplexFraction other) {
        return this.numerator * other.denominator > other.numerator *
this.denominator;
    }
    //분수가 같은지 확인
    public boolean equalsTo(ComplexFraction other) {
        return this.numerator * other.denominator == other.numerator *
this.denominator;
    }
    //분수를 double 로 출력
    public void print() {
        System.out.println( (double) numerator / denominator ) ;
    }
}
```

분자랑 분모를 int 형 인자로 받는 생성자를 정의합니다. 분모가 0인 경우는 수학적으로 정의될 수 없기에 예외처리를 해줍니다. 분모가 0인 조건을 if 로 검사하고 throw 키워드로 IllegalArgumentException예외 클래스를 만들어서 반환합니다.



Main 함수에서 생성자로 A 분수를 만들고 B 분수를 만듭니다. 그리고 divide 메서드를 통해서 A랑 B 를 나눈 결과를 C 분수로 만듭니다. print 메서드를 통해서 소수Format 으로 출력을 합니다. 추가적으로 toString 메서드로 분수형 format 을 반환하게 만들었습니다. greaterThan 이랑 equalTo 로 비교할 수 있는 메서드도 추가했습니다.

Task 2.

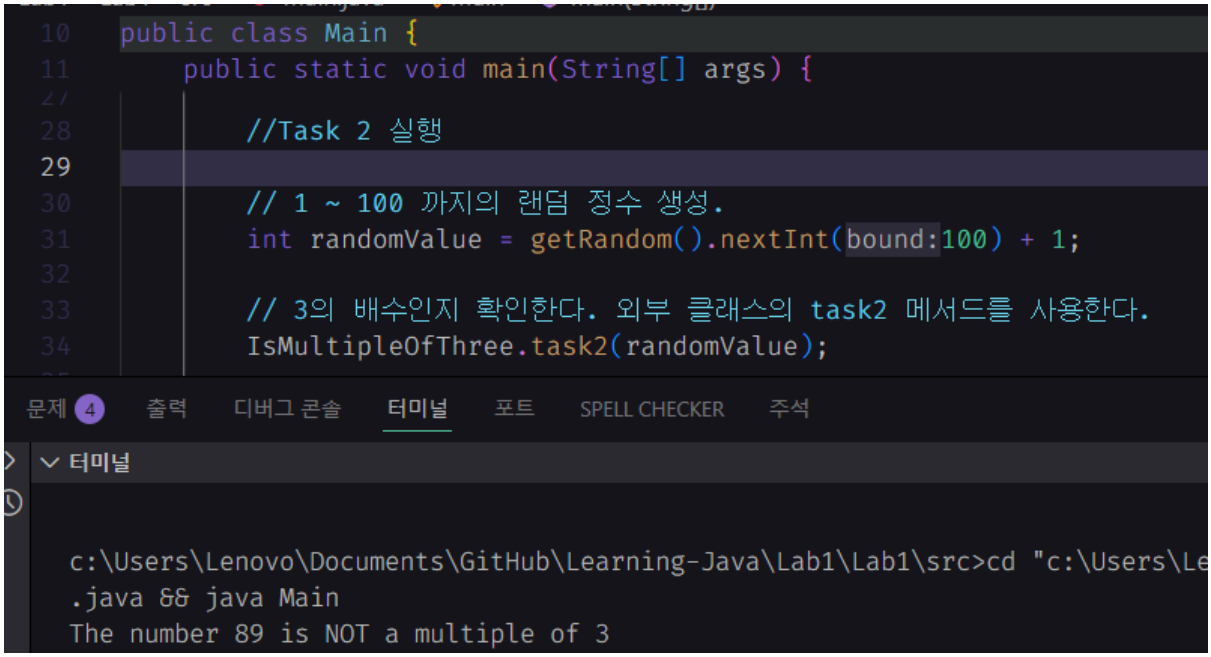


```java
public class IsMultipleOfThree {
    // task2 를 수행하는 함수.
    public static void task2 (int n) {
        //3의 배수인지 검사
        if(n % 3 == 0) {
            System.out.printf("The number %d is a multiple of 3\n", n);
        } else {
            System.out.printf("The number %d is NOT a multiple of 3\n", n);
        }
    }
}
```

다른 파일을 만들고 class 를 정의했습니다. Task2 를 수행하기 위해서 IsMultipleOfThree 클래스를 정의하고, task2 메서드를 만들었습니다. Static 키워드는 이 메서드를 Main 클래스의 main 함수에서 인스턴스를 생성하지않고 사용하기 하기 위해서 사용했습니다. 코드는 n 을 3으로 나눈 나머지를 통해서 배수인지 검사합니다.

