# PrimeNumber Part:A Mastery ErrorLog

The first rendition of my code involved a simple if else statement. Unfortunately this would result in the program declaring the number as prime if the number wasn't evenly divisible by the first divisor (2). Any odd number would result in it being declared prime.

```
package Mastery;

import java.util.Scanner;

public class PrimeNumberPartA {

    public static void main(String[] args) {

        //Preparing for user input
        Scanner userInput = new Scanner(System.in);

        //Prompt and record the user for
        System.out.print("Enter an integer value: ");
        int number = userInput.nextInt();


        for (int count=2; count <= (number - 1); count++) {
            if (number % count == 0) {
                System.out.print("Number isn't prime");
                break;
            } else {
                System.out.print("Number is prime");
                break;
            }
            }
    }

}
```

Example of said error:

```
Enter an integer value: 15
Number is prime
```

To remedy this I created a boolean expression that would be false by default. This would represent if the number was prime or not. I then removed the break statement from the else so that the loop would continue even if the first divisor wouldn't work.

```
//Preparing for user input
Scanner userInput = new Scanner(System.in);

//Prompt and record the user for
System.out.print("Enter an integer value: ");
int number = userInput.nextInt();

boolean prime = false;

for (int count=2; count <= (number - 1); count++) {
    if (number % count == 0) {
        prime = false;
        break;
    } else {
        prime = true;
    }
}

if (prime == false) {
    System.out.print("Number isn't prime");
} else if (prime == true) {
    System.out.print("Number is prime");
}
```

This correctly identified odd numbers as non-prime, but was inefficient and returned 2 as non prime. This was due to the fact that prime was false by default. I ended up declaring prime as true by default, and then added an or statement to the final if else so that 1 would return as non-prime.

This is the final rendition of my code which is further optimized, and accounts for 1 / odd numbers.

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

    //Preparing for user input
    Scanner userInput = new Scanner(System.in);

    //Prompt and record the user for
    System.out.print("Enter an integer value: ");
    int number = userInput.nextInt();

    //Declaration
    boolean prime = true;

    //Loops an if else statement and increments count by 1 while the
    //count is less than or equal to 2 less than the inputed number
    for (int count=2; count <= (number - 2); count++) {
        //Checks if the modulo of the number is zero
        if (number % count == 0) {
            //Declares the number as not prime and breaks loop
            prime = false;
            break;
        }
    }

    //Checks if the number is prime or is 1
    if (prime == false || number == 1) {
        //Prints according statement
        System.out.print("Number isn't prime");
    } else {
        System.out.print("Number is prime");
    }
```

Example output:

```
Enter an integer value: 15
Number isn't prime
```