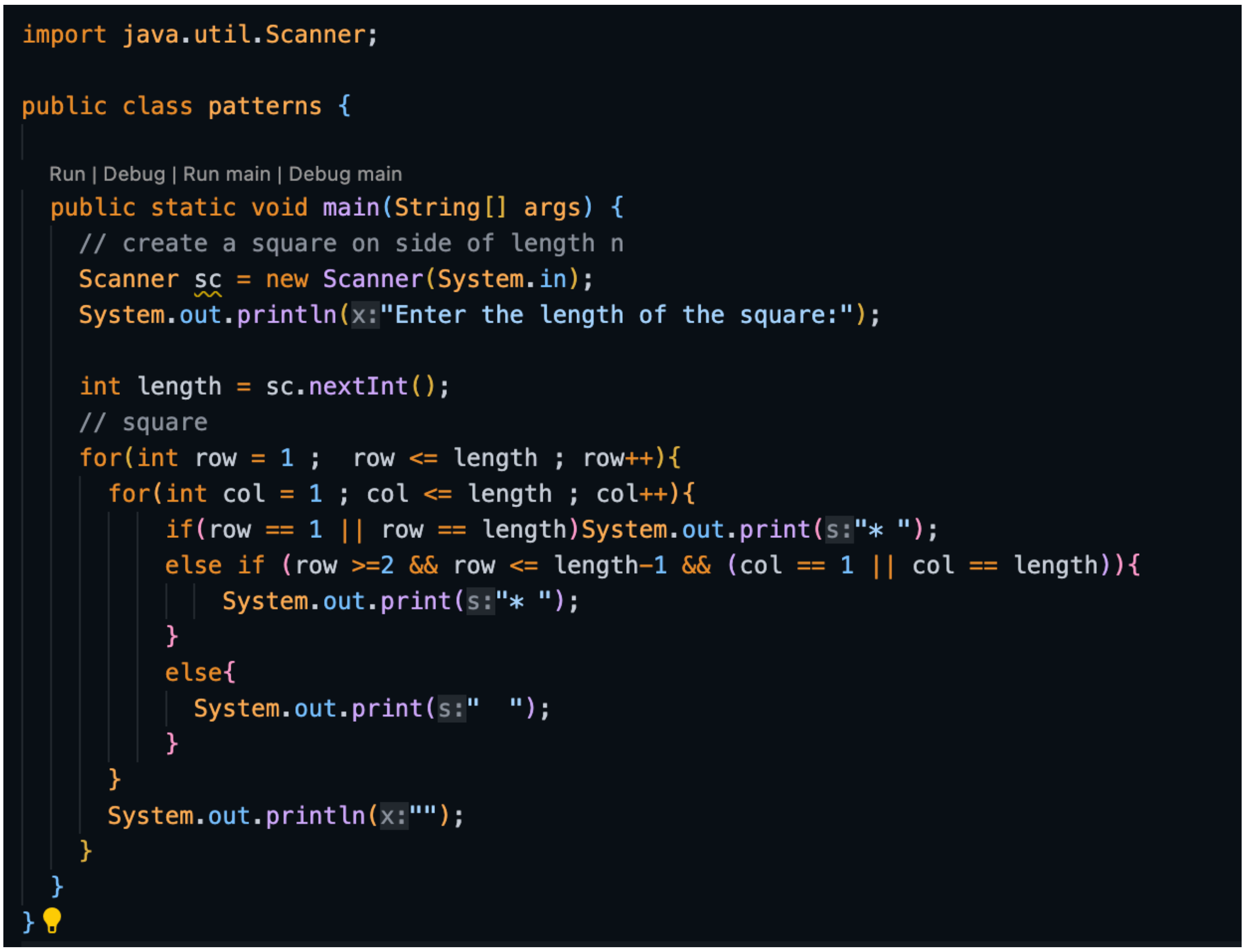
Java Assignment

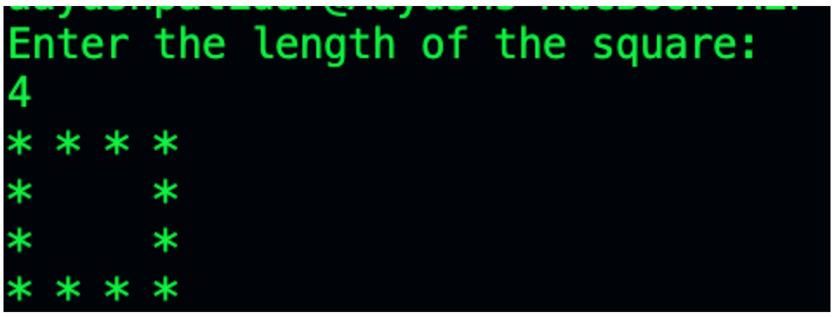
Basic Java:

1) Use loops to print patterns like a triangle or square.



```java
import java.util.Scanner;

public class patterns {

  Run | Debug | Run main | Debug main
  public static void main(String[] args) {
    // create a square on side of length n
    Scanner sc = new Scanner(System.in);
    System.out.println(x:"Enter the length of the square:");

    int length = sc.nextInt();
    // square
    for(int row = 1 ;  row <= length ; row++){
      for(int col = 1 ; col <= length ; col++){
        if(row == 1 || row == length)System.out.print(s:"* ");
        else if (row >=2 && row <= length-1 && (col == 1 || col == length)){
          System.out.print(s:"* ");
        }
        else{
          System.out.print(s:"  ");
        }
      }
      System.out.println(x:"");
    }
  }
}
```

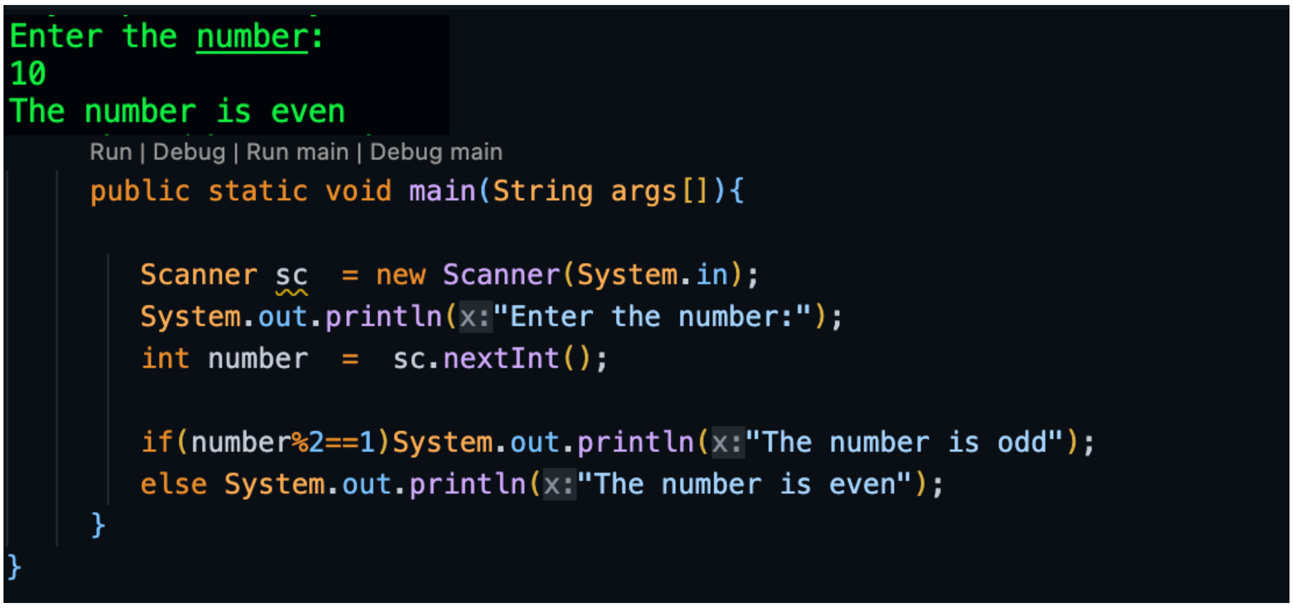
O/P Example:



```
Enter the length of the square:
4
* * * *
*     *
*     *
* * * *
```

2.Create a program to check if a number is even or odd.

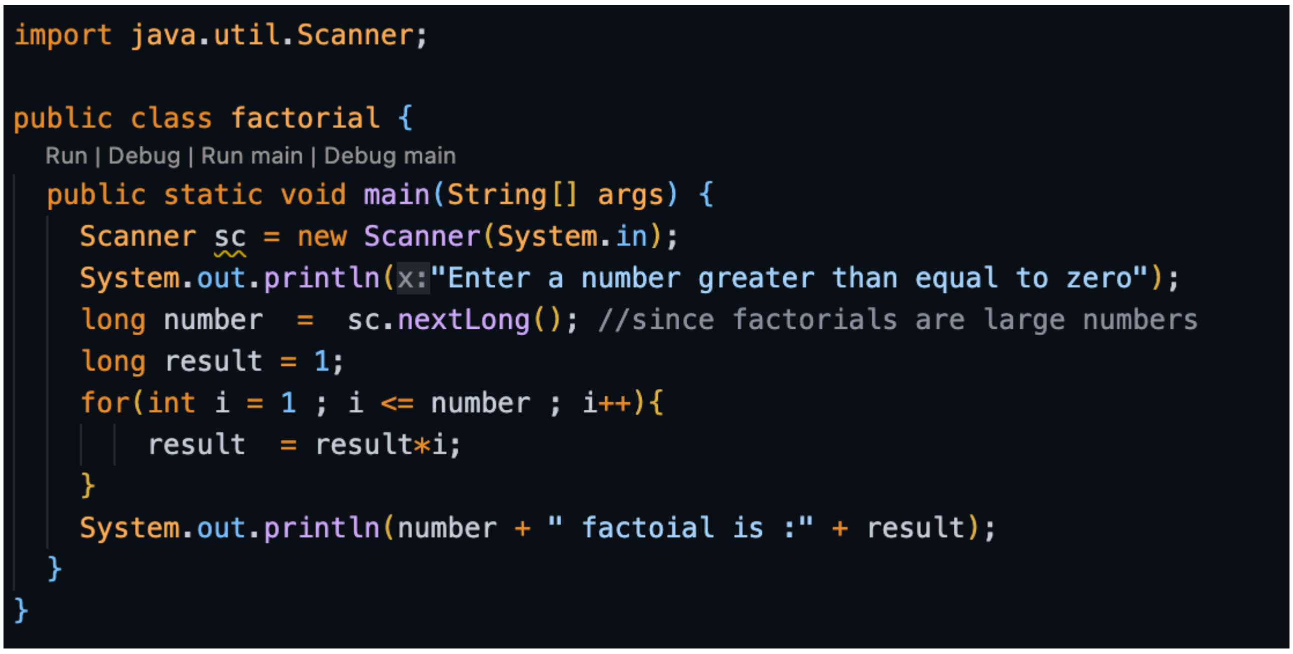
O/P Example



```
Enter the number:
10
The number is even
```

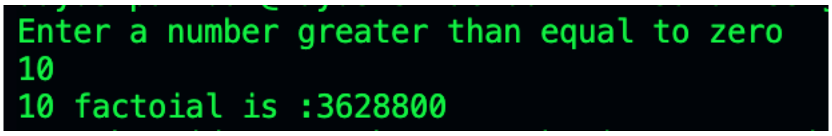
```
Run | Debug | Run main | Debug main
    public static void main(String args[]){

        Scanner sc  = new Scanner(System.in);
        System.out.println(x:"Enter the number:");
        int number  =  sc.nextInt();

        if(number%2==1)System.out.println(x:"The number is odd");
        else System.out.println(x:"The number is even");
    }
}
```

3.Implement a program to find the factorial of a given number.



