Prime Number Program in Java

Introduction

Any beginner-level coding class includes learning to do the prime number program in Java. Despite its simplicity, students may initially find it confusing with the various programs involved in the process. This comprehensive guide explains with examples how to run the prime number program in Java using a variety of processes.

Overview

Any natural number that is greater than 1 and divisible by 1 and itself only is termed a prime number. The ***isPrime()*** function is usually used in Java to determine whether the input number is prime or not.

What are Prime Numbers?

All natural numbers can be categorized into 2 classes _ prime numbers and composite numbers.

A prime number is defined as a natural number greater than 1 which is divisible by 1 and the number itself only. It has only 2 divisors. Some examples of prime numbers are 2,3,5,7,13,47,53…
On the other hand, composite or non-prime numbers are divisible by more than 2 numbers apart from 1 and itself.

Some interesting facts about Prime Numbers

- 0 and 1 are neither prime nor composite numbers.
- The only consecutive natural prime numbers are 2 and 3.
- No prime number greater than 5 ends in 5. This is because all numbers ending with 5 are divisible by 5 besides the number itself.
- **Twin prime numbers** are consecutive primes that come with a composite number between them. Their difference is always. Example - (5,7), (11,13), (17,19), (29,3,), (41,43), (59,61), (71,73).
- 17 and 71 are called **twisted prime numbers** as both are prime and mirror images of each other. Other examples - (13,31), (37,73), (79,97).

Properties of Prime Numbers

Prime number properties

- All numbers greater than 1 can be divided by at least one prime number.
- 2 is the smallest as well as the only even prime number. All other prime numbers are odd.
- All prime numbers except 2 and 3 can be represented in the form of either 6n+1 or 6n-1. (n= any natural number)
- Any given positive integer that is greater than 2 can be written as a sum of two prime integers.
- Two prime numbers are always coprimes of one another.
- Any given composite number can be divided into prime factors, which individually are unique.

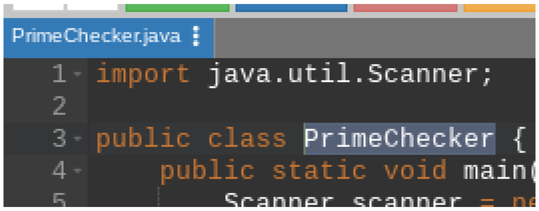
Prime Numbers and Co-prime Numbers

Prime and co-prime numbers are distinctly different.

Prime Numbers	Co-prime Numbers
Single number	Always come in pairs.
Only factors are 1 and itself.	Highest common factor is always 1.
Only primes.	Can be prime or composite.
Examples - 17,23,67	Examples- (17,25), (6,13), (8,15)

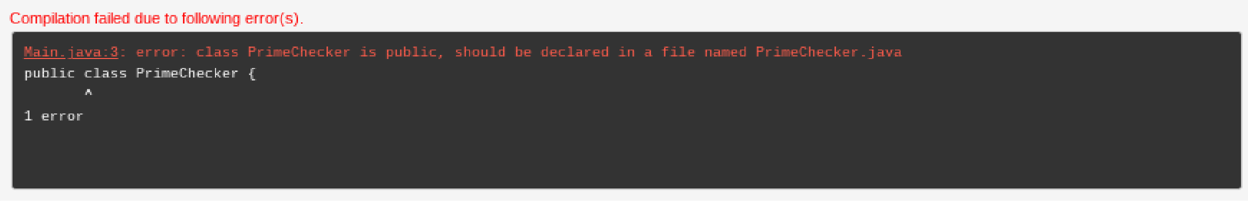
How to check whether a number is Prime or not?

We will be using the ‘isPrime’ method for finding out if a number is prime or not in the following program. However, in this method, you must name the .java file where you are writing the code as ‘***PrimeCheker.java’*** or it will cause an error during compilation.



```
import java.util.Scanner;

public class PrimeChecker {
    public static void main(
        Scanner scanner = ne
```

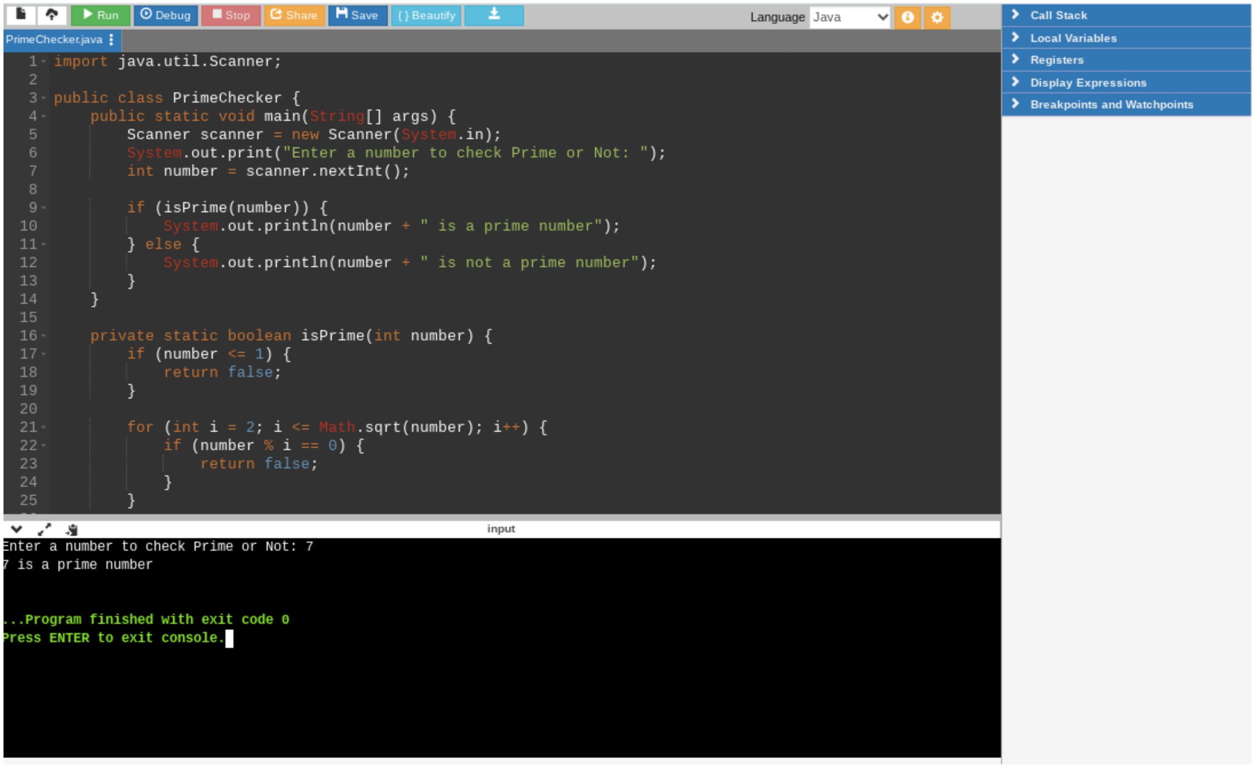
If the ‘***PrimeChecker’*** class is not declared in a PrimeChecker.java file, it will cause the following error after execution:



Compilation failed due to following error(s).

```
Main.java:3: error: class PrimeChecker is public, should be declared in a file named PrimeChecker.java
public class PrimeChecker {
       ^
1 error
```

Here is the program:



```
import java.util.Scanner;

public class PrimeChecker {
    public static void main(String[] args) {
        Scanner scanner = new Scanner(System.in);
        System.out.print("Enter a number to check Prime or Not: ");
        int number = scanner.nextInt();

        if (isPrime(number)) {
            System.out.println(number + " is a prime number");
        } else {
            System.out.println(number + " is not a prime number");
        }
    }

    private static boolean isPrime(int number) {
        if (number <= 1) {
            return false;
        }

        for (int i = 2; i <= Math.sqrt(number); i++) {
            if (number % i == 0) {
                return false;
            }
        }
```

```
Enter a number to check Prime or Not: 7
7 is a prime number


...Program finished with exit code 0
Press ENTER to exit console.
```

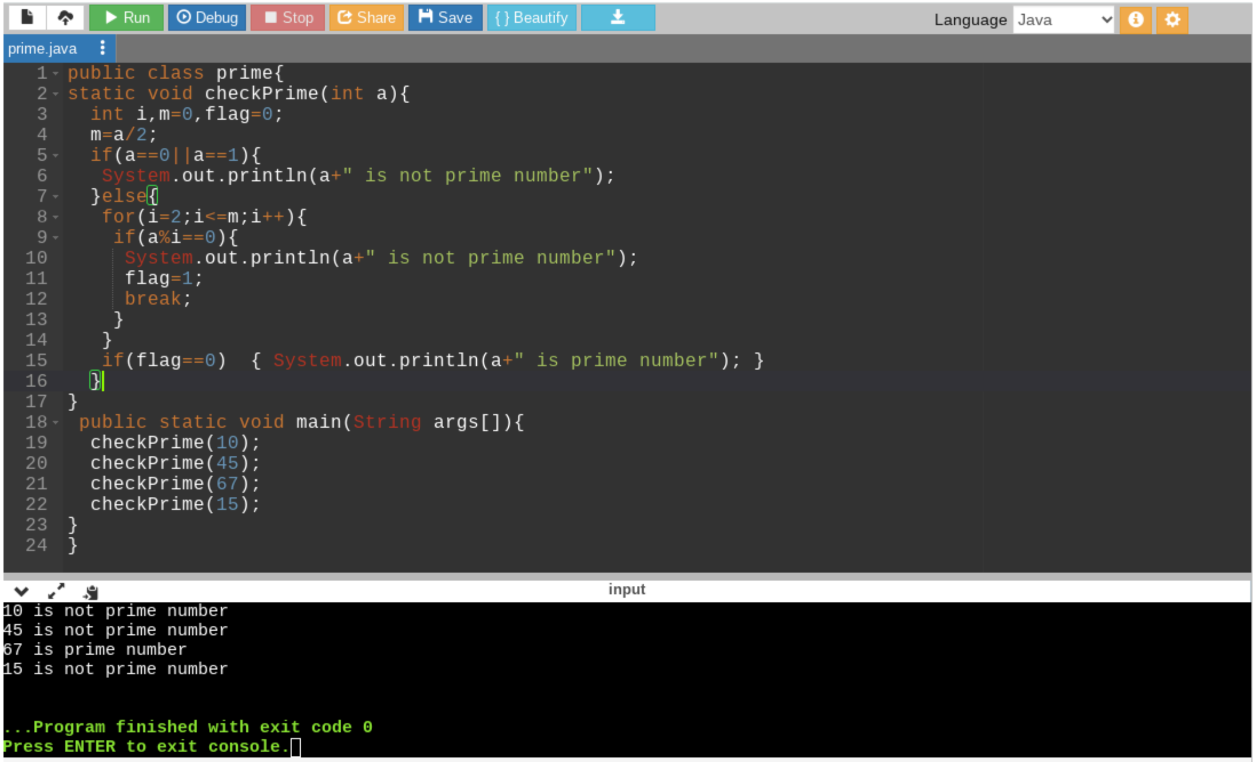
```
import java.util.Scanner;

public class PrimeChecker {
    public static void main(String[] args) {
        Scanner scanner = new Scanner(System.in);
        System.out.print("Enter a number to check Prime or Not: ");
        int number = scanner.nextInt();

        if (isPrime(number)) {
            System.out.println(number + " is a prime number");
        } else {
            System.out.println(number + " is not a prime number");
        }
    }
    private static boolean isPrime(int number) {
        if (number <= 1) {
            return false;
        }

        for (int i = 2; i <= Math.sqrt(number); i++) {
            if (number % i == 0) {
                return false;
            }
        }

        return true;
    }
}
```

The prime number program in javascript begins with importing the Scanner class from java.util package using ***“import java.util.Scanner;”***. Next, an instance of the Scanner class is created using ***“Scanner scanner = new Scanner(System.in);”*** which allows users to input a number. This is followed by the program prompt ***“System.out.print("Enter a number to check Prime or Not: ");”***.

The program reads the user’s input using the ***“scanner.nextInt()”*** method. The entered number’s value is stored in an integer variable termed “number” using “int number = scanner.nextInt();”. The isPrime() method with the entered value is called an “argument” which checks whether the number is prime or not.

If the java program to print prime numbers in a given range returns true with the statement flashing ***“System.out.println(number + " is a prime number");”*** then the entered value is prime. If not, the ***isPrime() method*** returns false with the message ***“System.out.println(number + " is not a prime number");”***. This indicates the entered value is not prime.

Prime Number Program using checkPrime Method in Java



```
public class prime{
static void checkPrime(int a){
  int i,m=0,flag=0;
  m=a/2;
  if(a==0||a==1){
   System.out.println(a+" is not prime number");
  }else{
   for(i=2;i<=m;i++){
    if(a%i==0){
     System.out.println(a+" is not prime number");
     flag=1;
     break;
    }
   }
   if(flag==0)  { System.out.println(a+" is prime number"); }
  }
}
 public static void main(String args[]){
  checkPrime(10);
  checkPrime(45);
  checkPrime(67);
  checkPrime(15);
}
}
```

```
10 is not prime number
45 is not prime number
67 is prime number
15 is not prime number


...Program finished with exit code 0
Press ENTER to exit console.
```