```java
static Random getRandom( ) {
    //랜덤 객체 생성
    Random random = new Random( );

    //시드값을 현재시간으로 설정
    random.setSeed(System.currentTimeMillis( ));

    return random;
}
```

Main 클래스 내에 Random 객체를 반환하는 getRandom 메서드입니다. Random 값을 만들기 위해서 쓸 일이 많아서 호출 할 때 마다 객체를 만들고, 시드를 현재시간을 초기화 합니다.



위 getRandom 메서드를 통해서 랜덤 정수를 생성하고, task2 를 수행하는 함수를 호출합니다.

Task 3.

```java
public class FastExponentiation {
    // 지수를 분할정복으로 계산하는 함수
    public static int power(int base, int exponent, int mod) {
        // base case. 지수가 0이면 1을 반환한다.
        if (exponent == 0) return 1;

        // 분할정복. 지수를 반으로 나눈 값을 구하기 위해 재귀호출 한다.
        int half = power(base, exponent / 2, mod);

        // 만약 지수가 홀수면 추가적으로 곱해준다.
        if (exponent % 2 == 0) {
            return half * half % mod;
        } else {
            return half * half * base % mod;
        }
    }
}
```

Main 클래스 말고, 외부에 FastExponentiation 클래스를 작성하였습니다. 이 클래스는 분할정복으로 n제곱을 계산하여 빠르게 지수 계산을 돕는 클래스 입니다. 인스턴스 생성 없이 사용할 ㅅ ㅜ있도록 static 키워드로 power 메서드를 생성해줍니다. 인자로는 base(기수) exponent(지수) mod(나눌수) 를 받아서 base ^ exponent % mod 의 결과를 반환합니다.

중간에 자기 자신을 호출하는 재귀함수로, base case 는 지수가 0일 때 1을 반환하도록 함수 초기에 if문으로 조건분기를 하였습니다.

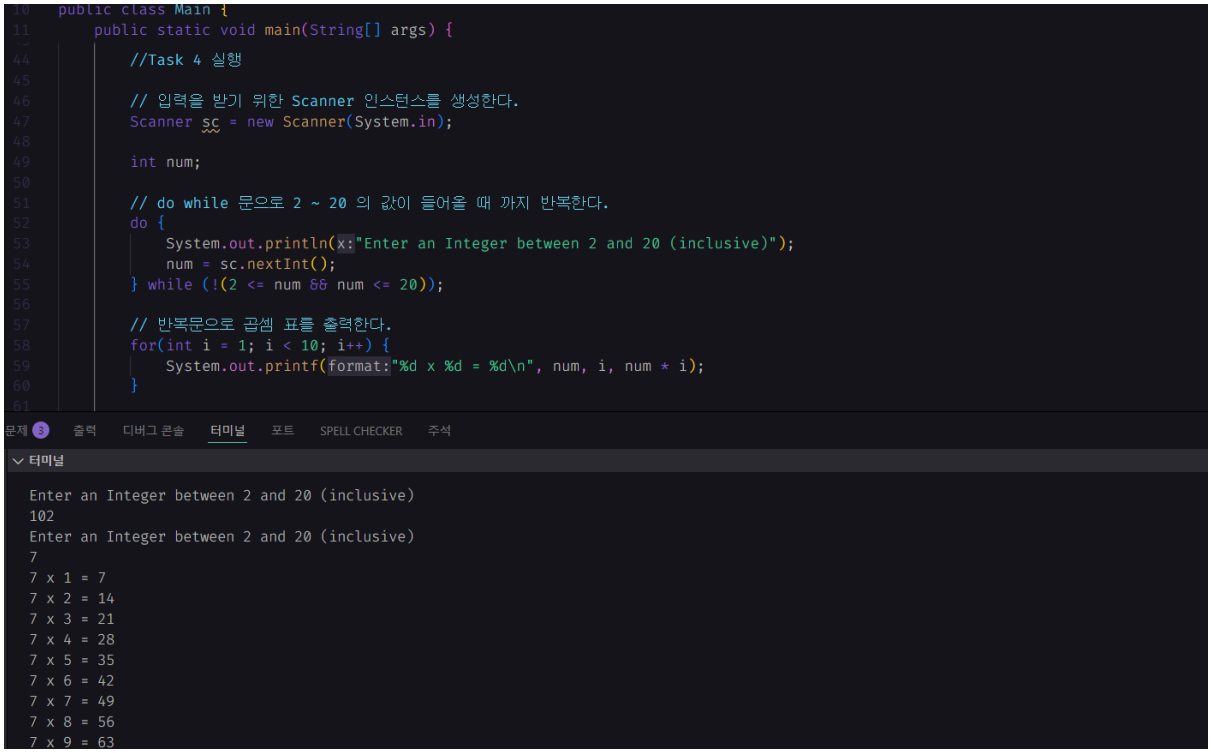
이후에는 지수를 반으로 줄인 재귀 호출을 합니다. 이후에는 재귀호출한 값을 가지고 반환합니다. 만약에 지수가 홀수라면 나눗셈 과정에서 하나가 계산되지 못 했을 테니 곱해준 다음 반환합니다.