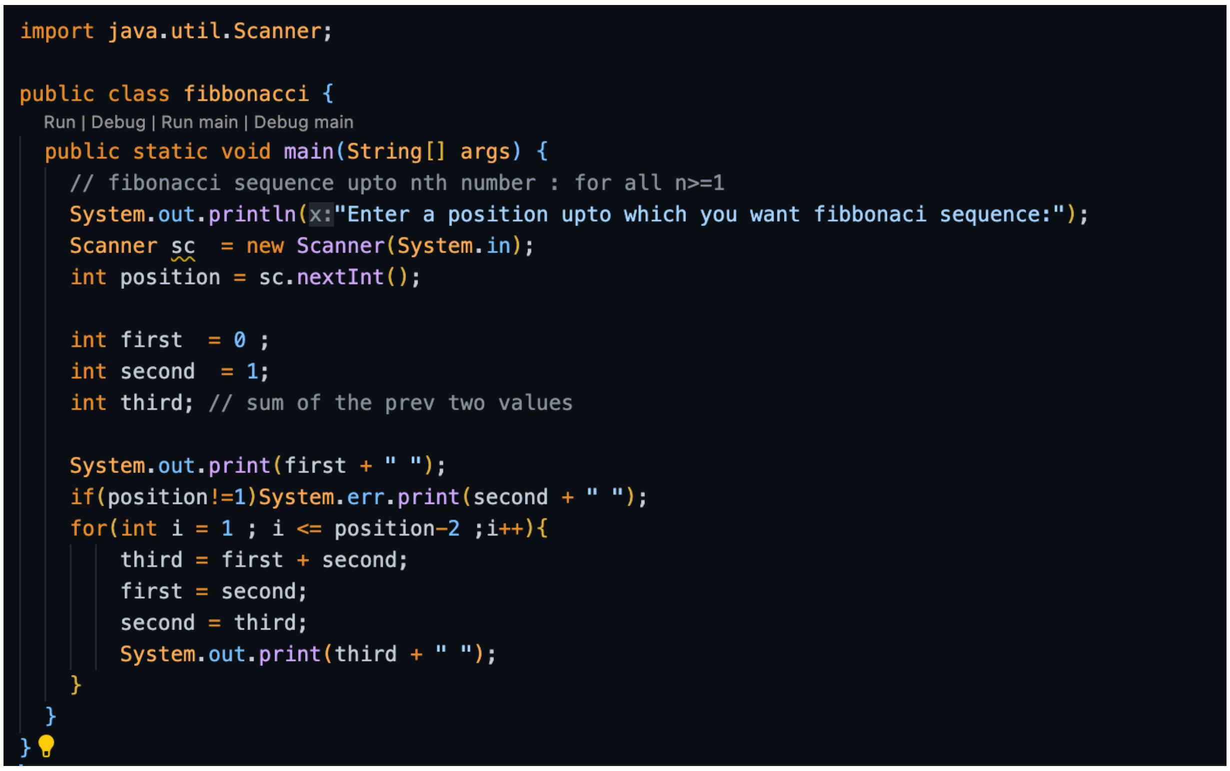
```
import java.util.Scanner;

public class factorial {
  Run | Debug | Run main | Debug main
  public static void main(String[] args) {
    Scanner sc = new Scanner(System.in);
    System.out.println(x:"Enter a number greater than equal to zero");
    long number  =  sc.nextLong(); //since factorials are large numbers
    long result = 1;
    for(int i = 1 ; i <= number ; i++){
        result  = result*i;
    }
    System.out.println(number + " factoial is :" + result);
  }
}
```

O/P Example:



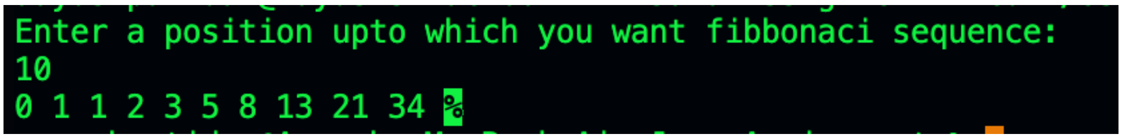
```
Enter a number greater than equal to zero
10
10 factoial is :3628800
```

4.Write a program to print the Fibonacci sequence up to a specified number.



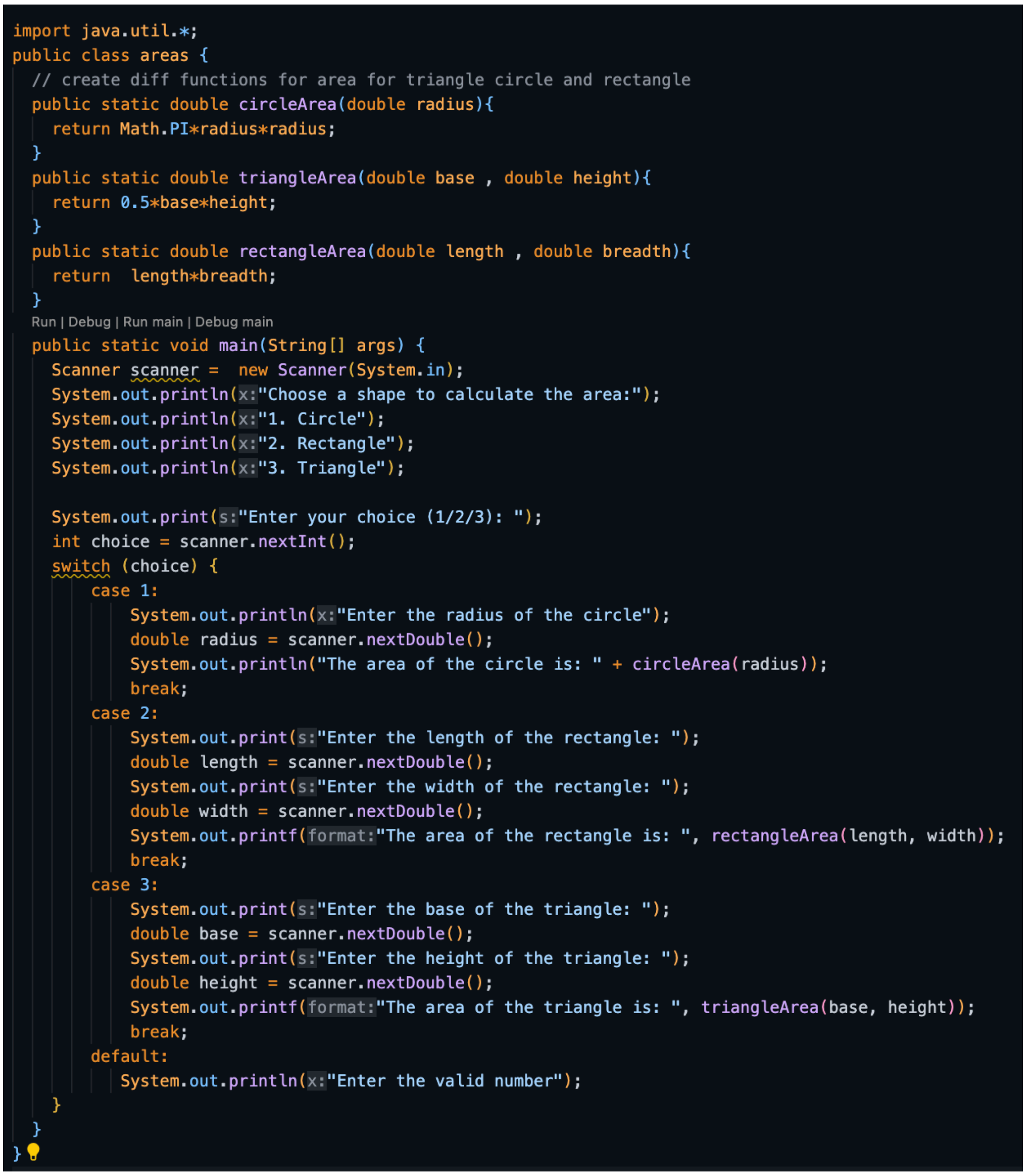
```java
import java.util.Scanner;

public class fibbonacci {
  public static void main(String[] args) {
    // fibonacci sequence upto nth number : for all n>=1
    System.out.println(x:"Enter a position upto which you want fibbonaci sequence:");
    Scanner sc  = new Scanner(System.in);
    int position = sc.nextInt();

    int first  = 0 ;
    int second  = 1;
    int third; // sum of the prev two values

    System.out.print(first + " ");
    if(position!=1)System.err.print(second + " ");
    for(int i = 1 ; i <= position-2 ;i++){
        third = first + second;
        first = second;
        second = third;
        System.out.print(third + " ");
    }
  }
}
```

O/P Example:



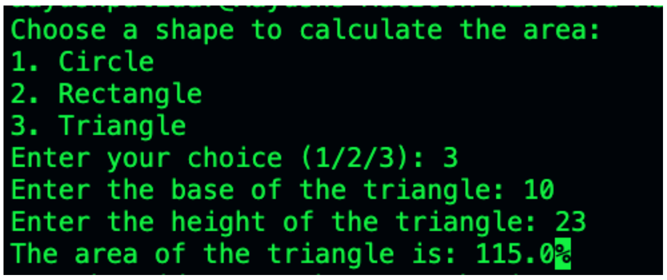
```
Enter a position upto which you want fibbonaci sequence:
10
0 1 1 2 3 5 8 13 21 34 %
```

5.Write a program to calculate the area of a circle, rectangle, or triangle based on user input.



```
import java.util.*;
public class areas {
    // create diff functions for area for triangle circle and rectangle
    public static double circleArea(double radius){
        return Math.PI*radius*radius;
    }
    public static double triangleArea(double base , double height){
        return 0.5*base*height;
    }
    public static double rectangleArea(double length , double breadth){
        return  length*breadth;
    }
    Run | Debug | Run main | Debug main
    public static void main(String[] args) {
        Scanner scanner =  new Scanner(System.in);
        System.out.println(x:"Choose a shape to calculate the area:");
        System.out.println(x:"1. Circle");
        System.out.println(x:"2. Rectangle");
        System.out.println(x:"3. Triangle");

        System.out.print(s:"Enter your choice (1/2/3): ");
        int choice = scanner.nextInt();
        switch (choice) {
            case 1:
                System.out.println(x:"Enter the radius of the circle");
                double radius = scanner.nextDouble();
                System.out.println("The area of the circle is: " + circleArea(radius));
                break;
            case 2:
                System.out.print(s:"Enter the length of the rectangle: ");
                double length = scanner.nextDouble();
                System.out.print(s:"Enter the width of the rectangle: ");
                double width = scanner.nextDouble();
                System.out.printf(format:"The area of the rectangle is: ", rectangleArea(length, width));
                break;
            case 3:
                System.out.print(s:"Enter the base of the triangle: ");
                double base = scanner.nextDouble();
                System.out.print(s:"Enter the height of the triangle: ");
                double height = scanner.nextDouble();
                System.out.printf(format:"The area of the triangle is: ", triangleArea(base, height));
                break;
            default:
               System.out.println(x:"Enter the valid number");
        }
    }
}
```