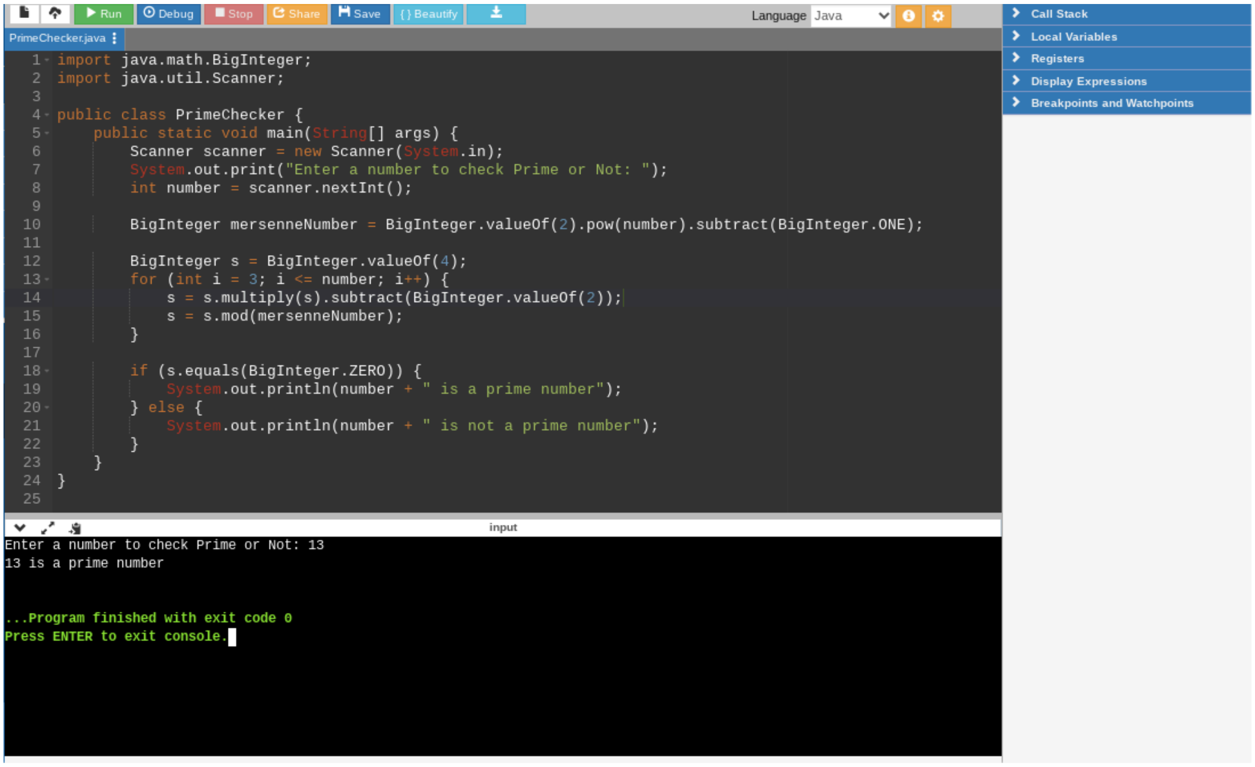
```
//Prime Number Program using Method in Java
public class prime{
static void checkPrime(int a){
  int i,m=0,flag=0;
  m=a/2;
  if(a==0||a==1){
   System.out.println(a+" is not prime number");
  }else{
   for(i=2;i<=m;i++){
    if(a%i==0){
     System.out.println(a+" is not prime number");
     flag=1;
     break;
    }
   }
   if(flag==0)  { System.out.println(a+" is prime number"); }
  }//end of else
}
 public static void main(String args[]){
  checkPrime(10);
  checkPrime(45);
  checkPrime(67);
  checkPrime(15);
}
}
```

Another way to write a program to check if a given number is prime or not is by using the "***checkPrime***" method. The checkPrime method takes an integer "***a***" as input and declares its variables "***i***," "***m***," and "***fl***ag" to 0. The value of "m" is set to "a/2," and the program prints ***"a is not a prime number"*** when the variable is equal to 0 or 1. The program then enters a "***for***" loop that initializes "***i***" to 2 and continues until ***"i"*** is less than or equal to ***"m."***

Among the given inputs, only 67 returns as prime.

Prime Number Program in Java (Another way)

In the following program, we will be using the Lucas-Lehmer test for Mersenne primes to find out if a number is prime or not.

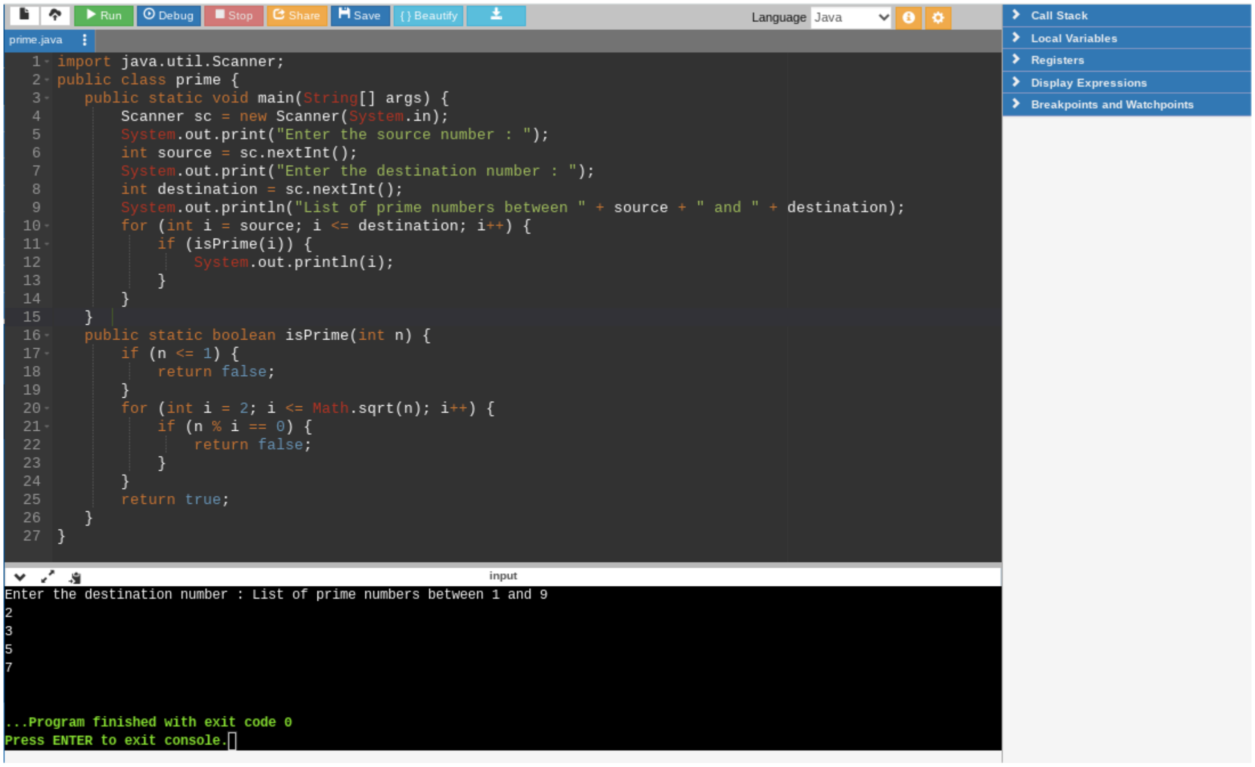


```
import java.math.BigInteger;
import java.util.Scanner;

public class PrimeChecker {
    public static void main(String[] args) {
        Scanner scanner = new Scanner(System.in);
        System.out.print("Enter a number to check Prime or Not: ");
        int number = scanner.nextInt();

        BigInteger mersenneNumber = BigInteger.valueOf(2).pow(number).subtract(BigInteger.ONE);

        BigInteger s = BigInteger.valueOf(4);
        for (int i = 3; i <= number; i++) {
            s = s.multiply(s).subtract(BigInteger.valueOf(2));
            s = s.mod(mersenneNumber);
        }

        if (s.equals(BigInteger.ZERO)) {
            System.out.println(number + " is a prime number");
        } else {
            System.out.println(number + " is not a prime number");
        }
    }
}
```

```
Enter a number to check Prime or Not: 13
13 is a prime number


...Program finished with exit code 0
Press ENTER to exit console.
```

```
import java.math.BigInteger;
import java.util.Scanner;

public class PrimeChecker {
    public static void main(String[] args) {
        Scanner scanner = new Scanner(System.in);
        System.out.print("Enter a number to check Prime or Not: ");
        int number = scanner.nextInt();

        BigInteger mersenneNumber =
BigInteger.valueOf(2).pow(number).subtract(BigInteger.ONE);

        BigInteger s = BigInteger.valueOf(4);
        for (int i = 3; i <= number; i++) {
            s = s.multiply(s).subtract(BigInteger.valueOf(2));
            s = s.mod(mersenneNumber);
        }

        if (s.equals(BigInteger.ZERO)) {
            System.out.println(number + " is a prime number");
        } else {
            System.out.println(number + " is not a prime number");
        }
    }
}
```