```java
public class Main {
    public static void main(String[] args) {

        //Task 3 실행

        // 랜덤 값을 뽑는다.
        int base = getRandom().nextInt(bound:10) + 1;
        int exp = getRandom().nextInt(bound:6) + 1;
        int mod = 1000000007;
        System.out.printf(format:"base: %d exp: %d, mod: %d, result: %d\n",base,exp,mod,FastExponentiation.power(base, exp, mod));
```

```
문제 3  출력  디버그 콘솔  터미널  포트  SPELL CHECKER  주석
터미널

c:\Users\Lenovo\Documents\GitHub\Learning-Java\Lab1\Lab1\src>cd "c:\Users\Lenovo\Documents\GitHub\Learning-Java\Lab1\Lab1\src\" && javac Main
.java && java Main
3.2142857142857144
45 / 14
The number 32 is NOT a multiple of 3
base: 9 exp: 5, mod: 1000000007, result: 59049
Enter an Integer between 2 and 20 (inclusive)
```

랜덤값을 생성하고, power 를 구하는 task3 코드입니다.

Task 4.



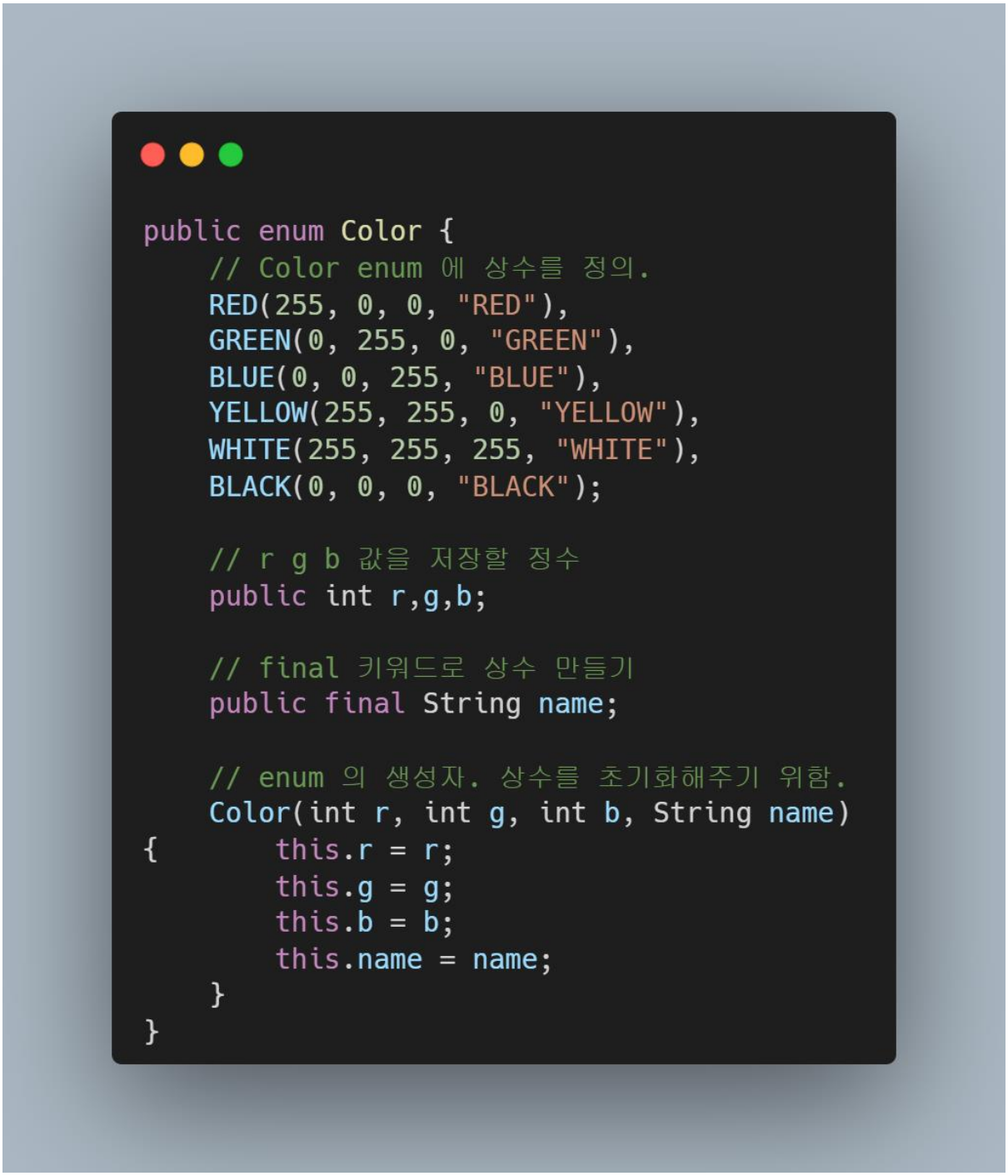
```
public class Main {
    public static void main(String[] args) {
        //Task 4 실행