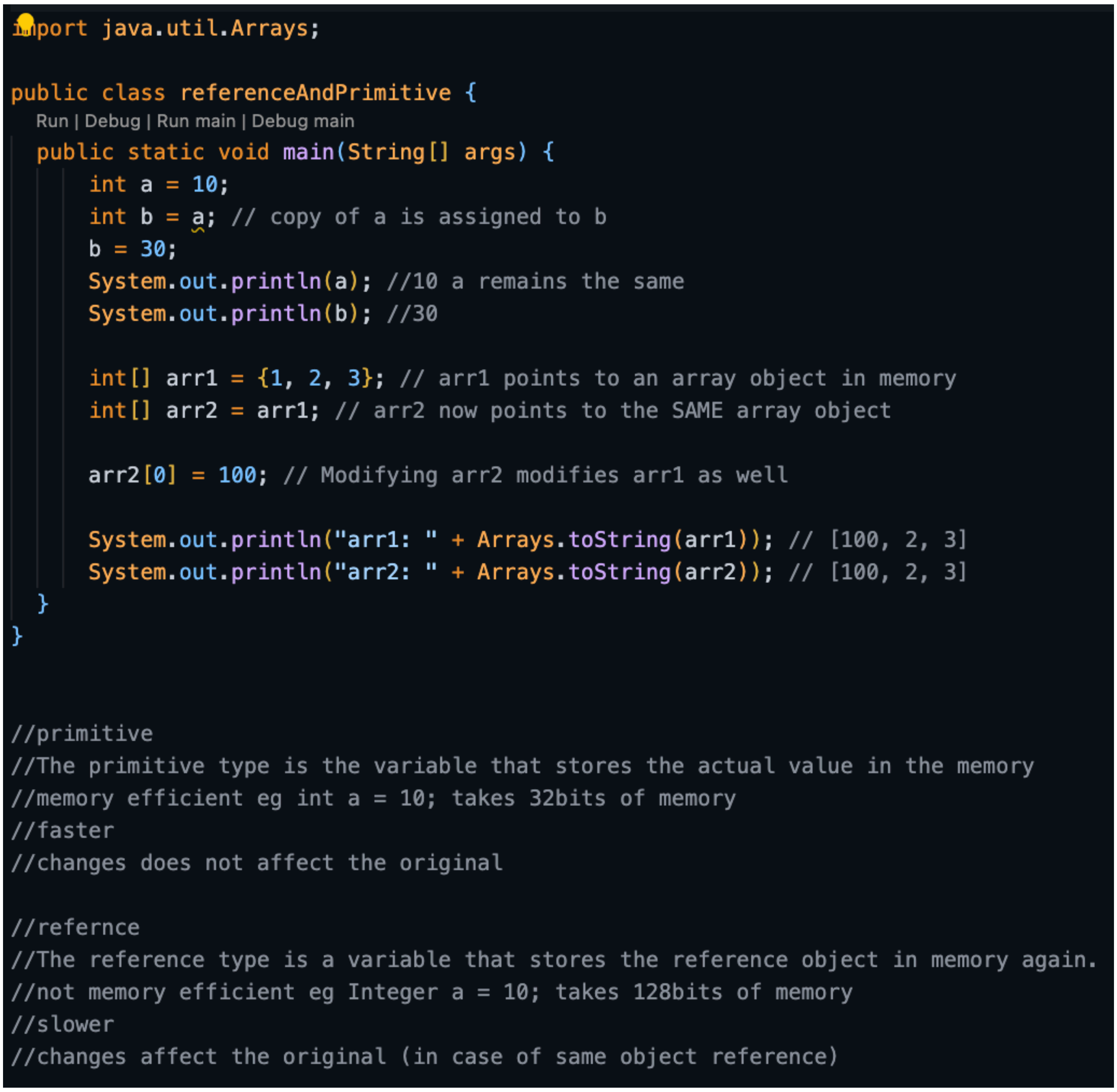
O/P Example:



```
Choose a shape to calculate the area:
1. Circle
2. Rectangle
3. Triangle
Enter your choice (1/2/3): 3
Enter the base of the triangle: 10
Enter the height of the triangle: 23
The area of the triangle is: 115.0%
```

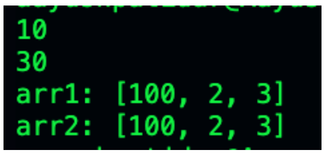
Data Types and Operators:

1. Explain the difference between primitive and reference data types with examples.



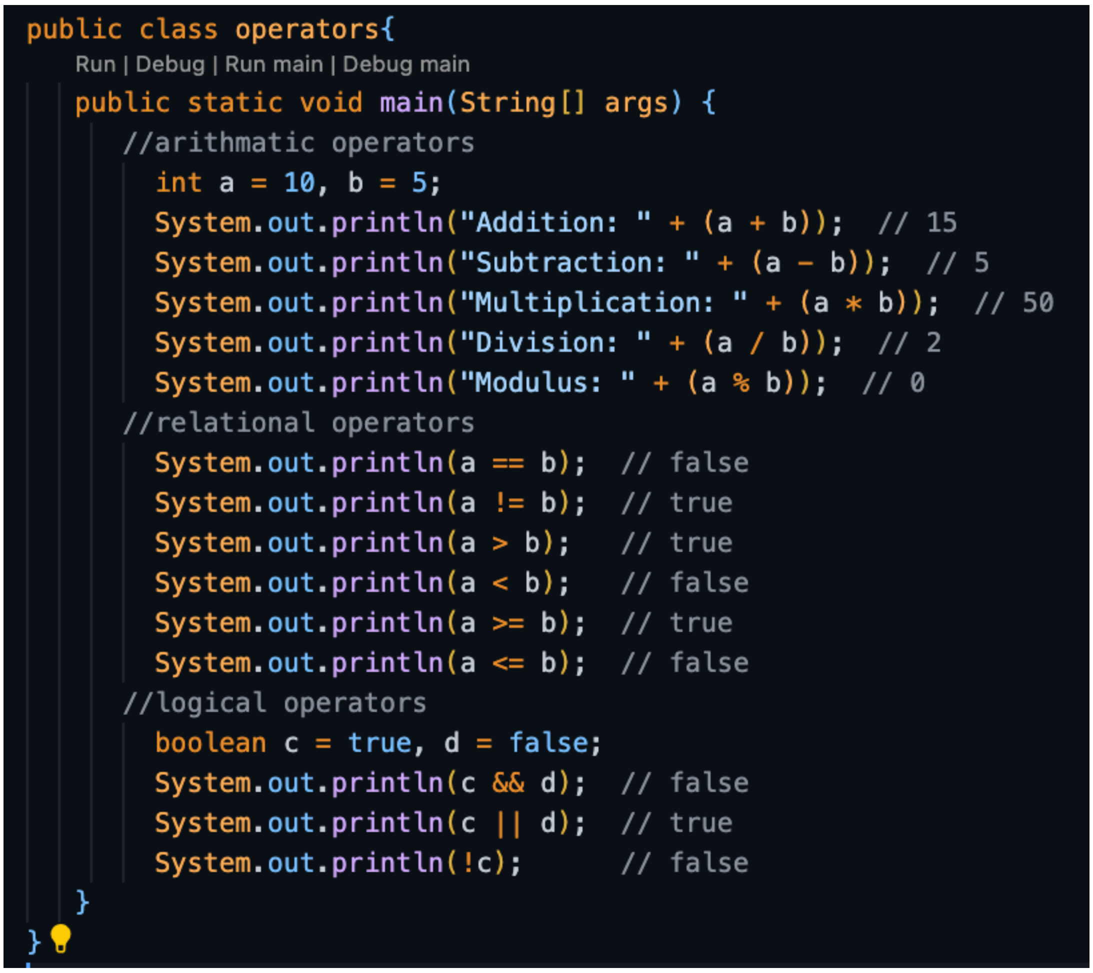
```java
import java.util.Arrays;

public class referenceAndPrimitive {
  public static void main(String[] args) {
      int a = 10;
      int b = a; // copy of a is assigned to b
      b = 30;
      System.out.println(a); //10 a remains the same
      System.out.println(b); //30

      int[] arr1 = {1, 2, 3}; // arr1 points to an array object in memory
      int[] arr2 = arr1; // arr2 now points to the SAME array object

      arr2[0] = 100; // Modifying arr2 modifies arr1 as well

      System.out.println("arr1: " + Arrays.toString(arr1)); // [100, 2, 3]
      System.out.println("arr2: " + Arrays.toString(arr2)); // [100, 2, 3]
  }
}


//primitive
//The primitive type is the variable that stores the actual value in the memory
//memory efficient eg int a = 10; takes 32bits of memory
//faster
//changes does not affect the original

//refernce
//The reference type is a variable that stores the reference object in memory again.
//not memory efficient eg Integer a = 10; takes 128bits of memory
//slower
//changes affect the original (in case of same object reference)
```

O/P:



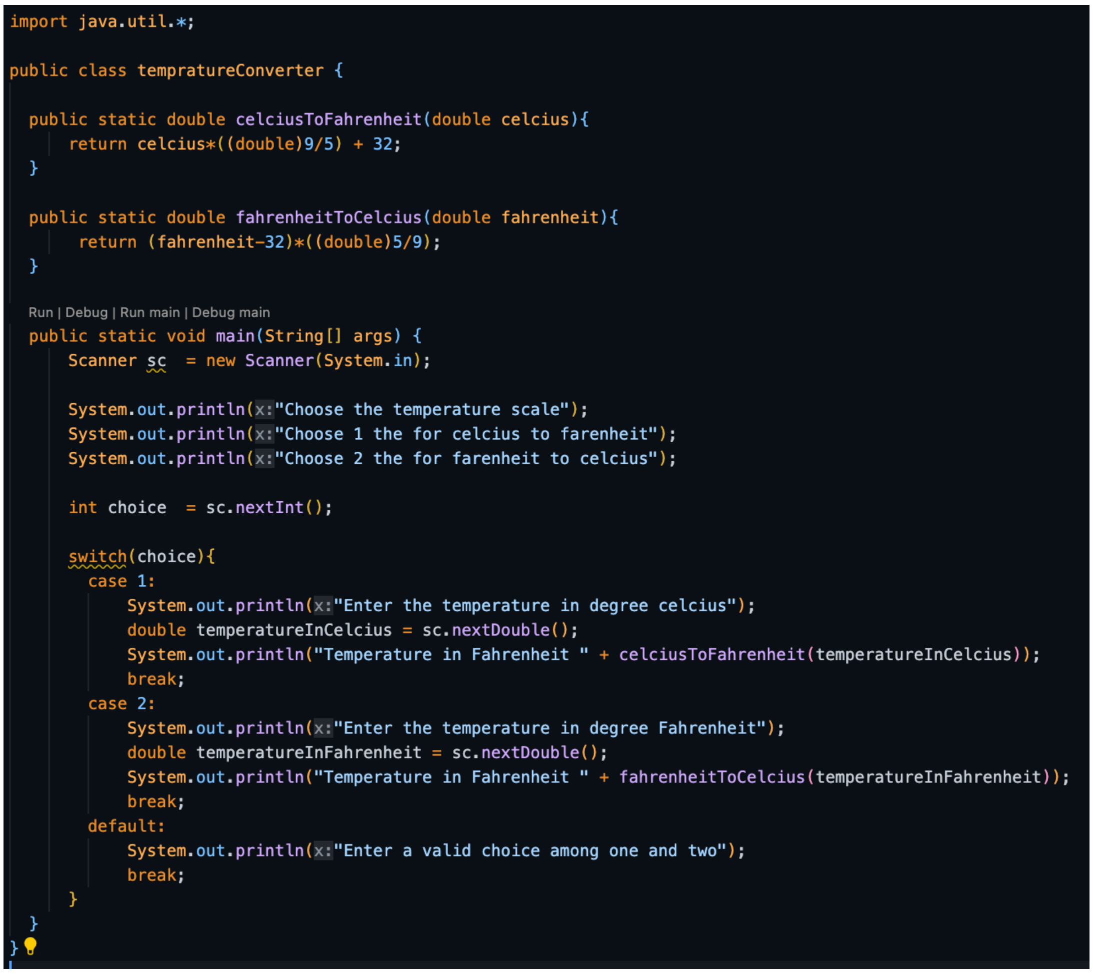
```
10
30
arr1: [100, 2, 3]
arr2: [100, 2, 3]
```

2.Write a program to demonstrate the use of arithmetic, logical, and relational operators.