Find prime numbers between two numbers



```
import java.util.Scanner;
public class prime {
    public static void main(String[] args) {
        Scanner sc = new Scanner(System.in);
        System.out.print("Enter the source number : ");
        int source = sc.nextInt();
        System.out.print("Enter the destination number : ");
        int destination = sc.nextInt();
        System.out.println("List of prime numbers between " + source + " and " + destination);
        for (int i = source; i <= destination; i++) {
            if (isPrime(i)) {
                System.out.println(i);
            }
        }
    }
    public static boolean isPrime(int n) {
        if (n <= 1) {
            return false;
        }
        for (int i = 2; i <= Math.sqrt(n); i++) {
            if (n % i == 0) {
                return false;
            }
        }
        return true;
    }
}
```

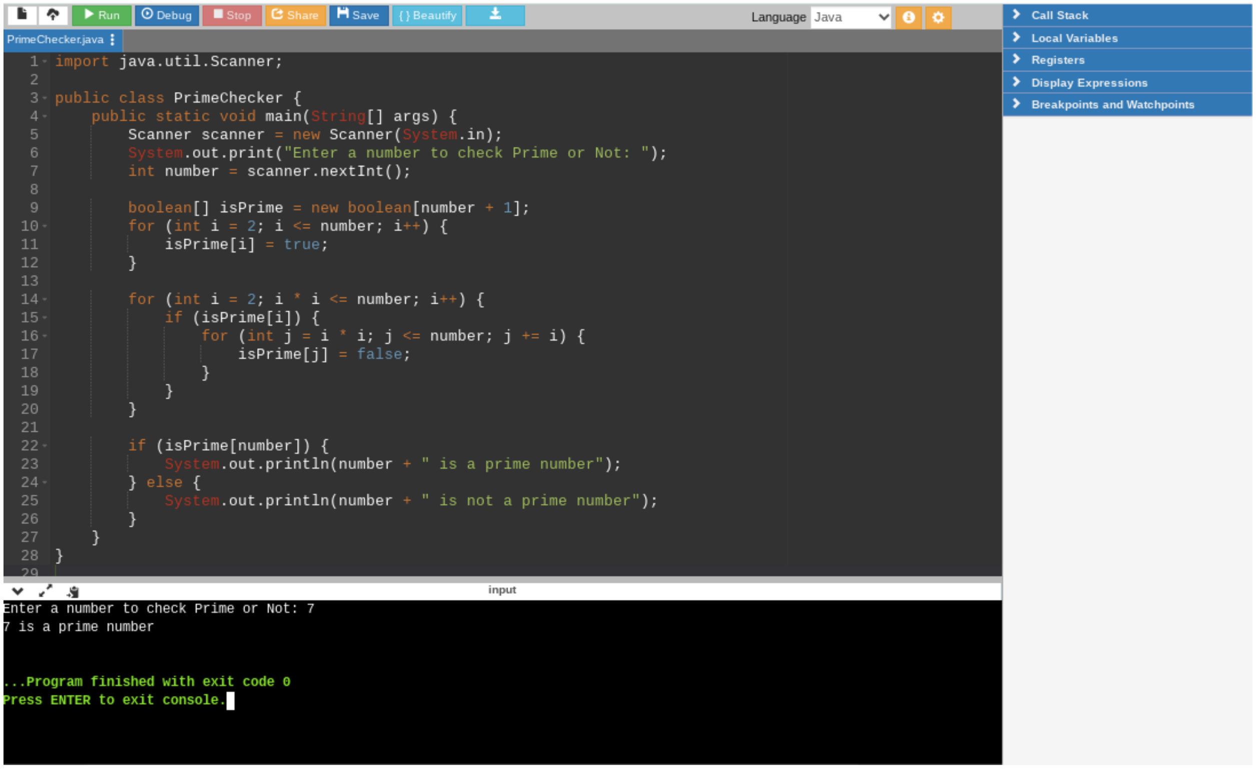
```
Enter the destination number : List of prime numbers between 1 and 9
2
3
5
7


...Program finished with exit code 0
Press ENTER to exit console.
```

//Find prime numbers between two numbers

```
import java.util.Scanner;
public class prime {
    public static void main(String[] args) {
        Scanner sc = new Scanner(System.in);
        System.out.print("Enter the source number : ");
        int source = sc.nextInt();
        System.out.print("Enter the destination number : ");
        int destination = sc.nextInt();
        System.out.println("List of prime numbers between " + source + " and " + destination);
        for (int i = source; i <= destination; i++) {
            if (isPrime(i)) {
                System.out.println(i);
            }
        }
    }
    public static boolean isPrime(int n) {
        if (n <= 1) {
            return false;
        }
        for (int i = 2; i <= Math.sqrt(n); i++) {
            if (n % i == 0) {
                return false;
            }
        }
        return true;
    }
}
```

How We Check Whether a Number is Prime or Not?

In the following program, we will use the Sieve of Eratosthenes method to determine if a number is prime.