        // 입력을 받기 위한 Scanner 인스턴스를 생성한다.
        Scanner sc = new Scanner(System.in);

        int num;

        // do while 문으로 2 ~ 20 의 값이 들어올 때 까지 반복한다.
        do {
            System.out.println(x:"Enter an Integer between 2 and 20 (inclusive)");
            num = sc.nextInt();
        } while (!(2 <= num && num <= 20));

        // 반복문으로 곱셈 표를 출력한다.
        for(int i = 1; i < 10; i++) {
            System.out.printf(format:"%d x %d = %d\n", num, i, num * i);
        }
```

```
문제 3  출력  디버그 콘솔  터미널  포트  SPELL CHECKER  주석
터미널

Enter an Integer between 2 and 20 (inclusive)
102
Enter an Integer between 2 and 20 (inclusive)
7
7 x 1 = 7
7 x 2 = 14
7 x 3 = 21
7 x 4 = 28
7 x 5 = 35
7 x 6 = 42
7 x 7 = 49
7 x 8 = 56
7 x 9 = 63
```

Java 의 표준 입출력을 받을 수 있도록 Scanner 클래스의 인스턴스 sc 를 생성합니다. New Scanner() 안에 System.in 을 인자로 주어 Scanner 객체를 생성하고 참조하는 변수로 sc 를 만듭니다.

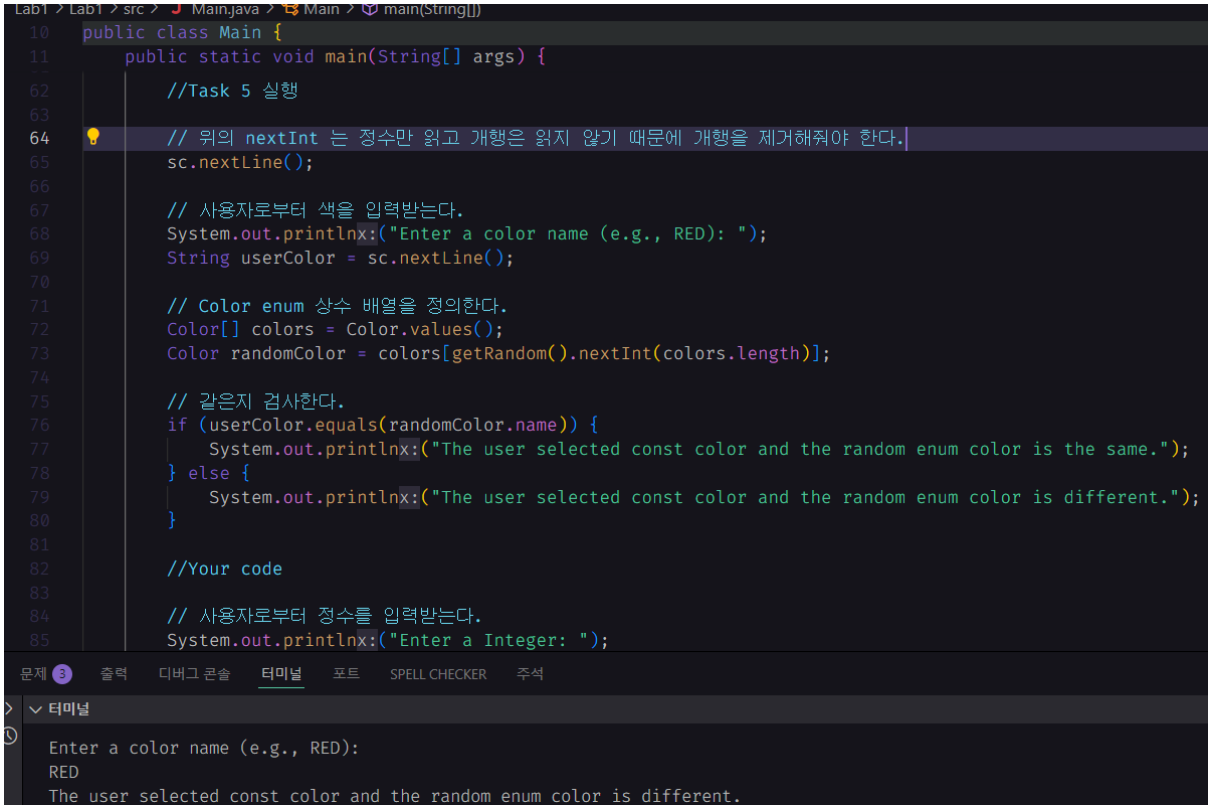
do while 문으로 입력을 받은 후, 입력값이 유효한지 검사를 하며 유효한 값을 받을 때 까지 반복합니다. 유효한 값이 들어왔다면 do while 문을 빠져나와서 곱셈표를 출력합니다.

Task 5.