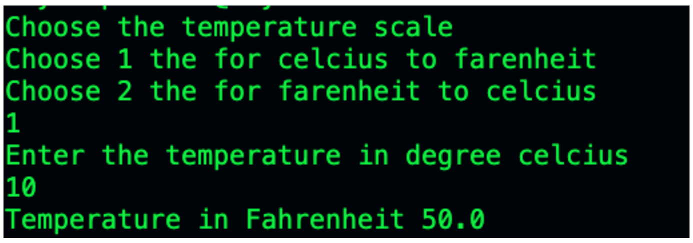
```
public class operators{
    Run | Debug | Run main | Debug main
    public static void main(String[] args) {
        //arithmatic operators
          int a = 10, b = 5;
          System.out.println("Addition: " + (a + b));  // 15
          System.out.println("Subtraction: " + (a - b));  // 5
          System.out.println("Multiplication: " + (a * b));  // 50
          System.out.println("Division: " + (a / b));  // 2
          System.out.println("Modulus: " + (a % b));  // 0
        //relational operators
          System.out.println(a == b);  // false
          System.out.println(a != b);  // true
          System.out.println(a > b);   // true
          System.out.println(a < b);   // false
          System.out.println(a >= b);  // true
          System.out.println(a <= b);  // false
        //logical operators
          boolean c = true, d = false;
          System.out.println(c && d);  // false
          System.out.println(c || d);  // true
          System.out.println(!c);      // false
    }
}
```

3.Create a program to convert a temperature from Celsius to Fahrenheit and vice versa.



```
import java.util.*;

public class tempratureConverter {

  public static double celciusToFahrenheit(double celcius){
      return celcius*((double)9/5) + 32;
  }

  public static double fahrenheitToCelcius(double fahrenheit){
       return (fahrenheit-32)*((double)5/9);
  }

  Run | Debug | Run main | Debug main
  public static void main(String[] args) {
      Scanner sc  = new Scanner(System.in);

      System.out.println(x:"Choose the temperature scale");
      System.out.println(x:"Choose 1 the for celcius to farenheit");
      System.out.println(x:"Choose 2 the for farenheit to celcius");

      int choice  = sc.nextInt();

      switch(choice){
        case 1:
            System.out.println(x:"Enter the temperature in degree celcius");
            double temperatureInCelcius = sc.nextDouble();
            System.out.println("Temperature in Fahrenheit " + celciusToFahrenheit(temperatureInCelcius));
            break;
        case 2:
            System.out.println(x:"Enter the temperature in degree Fahrenheit");
            double temperatureInFahrenheit = sc.nextDouble();
            System.out.println("Temperature in Fahrenheit " + fahrenheitToCelcius(temperatureInFahrenheit));
            break;
        default:
            System.out.println(x:"Enter a valid choice among one and two");
            break;
      }
  }
}
```

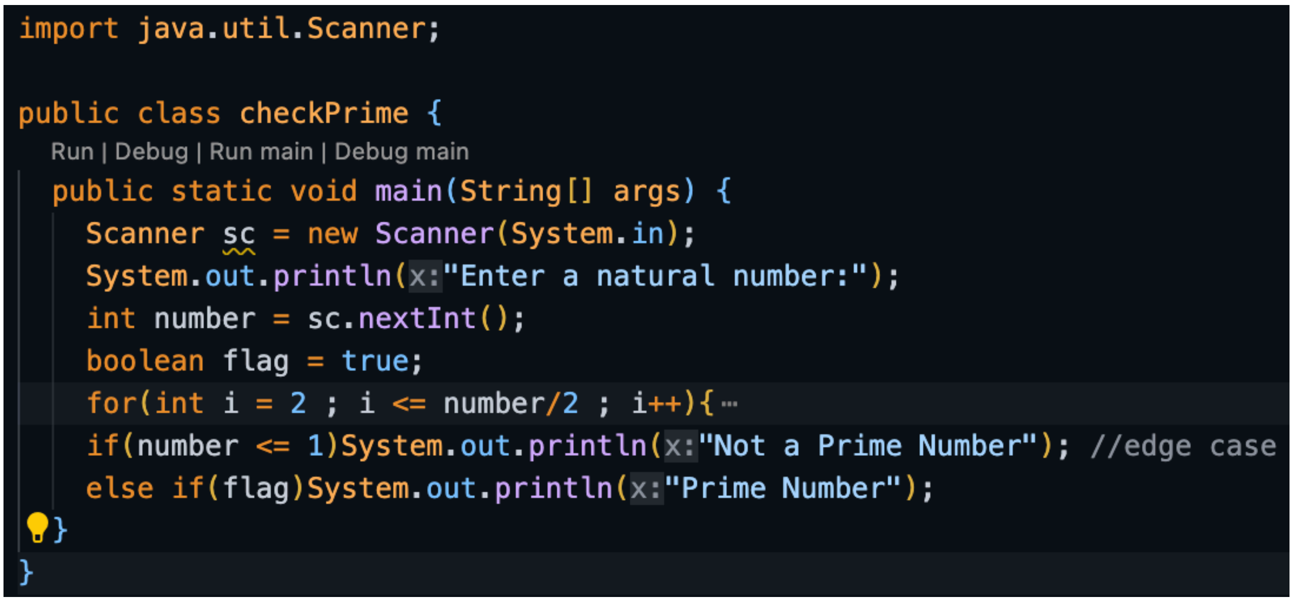
O/P Example:



```
Choose the temperature scale
Choose 1 the for celcius to farenheit
Choose 2 the for farenheit to celcius
1
Enter the temperature in degree celcius
10
Temperature in Fahrenheit 50.0
```

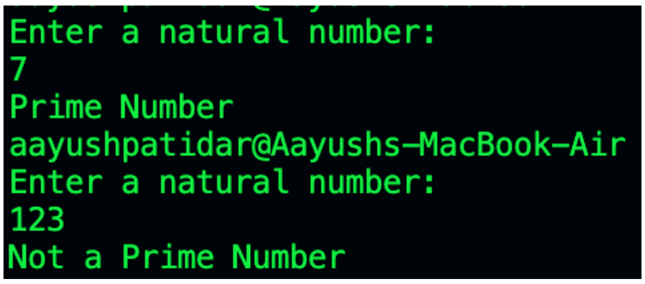
Control Flow Statements:

1.Write a program to check if a given number is prime using an if-else statement.



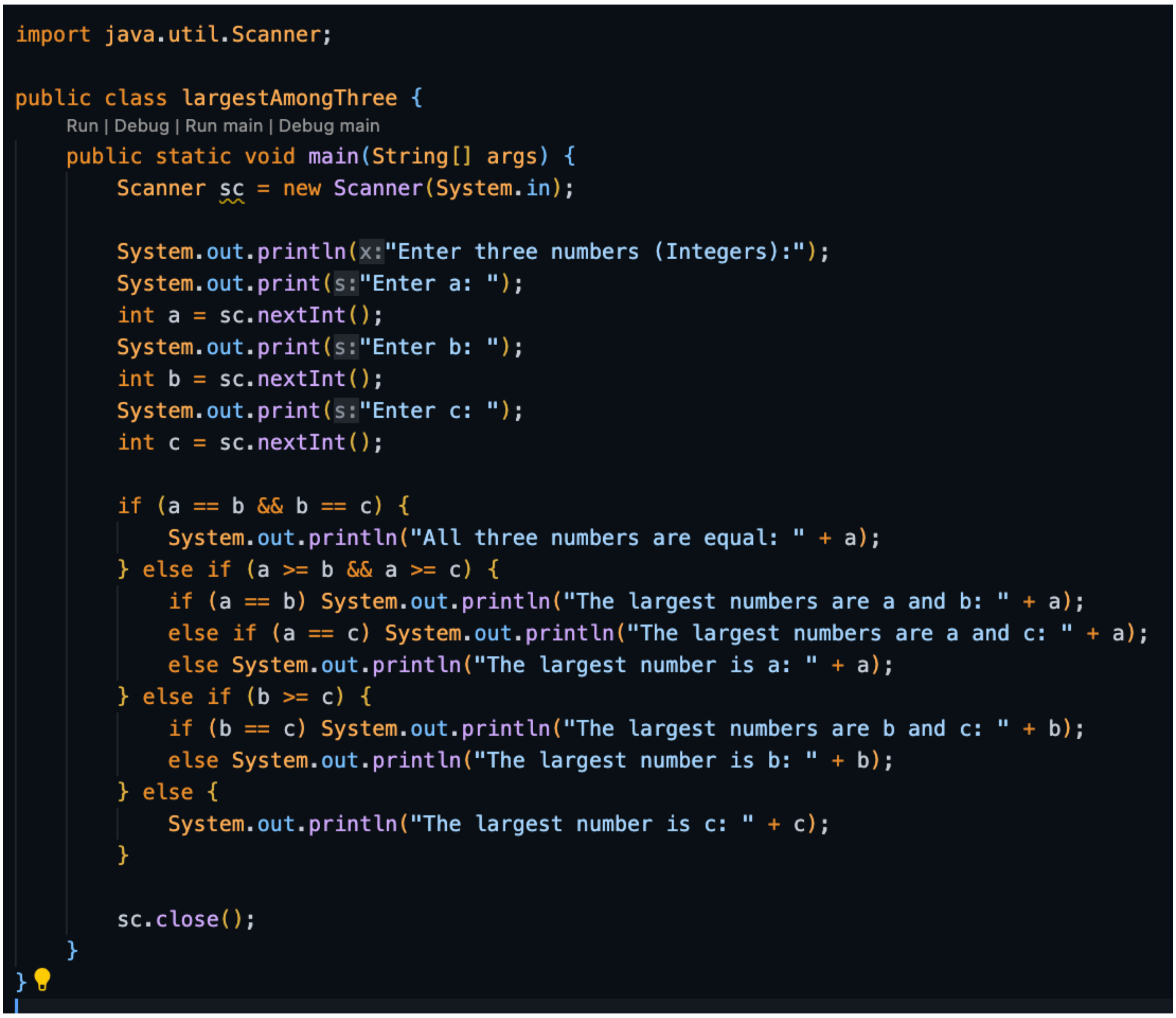
```java
import java.util.Scanner;

public class checkPrime {
  Run | Debug | Run main | Debug main
  public static void main(String[] args) {
    Scanner sc = new Scanner(System.in);
    System.out.println(x:"Enter a natural number:");
    int number = sc.nextInt();
    boolean flag = true;
    for(int i = 2 ; i <= number/2 ; i++){⋯
    if(number <= 1)System.out.println(x:"Not a Prime Number"); //edge case
    else if(flag)System.out.println(x:"Prime Number");
  }
}
```

O/P Example:



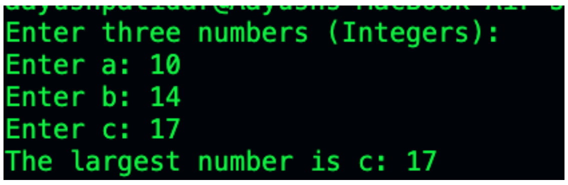
```
Enter a natural number:
7
Prime Number
aayushpatidar@Aayushs-MacBook-Air
Enter a natural number:
123
Not a Prime Number
```

2.Implement a program to find the largest number among three given numbers using a conditional statement.



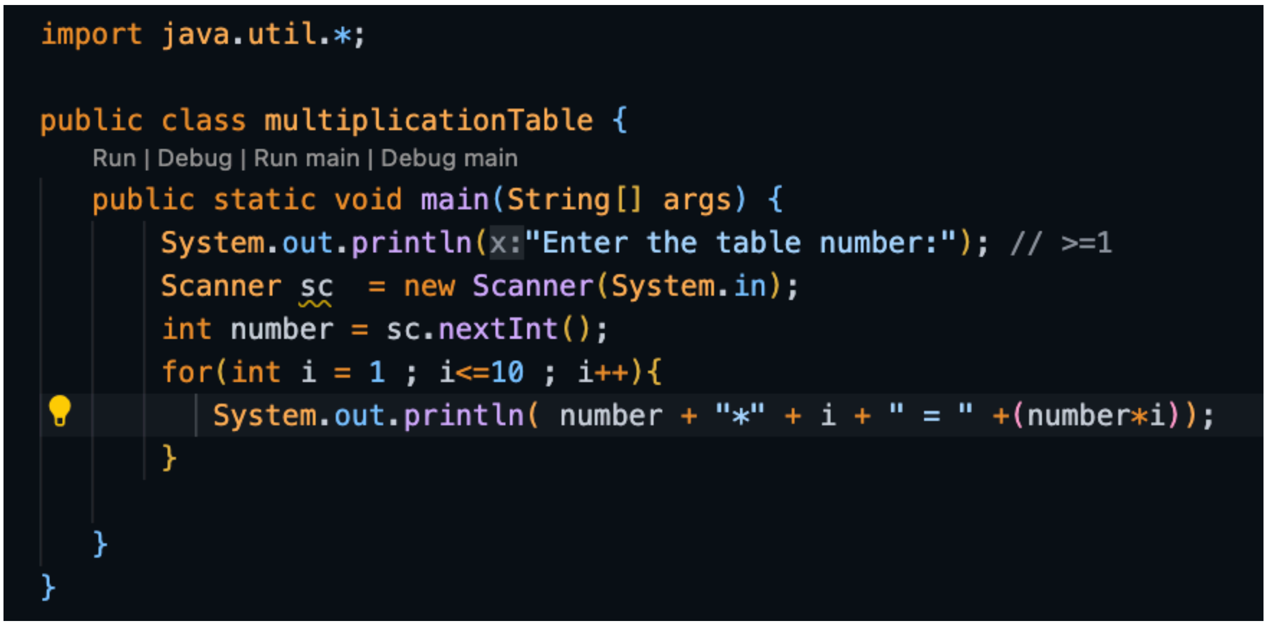
```java
import java.util.Scanner;

public class largestAmongThree {
    Run | Debug | Run main | Debug main
    public static void main(String[] args) {
        Scanner sc = new Scanner(System.in);

        System.out.println(x:"Enter three numbers (Integers):");
        System.out.print(s:"Enter a: ");
        int a = sc.nextInt();
        System.out.print(s:"Enter b: ");
        int b = sc.nextInt();
        System.out.print(s:"Enter c: ");
        int c = sc.nextInt();

        if (a == b && b == c) {
            System.out.println("All three numbers are equal: " + a);
        } else if (a >= b && a >= c) {
            if (a == b) System.out.println("The largest numbers are a and b: " + a);
            else if (a == c) System.out.println("The largest numbers are a and c: " + a);
            else System.out.println("The largest number is a: " + a);
        } else if (b >= c) {
            if (b == c) System.out.println("The largest numbers are b and c: " + b);
            else System.out.println("The largest number is b: " + b);
        } else {
            System.out.println("The largest number is c: " + c);
        }

        sc.close();
    }
}
```