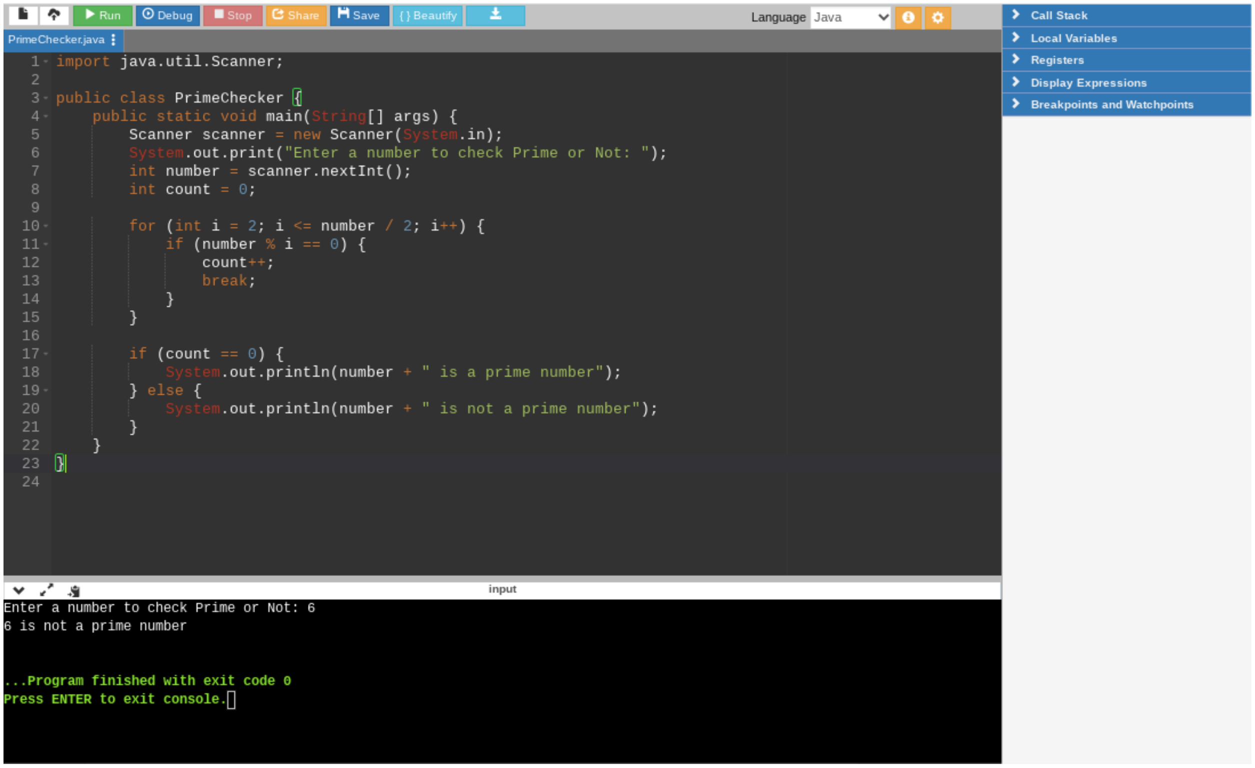


```java
import java.util.Scanner;

public class PrimeChecker {
    public static void main(String[] args) {
        Scanner scanner = new Scanner(System.in);
        System.out.print("Enter a number to check Prime or Not: ");
        int number = scanner.nextInt();

        boolean[] isPrime = new boolean[number + 1];
        for (int i = 2; i <= number; i++) {
            isPrime[i] = true;
        }

        for (int i = 2; i * i <= number; i++) {
            if (isPrime[i]) {
                for (int j = i * i; j <= number; j += i) {
                    isPrime[j] = false;
                }
            }
        }

        if (isPrime[number]) {
            System.out.println(number + " is a prime number");
        } else {
            System.out.println(number + " is not a prime number");
        }
    }
}
```

```
Enter a number to check Prime or Not: 7
7 is a prime number


...Program finished with exit code 0
Press ENTER to exit console.
```

```java
import java.util.Scanner;

public class PrimeChecker {
    public static void main(String[] args) {
        Scanner scanner = new Scanner(System.in);
        System.out.print("Enter a number to check Prime or Not: ");
        int number = scanner.nextInt();

        boolean[] isPrime = new boolean[number + 1];
        for (int i = 2; i <= number; i++) {
            isPrime[i] = true;
        }

        for (int i = 2; i * i <= number; i++) {
            if (isPrime[i]) {
                for (int j = i * i; j <= number; j += i) {
                    isPrime[j] = false;
                }
            }
        }

        if (isPrime[number]) {
            System.out.println(number + " is a prime number");
        } else {
            System.out.println(number + " is not a prime number");
        }
    }
}
```

Program to Check Prime Number Using a For Loop

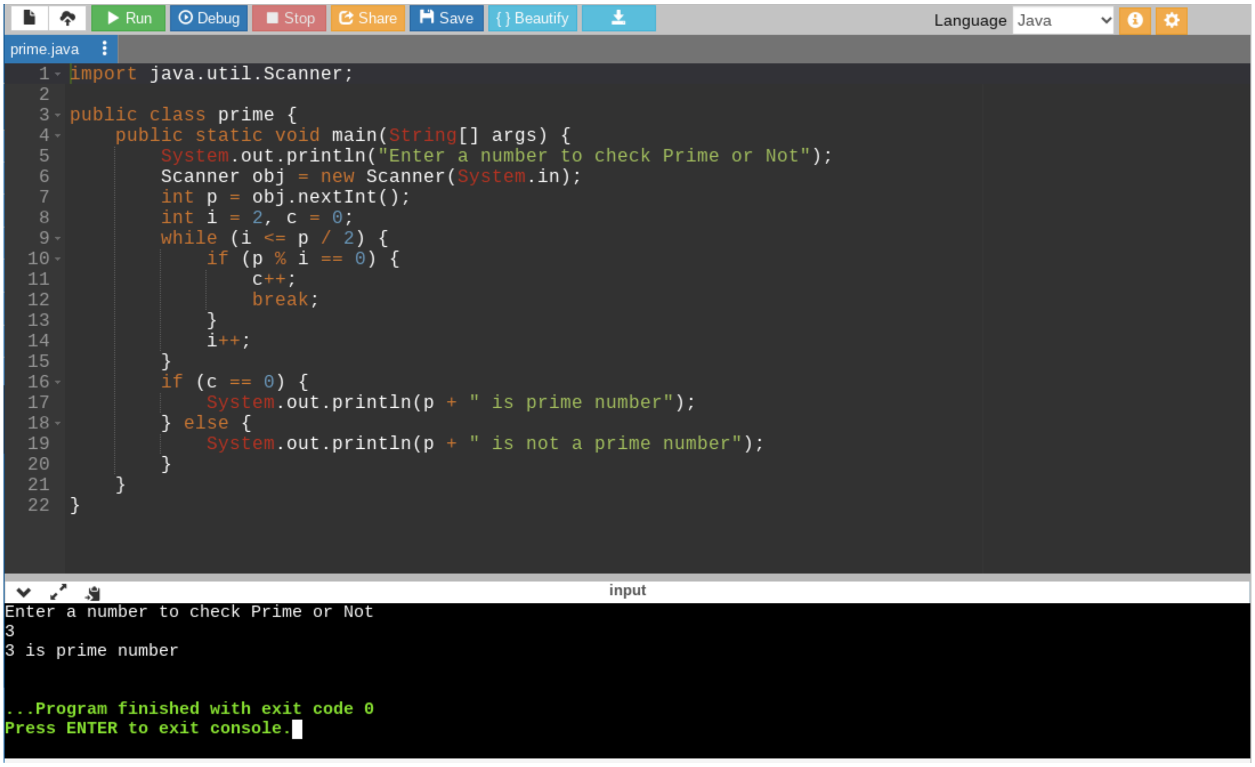


```
import java.util.Scanner;

public class PrimeChecker {
    public static void main(String[] args) {
        Scanner scanner = new Scanner(System.in);
        System.out.print("Enter a number to check Prime or Not: ");
        int number = scanner.nextInt();
        int count = 0;

        for (int i = 2; i <= number / 2; i++) {
            if (number % i == 0) {
                count++;
                break;
            }
        }

        if (count == 0) {
            System.out.println(number + " is a prime number");
        } else {
            System.out.println(number + " is not a prime number");
        }
    }
}
```

```
Enter a number to check Prime or Not: 6
6 is not a prime number


...Program finished with exit code 0
Press ENTER to exit console.
```

```
import java.util.Scanner;

public class PrimeChecker {
    public static void main(String[] args) {
        Scanner scanner = new Scanner(System.in);
        System.out.print("Enter a number to check Prime or Not: ");
        int number = scanner.nextInt();
        int count = 0;

        for (int i = 2; i <= number / 2; i++) {
            if (number % i == 0) {
                count++;
                break;
            }
        }

        if (count == 0) {
            System.out.println(number + " is a prime number");
        } else {
            System.out.println(number + " is not a prime number");
        }
    }
}
```