```
public enum Color {
    // Color enum 에 상수를 정의.
    RED(255, 0, 0, "RED"),
    GREEN(0, 255, 0, "GREEN"),
    BLUE(0, 0, 255, "BLUE"),
    YELLOW(255, 255, 0, "YELLOW"),
    WHITE(255, 255, 255, "WHITE"),
    BLACK(0, 0, 0, "BLACK");

    // r g b 값을 저장할 정수
    public int r,g,b;

    // final 키워드로 상수 만들기
    public final String name;

    // enum 의 생성자. 상수를 초기화해주기 위함.
    Color(int r, int g, int b, String name)
{       this.r = r;
        this.g = g;
        this.b = b;
        this.name = name;
    }
}
```

Color 이라는 이름의 enum을 정의했습니다.이 열거형에는 RED GREEN등 색상에 관련된 상수들을 멤버로 포함하고 있습니다. 각 상수들은 정수형 r g b 값과 문자열 상수의 이름을 갖습니다. Color(int r, int g, int b, String name)은 enum 의 생성자로서 각 상수의 속성을 초기화하는 역할을 합니다.



Main 클래스에 Task5 실행 코드를 작성하였습니다. 위에서 nextInt 로 정수를 입력 받은 후 nextLine 을 하면 개행문자가 남아있기 때문에 개행문자를 제거하기 위해서 nextLine 을 한 번 호출하였습니다. 이후에 사용자에게 색깔을 입력 받고, .values 메서드로 Color 열거형의 상수들이 있는 배열을 만들었습니다. 랜덤한 값을 선택하기 위해서 0 ~ colors 배열의 길이 중 랜덤 값을 선택하여 colors 인덱스에 접근하여 randomColor 을 선택하였습니다. 이후에 .equals 메서드를 사용하고, randomColor 로 선택된 enum 의 .name 속성을 비교하여 같은 값인지 검사를 합니다.

Task Your Code.

Your code 실행 코드입니다. isPrime 이라는 소수 판정 메서드를 정의합니다. 2 미만이면 false 를 반환하고, n 의 약수들을 나열했을 때, sqrt(n) 부터는 대칭적인 형태가 되므로, sqrt(n) 까지만 검사합니다. 2 부터 sqrt(n) 까지 검사를 하면서, 나누어지는 값이 있는지 봅니다. 하나라도 나누어지는 값이 있으면 소수가 아니므로 false 를 반환하고, 반복문의 검사를 모두 통과하고나면 true 를 반환합니다.

이 메서드를 활용하여 사용자로부터 입력을 받고 소수인지 아닌지를 출력합니다.