O/P Example:



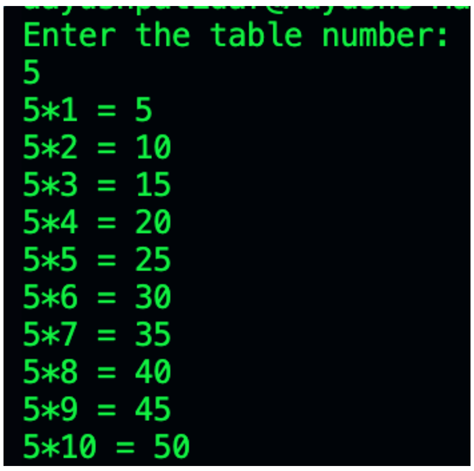
```
Enter three numbers (Integers):
Enter a: 10
Enter b: 14
Enter c: 17
The largest number is c: 17
```

3.Use a for loop to print a multiplication table.



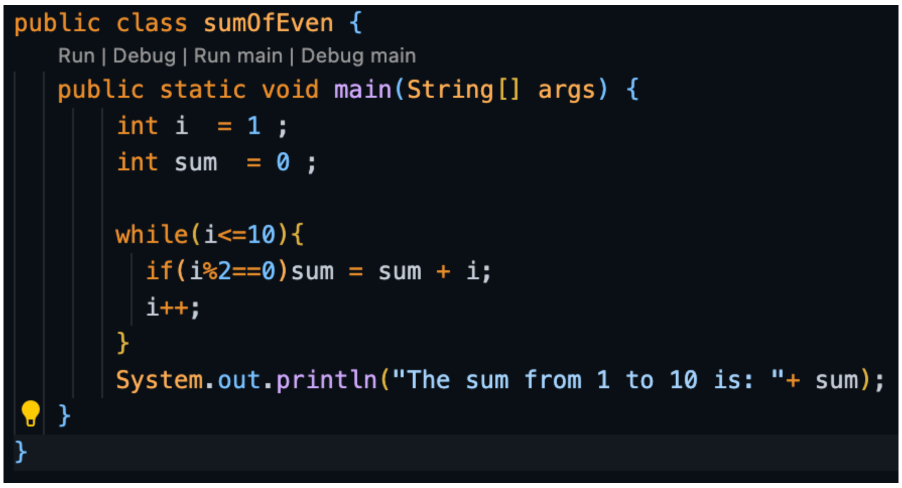
```java
import java.util.*;

public class multiplicationTable {
    public static void main(String[] args) {
        System.out.println(x:"Enter the table number:"); // >=1
        Scanner sc  = new Scanner(System.in);
        int number = sc.nextInt();
        for(int i = 1 ; i<=10 ; i++){
            System.out.println( number + "*" + i + " = " +(number*i));
        }

    }
}
```

O/P Example



```
Enter the table number:
5
5*1 = 5
5*2 = 10
5*3 = 15
5*4 = 20
5*5 = 25
5*6 = 30
5*7 = 35
5*8 = 40
5*9 = 45
5*10 = 50
```

4.Create a program to calculate the sum of even numbers from 1 to 10 using a while loop.



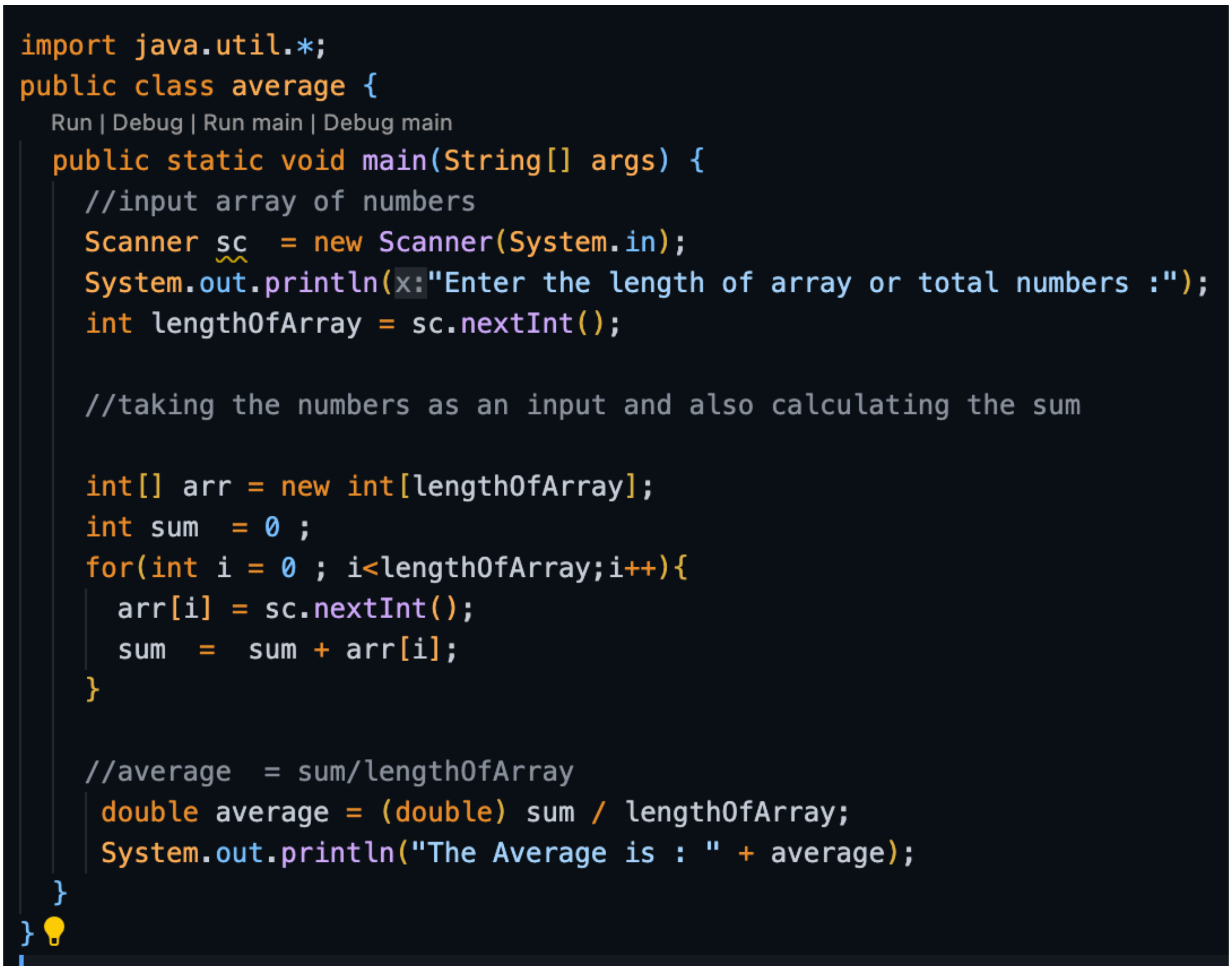
```java
public class sumOfEven {
    public static void main(String[] args) {
        int i  = 1 ;
        int sum  = 0 ;

        while(i<=10){
            if(i%2==0)sum = sum + i;
            i++;
        }
        System.out.println("The sum from 1 to 10 is: "+ sum);
    }
}
```

O/P Example

```
The sum from 1 to 10 is: 30
```

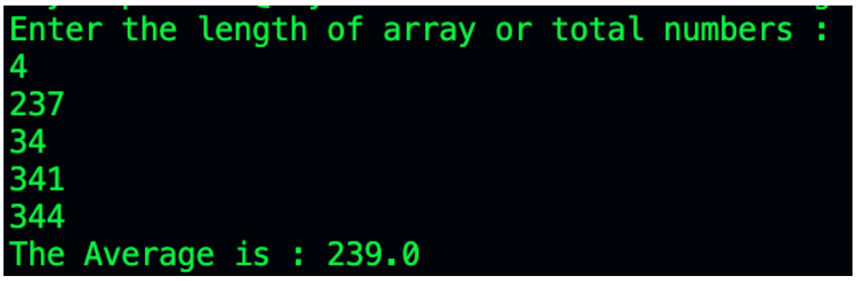
Arrays:

1.Write a program to find the average of elements in an array.



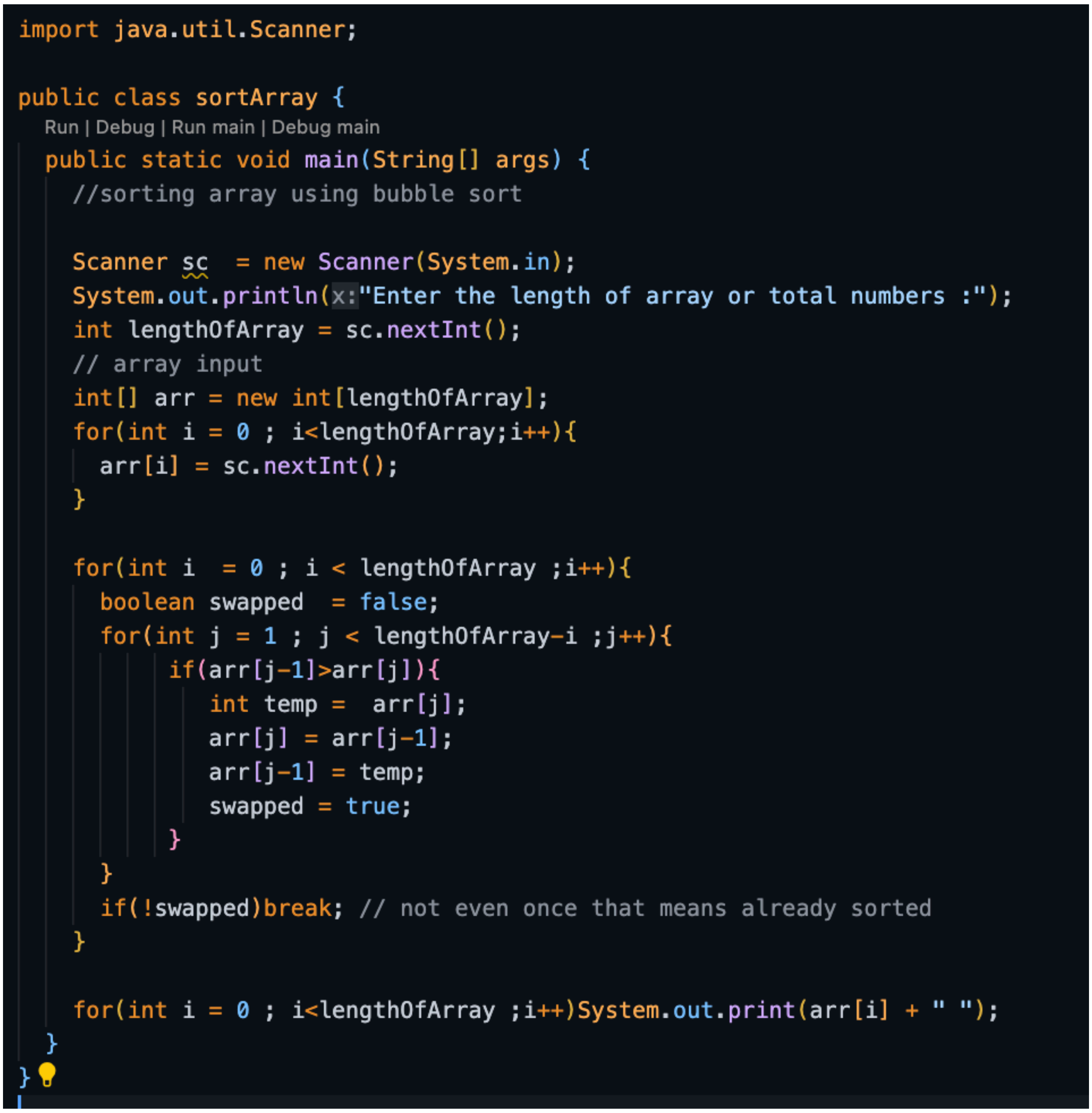
```
import java.util.*;
public class average {
    Run | Debug | Run main | Debug main
    public static void main(String[] args) {
        //input array of numbers
        Scanner sc  = new Scanner(System.in);
        System.out.println(x:"Enter the length of array or total numbers :");
        int lengthOfArray = sc.nextInt();

        //taking the numbers as an input and also calculating the sum

        int[] arr = new int[lengthOfArray];
        int sum  = 0 ;
        for(int i = 0 ; i<lengthOfArray;i++){
            arr[i] = sc.nextInt();
            sum  =  sum + arr[i];
        }

        //average  = sum/lengthOfArray
         double average = (double) sum / lengthOfArray;
         System.out.println("The Average is : " + average);
    }
}
```