Program to Check Prime Number Using a While Loop

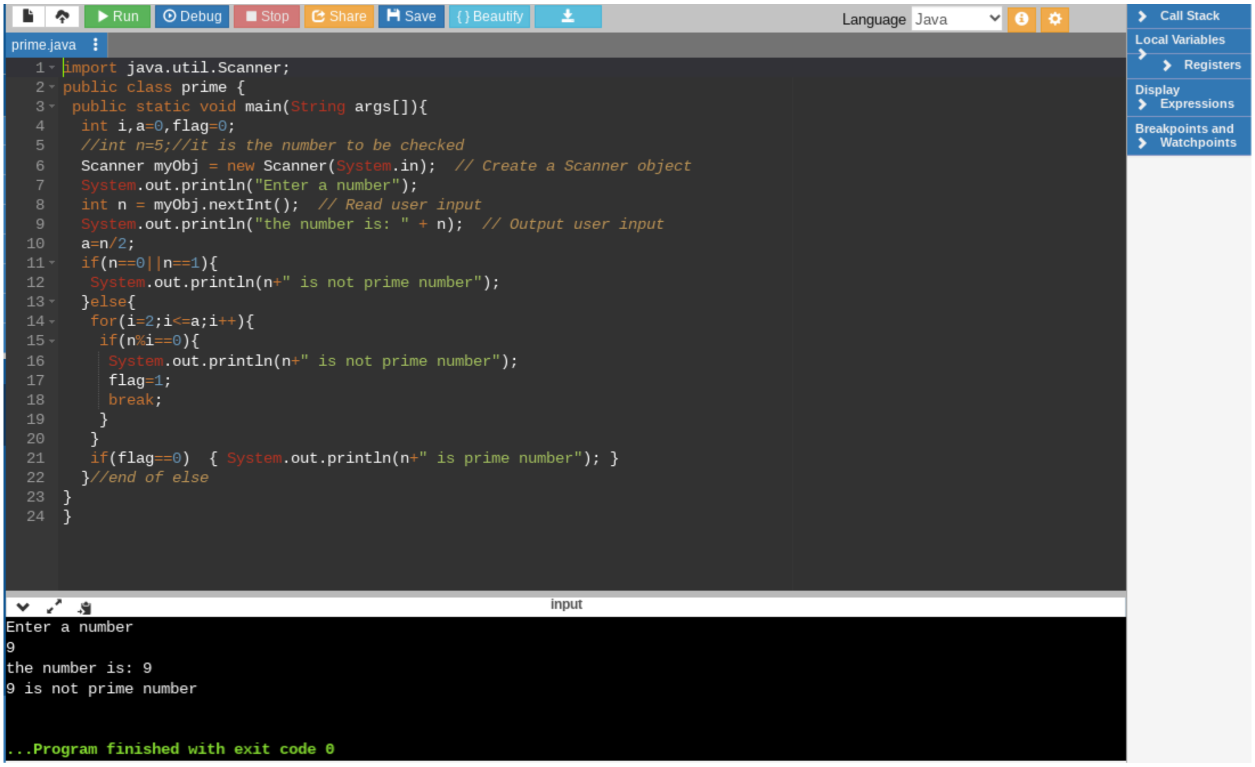


```
import java.util.Scanner;

public class prime {
    public static void main(String[] args) {
        System.out.println("Enter a number to check Prime or Not");
        Scanner obj = new Scanner(System.in);
        int p = obj.nextInt();
        int i = 2, c = 0;
        while (i <= p / 2) {
            if (p % i == 0) {
                c++;
                break;
            }
            i++;
        }
        if (c == 0) {
            System.out.println(p + " is prime number");
        } else {
            System.out.println(p + " is not a prime number");
        }
    }
}
```

```
Enter a number to check Prime or Not
3
3 is prime number


...Program finished with exit code 0
Press ENTER to exit console.
```

```
//Program to Check Prime Number Using a While Loop
import java.util.Scanner;

public class prime {
    public static void main(String[] args) {
        System.out.println("Enter a number to check Prime or Not");
        Scanner obj = new Scanner(System.in);
        int p = obj.nextInt();
        int i = 2, c = 0;
        while (i <= p / 2) {
            if (p % i == 0) {
                c++;
                break;
            }
            i++;
        }
        if (c == 0) {
            System.out.println(p + " is prime number");
        } else {
            System.out.println(p + " is not a prime number");
        }
    }
}
```