O/P Example



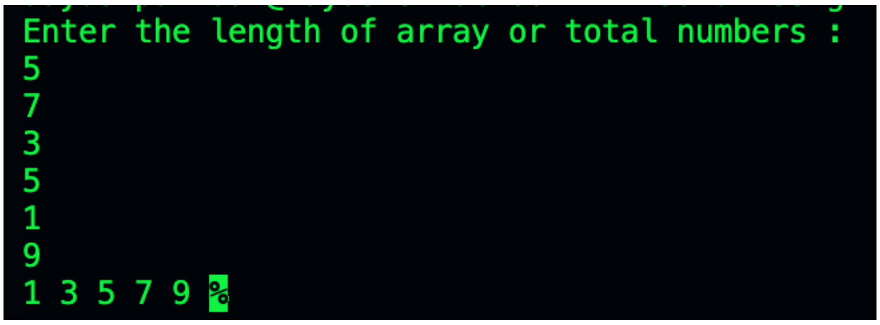
```
Enter the length of array or total numbers :
4
237
34
341
344
The Average is : 239.0
```

2.Implement a function to sort an array in ascending order using bubble sort or selection sort.



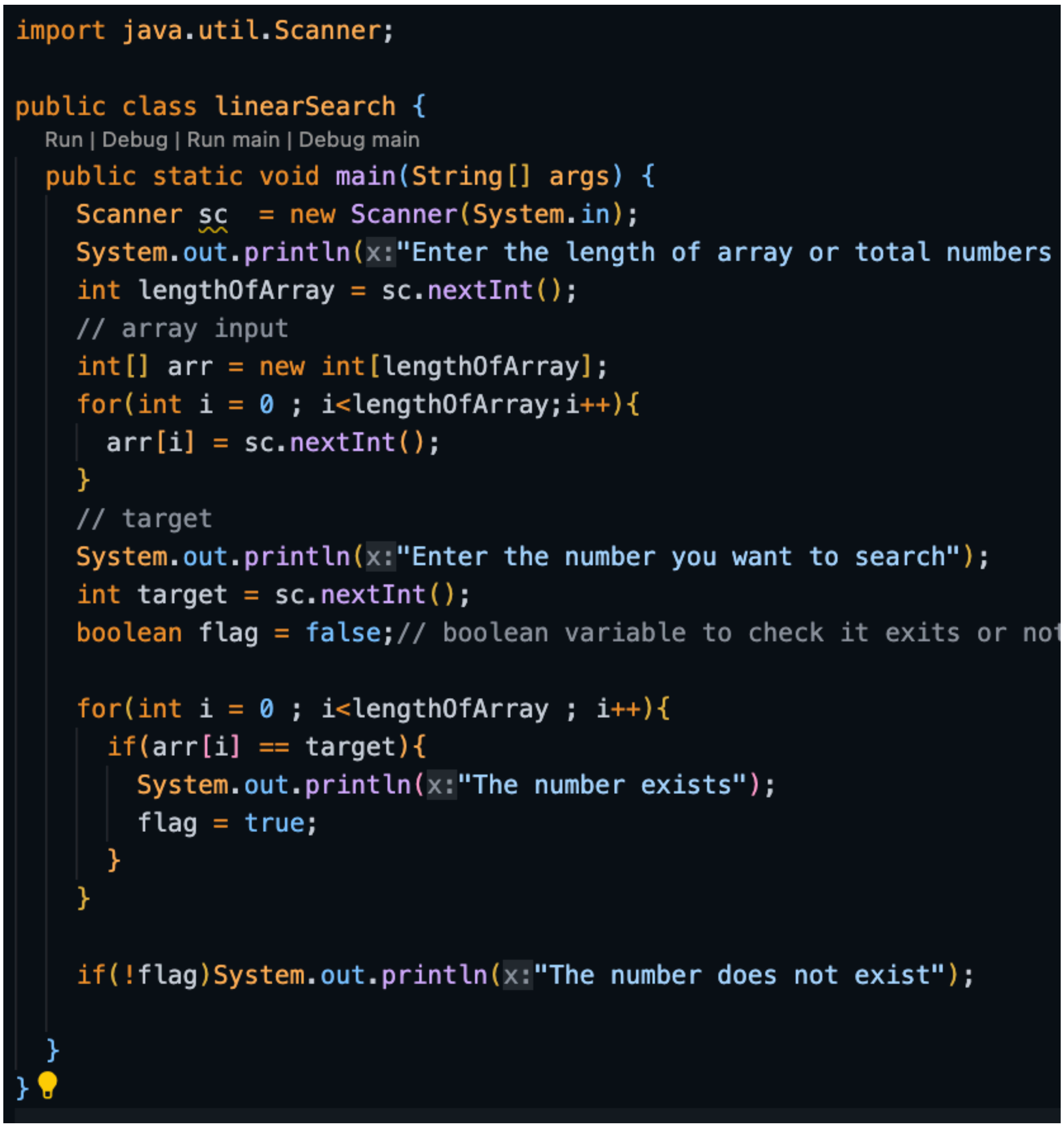
```
import java.util.Scanner;

public class sortArray {
  Run | Debug | Run main | Debug main
  public static void main(String[] args) {
    //sorting array using bubble sort

    Scanner sc  = new Scanner(System.in);
    System.out.println(x:"Enter the length of array or total numbers :");
    int lengthOfArray = sc.nextInt();
    // array input
    int[] arr = new int[lengthOfArray];
    for(int i = 0 ; i<lengthOfArray;i++){
      arr[i] = sc.nextInt();
    }

    for(int i  = 0 ; i < lengthOfArray ;i++){
      boolean swapped  = false;
      for(int j = 1 ; j < lengthOfArray-i ;j++){
          if(arr[j-1]>arr[j]){
             int temp =  arr[j];
             arr[j] = arr[j-1];
             arr[j-1] = temp;
             swapped = true;
          }
      }
      if(!swapped)break; // not even once that means already sorted
    }

    for(int i = 0 ; i<lengthOfArray ;i++)System.out.print(arr[i] + " ");
  }
}
```

O/P Example



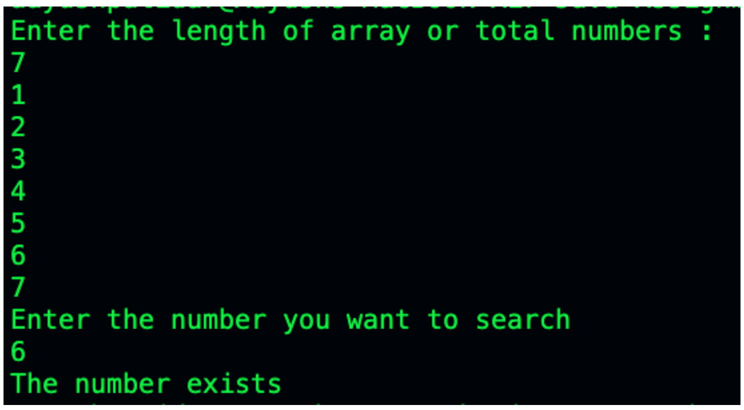
```
Enter the length of array or total numbers :
5
7
3
5
1
9
1 3 5 7 9 %
```

3.Create a program to search for a specific element within an array using linear search.



```
import java.util.Scanner;

public class linearSearch {
   Run | Debug | Run main | Debug main
   public static void main(String[] args) {
      Scanner sc  = new Scanner(System.in);
      System.out.println(x:"Enter the length of array or total numbers
      int lengthOfArray = sc.nextInt();
      // array input
      int[] arr = new int[lengthOfArray];
      for(int i = 0 ; i<lengthOfArray;i++){
         arr[i] = sc.nextInt();
      }
      // target
      System.out.println(x:"Enter the number you want to search");
      int target = sc.nextInt();
      boolean flag = false;// boolean variable to check it exits or not

      for(int i = 0 ; i<lengthOfArray ; i++){
         if(arr[i] == target){
            System.out.println(x:"The number exists");
            flag = true;
         }
      }

      if(!flag)System.out.println(x:"The number does not exist");

   }
}
```

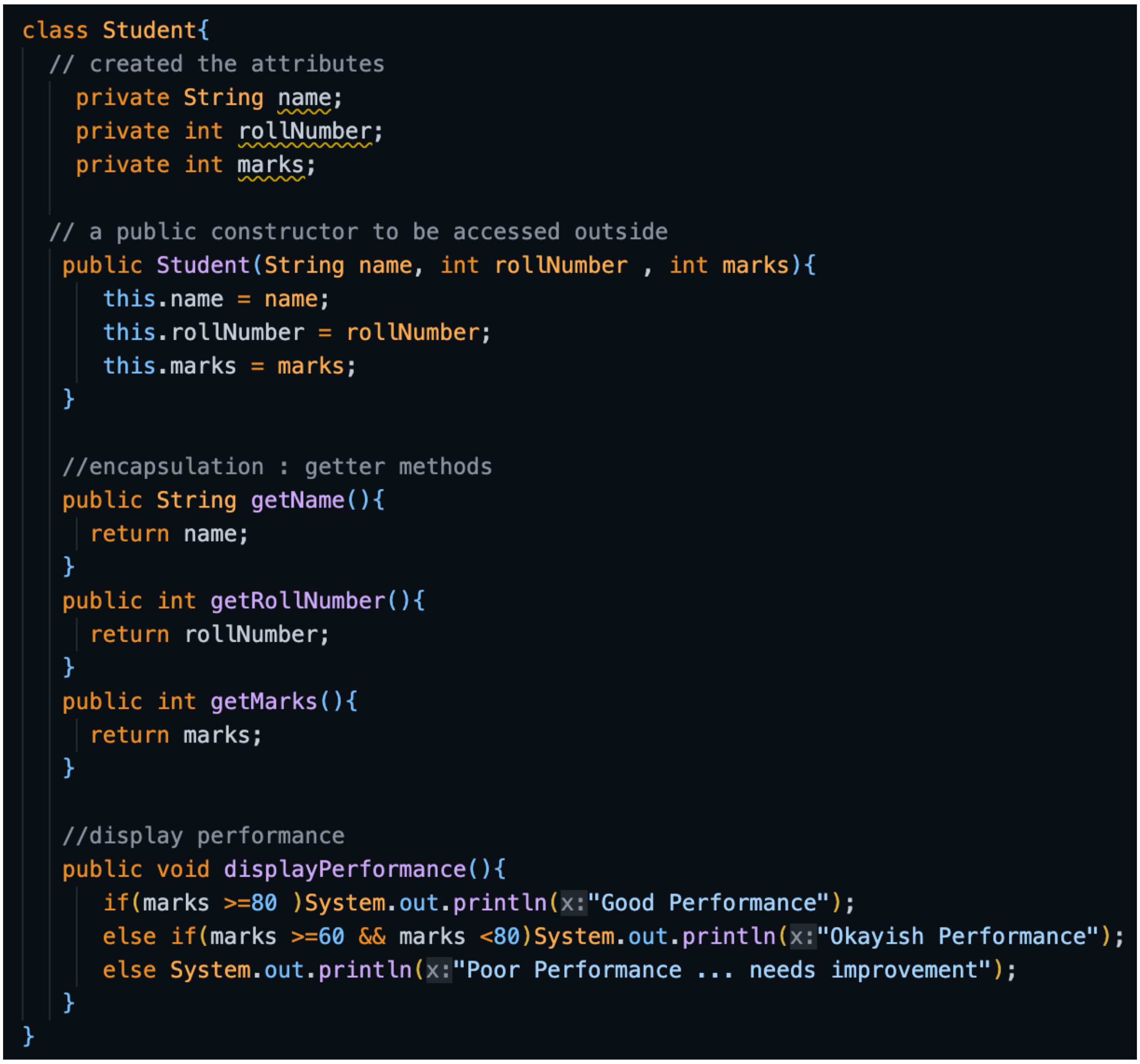
O/P Example



```
Enter the length of array or total numbers :
7
1
2
3
4
5
6
7
Enter the number you want to search
6
The number exists
```

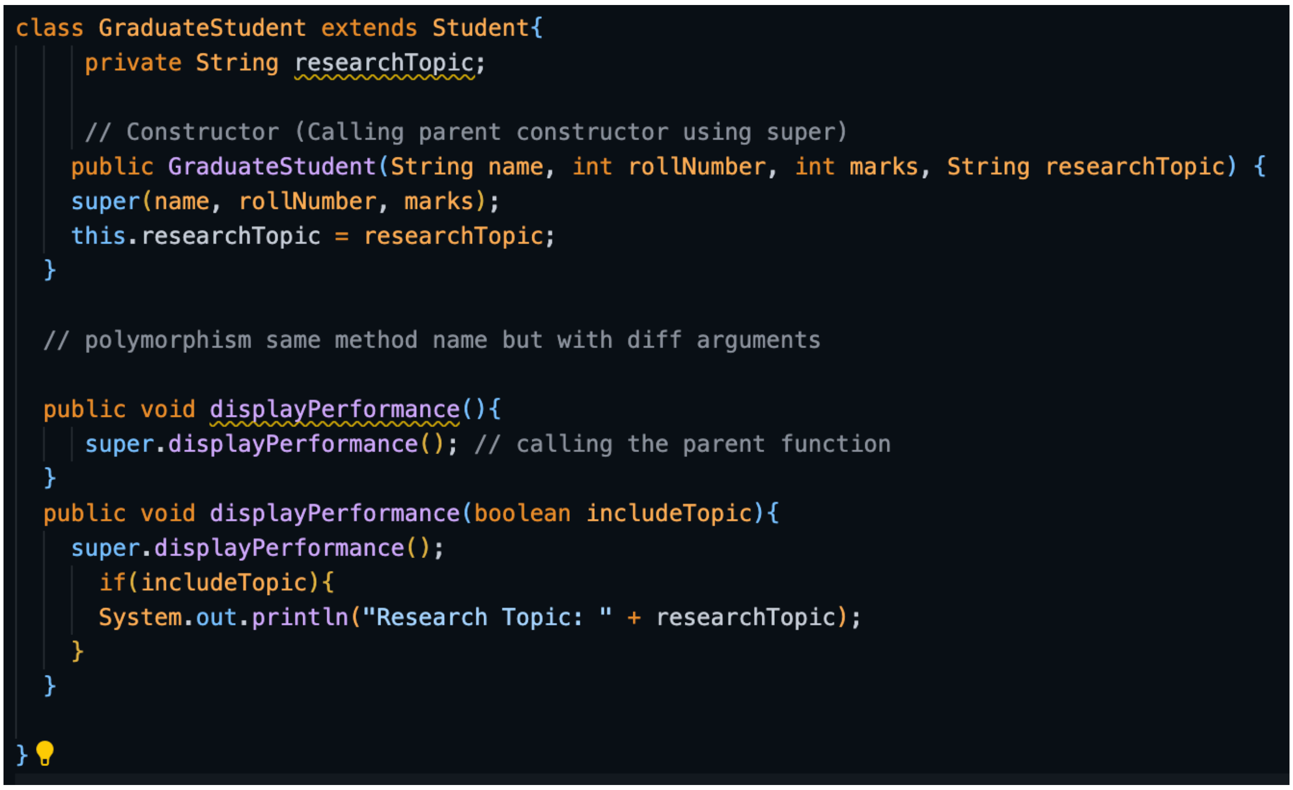
Object Oriented Programming (OOP):

Create a class to represent a student with attributes like name, roll number, and marks.
Implement inheritance to create a "GraduateStudent" class that extends the "Student" class with
additional features.
Demonstrate polymorphism by creating methods with the same name but different parameters in a
parent and child class.
Explain the concept of encapsulation with a suitable example.



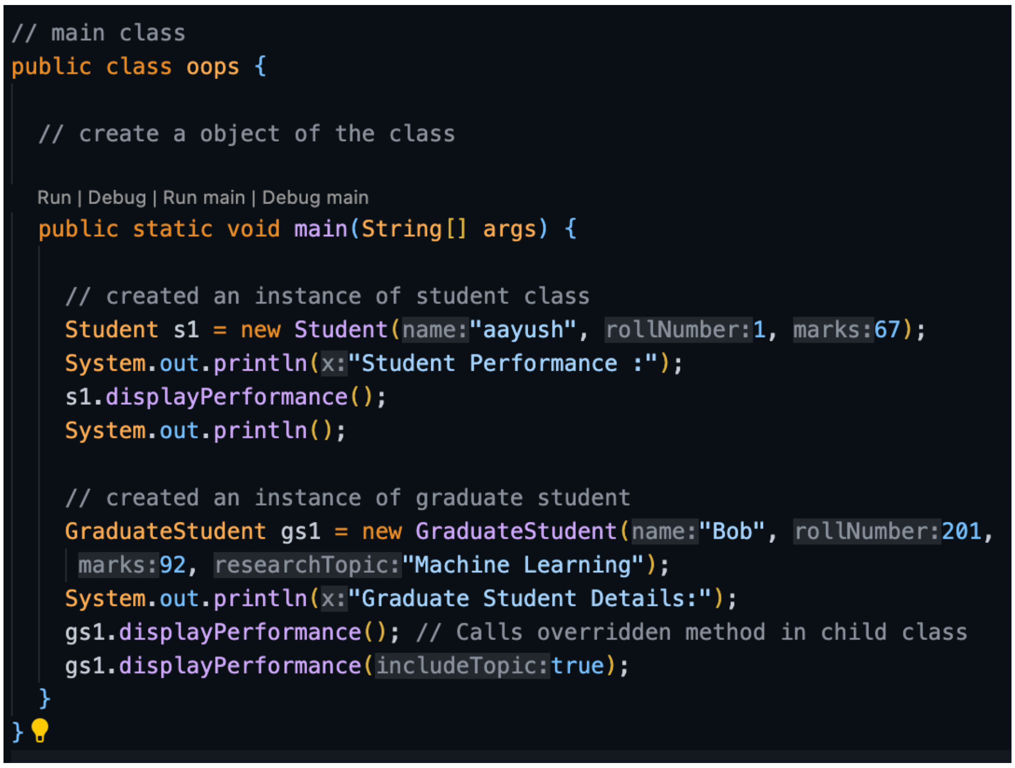
```java
class Student{
  // created the attributes
    private String name;
    private int rollNumber;
    private int marks;

  // a public constructor to be accessed outside
   public Student(String name, int rollNumber , int marks){
      this.name = name;
      this.rollNumber = rollNumber;
      this.marks = marks;
   }

   //encapsulation : getter methods
   public String getName(){
     return name;
   }
   public int getRollNumber(){
     return rollNumber;
   }
   public int getMarks(){
     return marks;
   }

   //display performance
   public void displayPerformance(){
      if(marks >=80 )System.out.println(x:"Good Performance");
      else if(marks >=60 && marks <80)System.out.println(x:"Okayish Performance");
      else System.out.println(x:"Poor Performance ... needs improvement");
   }
}
```

GraduateStudent Class



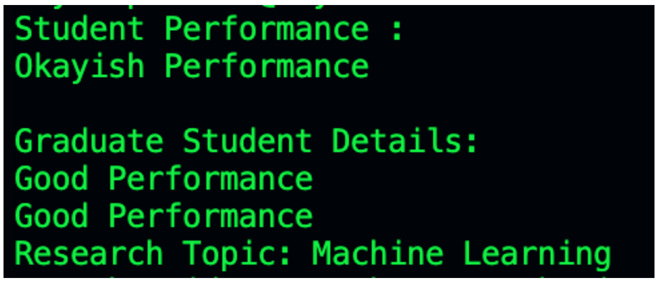
```java
class GraduateStudent extends Student{
      private String researchTopic;

      // Constructor (Calling parent constructor using super)
    public GraduateStudent(String name, int rollNumber, int marks, String researchTopic) {
    super(name, rollNumber, marks);
    this.researchTopic = researchTopic;
  }

  // polymorphism same method name but with diff arguments

  public void displayPerformance(){
      super.displayPerformance(); // calling the parent function
  }
  public void displayPerformance(boolean includeTopic){
    super.displayPerformance();
      if(includeTopic){
      System.out.println("Research Topic: " + researchTopic);
    }
  }

}
```

Main Class.



```java
// main class
public class oops {

  // create a object of the class

  Run | Debug | Run main | Debug main
  public static void main(String[] args) {

    // created an instance of student class
    Student s1 = new Student(name:"aayush", rollNumber:1, marks:67);
    System.out.println(x:"Student Performance :");
    s1.displayPerformance();
    System.out.println();

    // created an instance of graduate student
    GraduateStudent gs1 = new GraduateStudent(name:"Bob", rollNumber:201,
     marks:92, researchTopic:"Machine Learning");
    System.out.println(x:"Graduate Student Details:");
    gs1.displayPerformance(); // Calls overridden method in child class
    gs1.displayPerformance(includeTopic:true);
  }
}
```

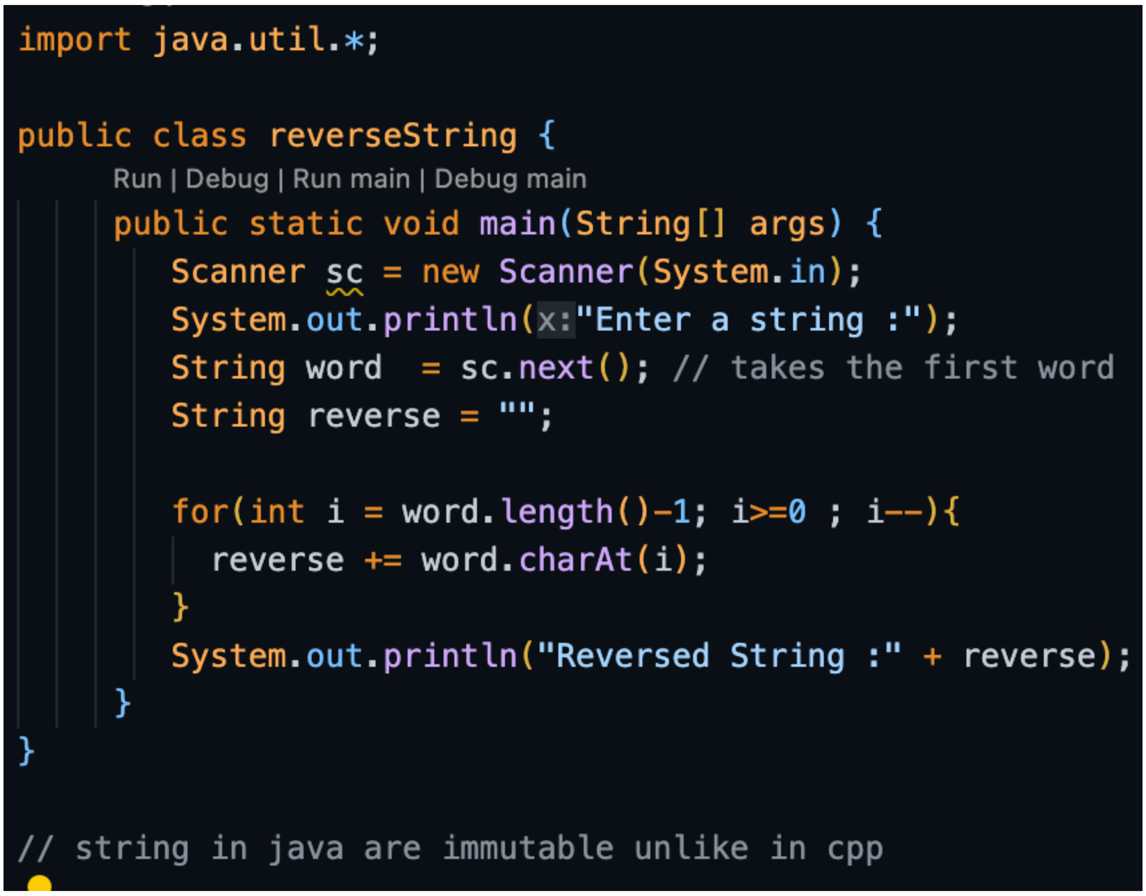
O/P Example



```
Student Performance :
Okayish Performance

Graduate Student Details:
Good Performance
Good Performance
Research Topic: Machine Learning
```