Program to Check If the Number is Prime or not using a Flag Variable



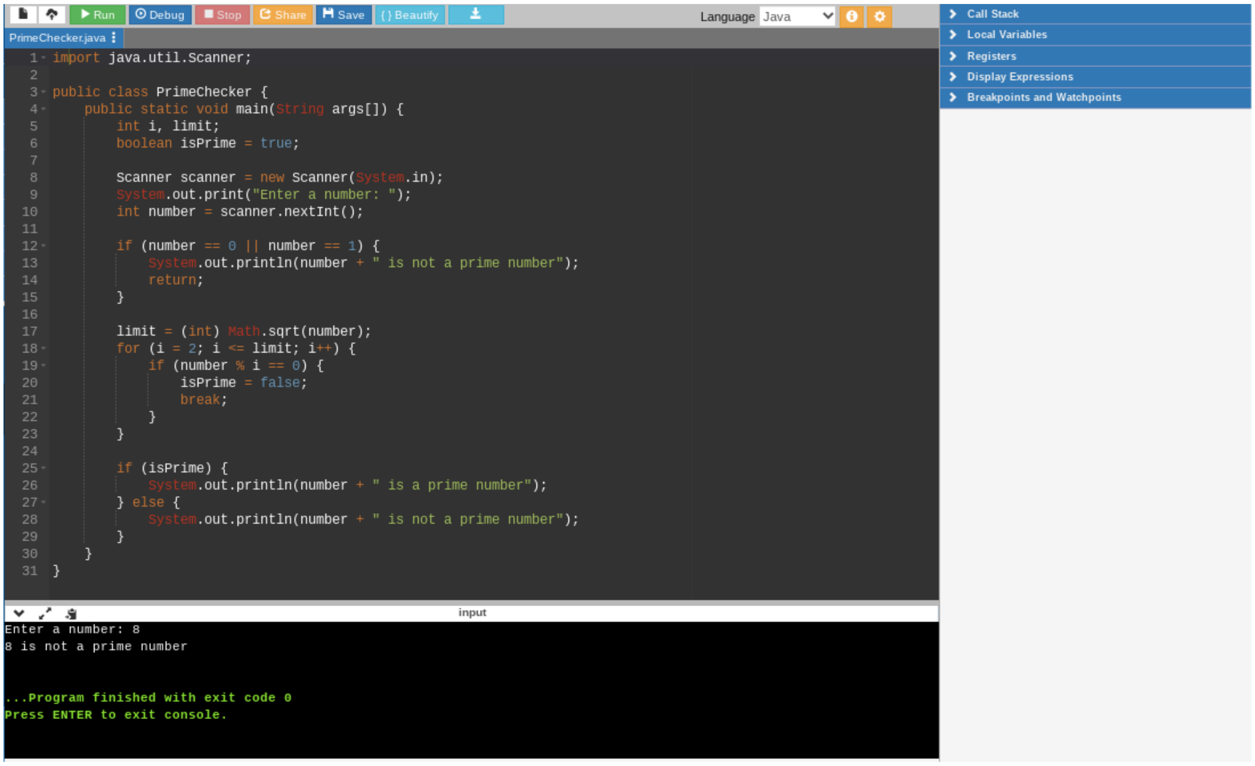
```java
import java.util.Scanner;
public class prime {
 public static void main(String args[]){
  int i,a=0,flag=0;
  //int n=5;//it is the number to be checked
  Scanner myObj = new Scanner(System.in);  // Create a Scanner object
  System.out.println("Enter a number");
  int n = myObj.nextInt();  // Read user input
  System.out.println("the number is: " + n);  // Output user input
  a=n/2;
  if(n==0||n==1){
   System.out.println(n+" is not prime number");
  }else{
   for(i=2;i<=a;i++){
    if(n%i==0){
     System.out.println(n+" is not prime number");
     flag=1;
     break;
    }
   }
   if(flag==0)  { System.out.println(n+" is prime number"); }
  }//end of else
}
}
```

```
Enter a number
9
the number is: 9
9 is not prime number


...Program finished with exit code 0
```

```
//Program to Check If the Number is Prime or not using a Flag Variable
import java.util.Scanner;
public class prime {
 public static void main(String args[]){
  int i,a=0,flag=0;
  //int n=5;//it is the number to be checked
  Scanner myObj = new Scanner(System.in);  // Create a Scanner object
  System.out.println("Enter a number");
  int n = myObj.nextInt();  // Read user input
  System.out.println("the number is: " + n);  // Output user input
  a=n/2;
  if(n==0||n==1){
   System.out.println(n+" is not prime number");
  }else{
   for(i=2;i<=a;i++){
    if(n%i==0){
     System.out.println(n+" is not prime number");
     flag=1;
     break;
    }
   }
   if(flag==0)  { System.out.println(n+" is prime number"); }
  }//end of else
}
}
```

Further Optimization

Let us now further optimize the above code by using a boolean variable ***‘isPrime’*** rather than a flag variable. The boolean variable will help us determine if the number is a prime number or not, based on the value of ***‘isPrime’***.

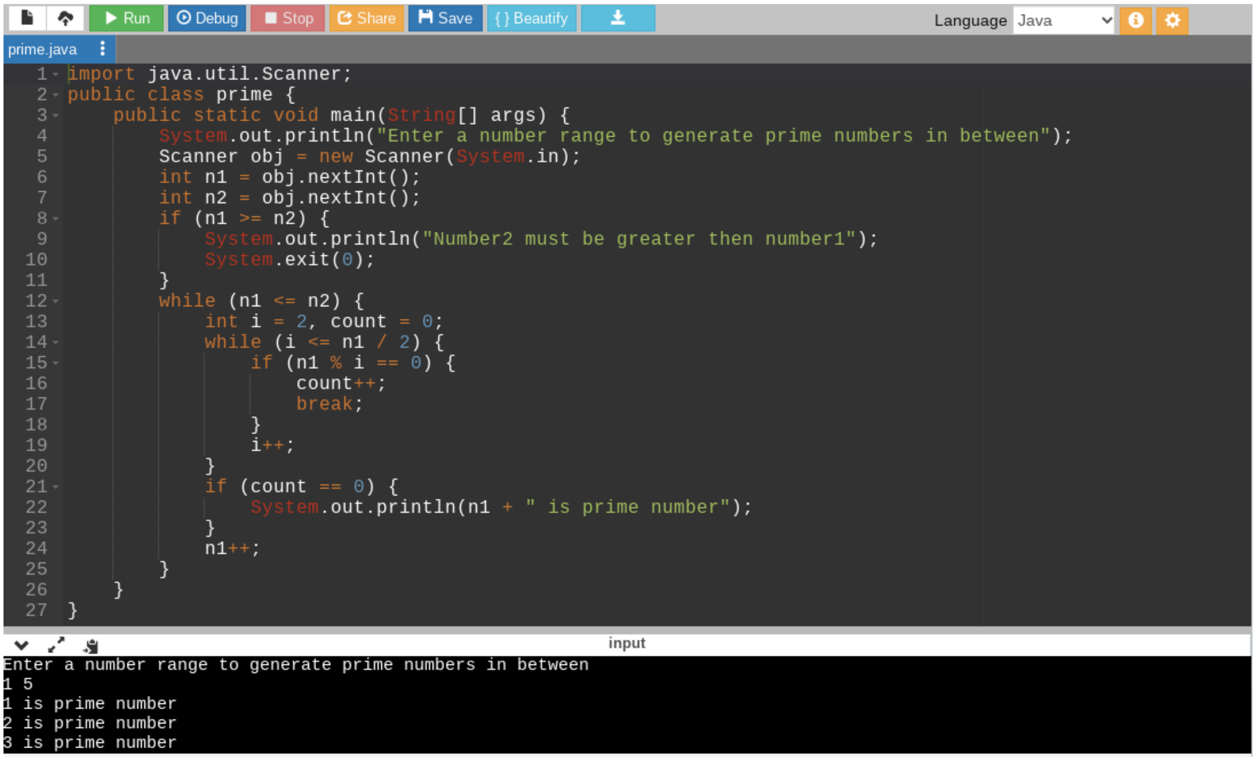


```
import java.util.Scanner;

public class PrimeChecker {
    public static void main(String args[]) {
        int i, limit;
        boolean isPrime = true;

        Scanner scanner = new Scanner(System.in);
        System.out.print("Enter a number: ");
        int number = scanner.nextInt();

        if (number == 0 || number == 1) {
            System.out.println(number + " is not a prime number");
            return;
        }

        limit = (int) Math.sqrt(number);
        for (i = 2; i <= limit; i++) {
            if (number % i == 0) {
                isPrime = false;
                break;
            }
        }

        if (isPrime) {
            System.out.println(number + " is a prime number");
        } else {
            System.out.println(number + " is not a prime number");
        }
    }
}
```

```
Enter a number: 8
8 is not a prime number


...Program finished with exit code 0
Press ENTER to exit console.
```

```
import java.util.Scanner;

public class PrimeChecker {
    public static void main(String args[]) {
        int i, limit;
        boolean isPrime = true;

        Scanner scanner = new Scanner(System.in);
        System.out.print("Enter a number: ");
        int number = scanner.nextInt();

        if (number == 0 || number == 1) {
            System.out.println(number + " is not a prime number");
            return;
        }

        limit = (int) Math.sqrt(number);
        for (i = 2; i <= limit; i++) {
            if (number % i == 0) {
                isPrime = false;
                break;
            }
        }
        if (isPrime) {
            System.out.println(number + " is a prime number");
        } else {
            System.out.println(number + " is not a prime number");
        }
    }
}
```

Program to Display the prime Numbers From 1 to 100



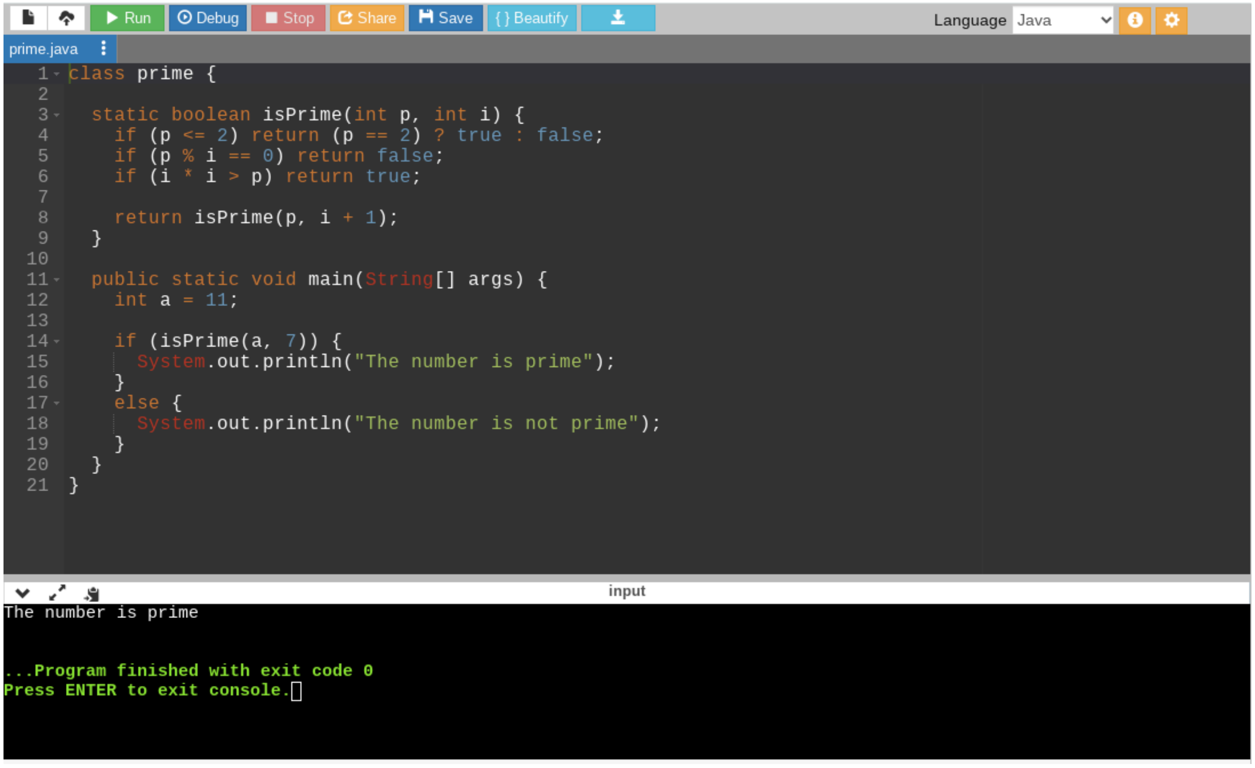
```
import java.util.Scanner;
public class prime {
    public static void main(String[] args) {
        System.out.println("Enter a number range to generate prime numbers in between");
        Scanner obj = new Scanner(System.in);
        int n1 = obj.nextInt();
        int n2 = obj.nextInt();
        if (n1 >= n2) {
            System.out.println("Number2 must be greater then number1");
            System.exit(0);
        }
        while (n1 <= n2) {
            int i = 2, count = 0;
            while (i <= n1 / 2) {
                if (n1 % i == 0) {
                    count++;
                    break;
                }
                i++;
            }
            if (count == 0) {
                System.out.println(n1 + " is prime number");
            }
            n1++;
        }
    }
}
```

```
Enter a number range to generate prime numbers in between
1 5
1 is prime number
2 is prime number
3 is prime number
```

```
//Program to Display the prime Numbers From 1 to 100
import java.util.Scanner;
public class prime {
    public static void main(String[] args) {
        System.out.println("Enter a number range to generate prime numbers in between");
        Scanner obj = new Scanner(System.in);
        int n1 = obj.nextInt();
        int n2 = obj.nextInt();
        if (n1 >= n2) {
            System.out.println("Number2 must be greater then number1");
            System.exit(0);
        }
        while (n1 <= n2) {
            int i = 2, count = 0;
            while (i <= n1 / 2) {
                if (n1 % i == 0) {
                    count++;
                    break;
                }
                i++;
            }
            if (count == 0) {
                System.out.println(n1 + " is prime number");
            }
            n1++;
        }
    }
}
```

To print prime numbers from 1 to 100 in Java, the program reads in two integer values from the user 1 and 100. An “***if***” statement checks if the first number is greater than or equal to the second.

Find Prime Number Using Recursion



```
class prime {

  static boolean isPrime(int p, int i) {
    if (p <= 2) return (p == 2) ? true : false;
    if (p % i == 0) return false;
    if (i * i > p) return true;

    return isPrime(p, i + 1);
  }

  public static void main(String[] args) {
    int a = 11;

    if (isPrime(a, 7)) {
      System.out.println("The number is prime");
    }
    else {
      System.out.println("The number is not prime");
    }
  }
}
```

```
The number is prime


...Program finished with exit code 0
Press ENTER to exit console.
```

```
//Find Prime Number Using Recursion
class prime {

  static boolean isPrime(int p, int i) {
    if (p <= 2) return (p == 2) ? true : false;
    if (p % i == 0) return false;
    if (i * i > p) return true;

    return isPrime(p, i + 1);
  }

  public static void main(String[] args) {
    int a = 11;

    if (isPrime(a, 7)) {
      System.out.println("The number is prime");
    }
    else {
      System.out.println("The number is not prime");
    }
  }
}
```

Conclusion

Learning the prime number logic in Java is as essential as it is easy. To launch your coding career, you must learn the prime number program in Javascript since it serves as the base for future lessons. Enroll in a reliable programming course to build a strong foundation.

FAQs

1. **How do you handle large prime numbers in Java?**

   To deal with huge prime numbers in Java, we can use the BigInteger class. BigInteger is a ***java.math*** package that lets you conduct calculations on integers of arbitrary size.

2. **Is 1 a prime number in Java?**

   Any positive integer greater than 1 divisible by only 1 and itself is termed a prime number. Since 1 does not meet these criteria, it is not a prime number.

3. **How can we find a prime number program in Java using an array?**

   To print prime numbers from 1 to n in java, we can use the '***Math.sqrt***' method. It helps limit the range of numbers to check their divisibility.