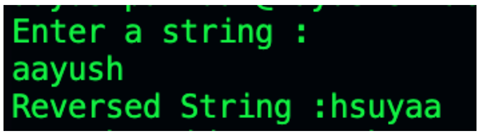
String Manipulation:

1.Write a program to reverse a given string.



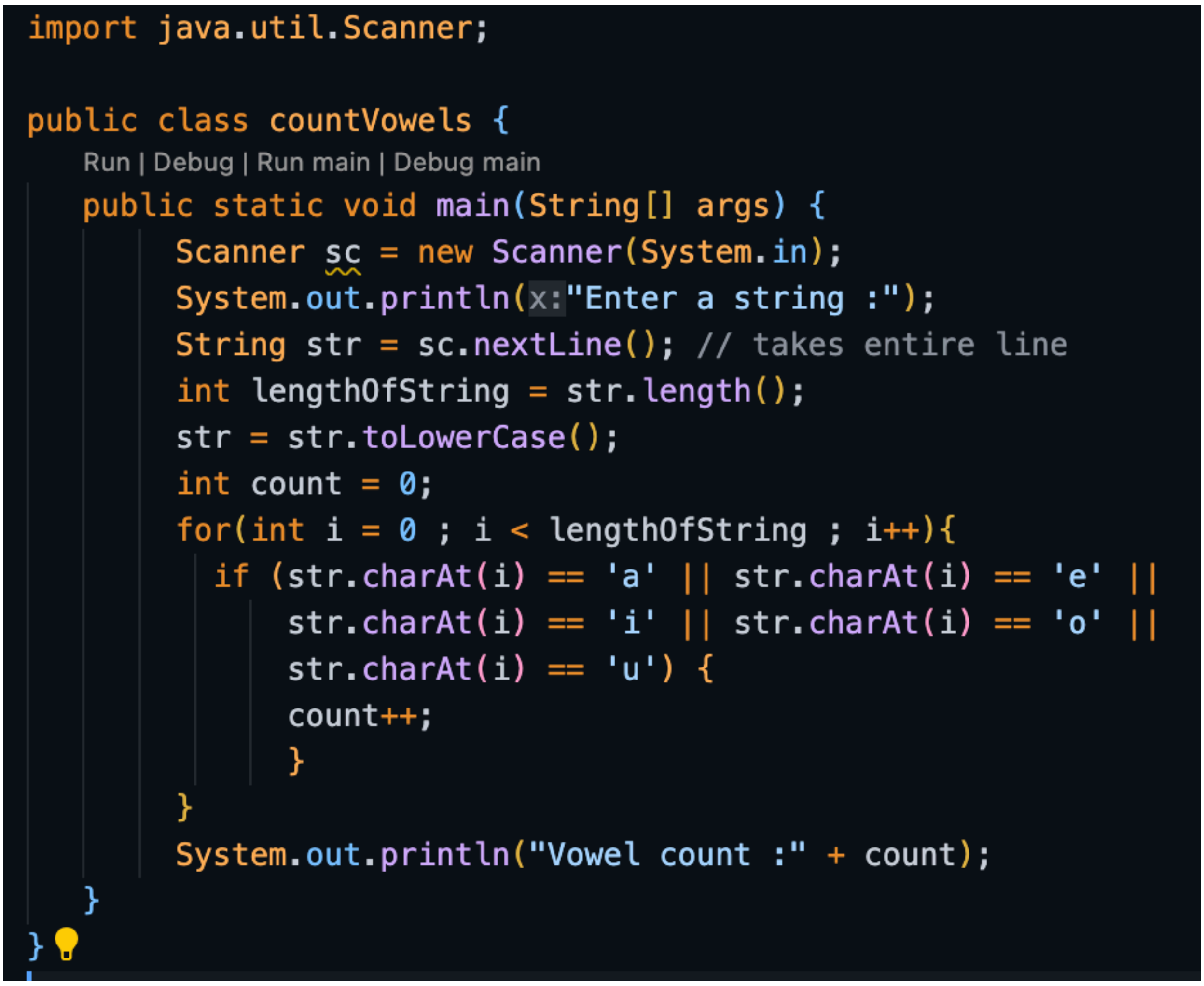
```java
import java.util.*;

public class reverseString {
    public static void main(String[] args) {
        Scanner sc = new Scanner(System.in);
        System.out.println(x:"Enter a string :");
        String word  = sc.next(); // takes the first word
        String reverse = "";

        for(int i = word.length()-1; i>=0 ; i--){
          reverse += word.charAt(i);
        }
        System.out.println("Reversed String :" + reverse);
    }
}

// string in java are immutable unlike in cpp
```

O/P Example



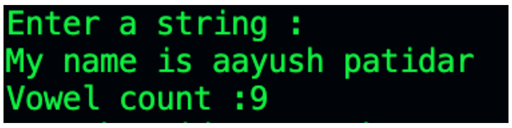
```
Enter a string :
aayush
Reversed String :hsuyaa
```

2.Implement a function to count the number of vowels in a string.



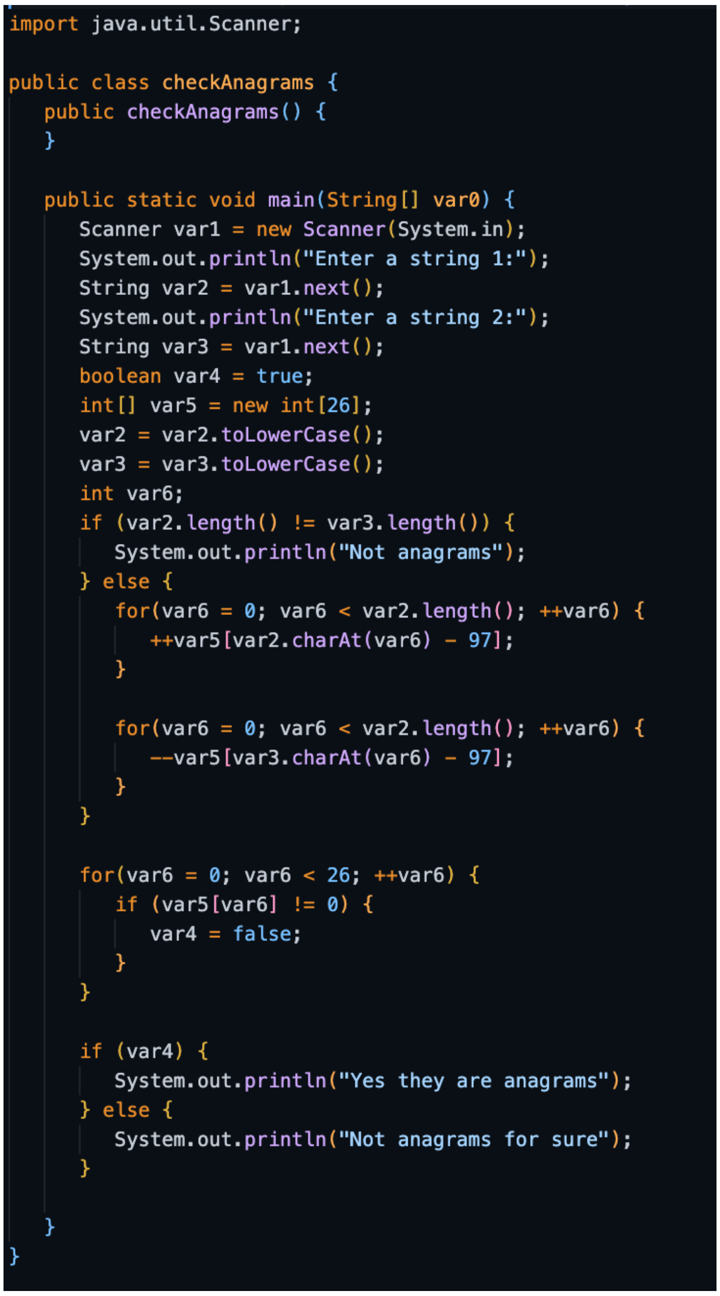
```java
import java.util.Scanner;

public class countVowels {
    Run | Debug | Run main | Debug main
    public static void main(String[] args) {
        Scanner sc = new Scanner(System.in);
        System.out.println(x:"Enter a string :");
        String str = sc.nextLine(); // takes entire line
        int lengthOfString = str.length();
        str = str.toLowerCase();
        int count = 0;
        for(int i = 0 ; i < lengthOfString ; i++){
          if (str.charAt(i) == 'a' || str.charAt(i) == 'e' ||
              str.charAt(i) == 'i' || str.charAt(i) == 'o' ||
              str.charAt(i) == 'u') {
              count++;
              }
        }
        System.out.println("Vowel count :" + count);
    }
}
```

O/P Example



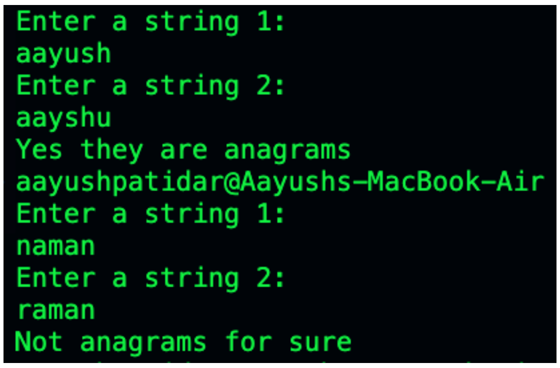
```
Enter a string :
My name is aayush patidar
Vowel count :9
```

3.Create a program to check if two strings are anagrams.



```java
import java.util.Scanner;

public class checkAnagrams {
   public checkAnagrams() {
   }

   public static void main(String[] var0) {
      Scanner var1 = new Scanner(System.in);
      System.out.println("Enter a string 1:");
      String var2 = var1.next();
      System.out.println("Enter a string 2:");
      String var3 = var1.next();
      boolean var4 = true;
      int[] var5 = new int[26];
      var2 = var2.toLowerCase();
      var3 = var3.toLowerCase();
      int var6;
      if (var2.length() != var3.length()) {
         System.out.println("Not anagrams");
      } else {
         for(var6 = 0; var6 < var2.length(); ++var6) {
            ++var5[var2.charAt(var6) - 97];
         }

         for(var6 = 0; var6 < var2.length(); ++var6) {
            --var5[var3.charAt(var6) - 97];
         }
      }

      for(var6 = 0; var6 < 26; ++var6) {
         if (var5[var6] != 0) {
            var4 = false;
         }
      }

      if (var4) {
         System.out.println("Yes they are anagrams");
      } else {
         System.out.println("Not anagrams for sure");
      }

   }
}
```

O/P Example



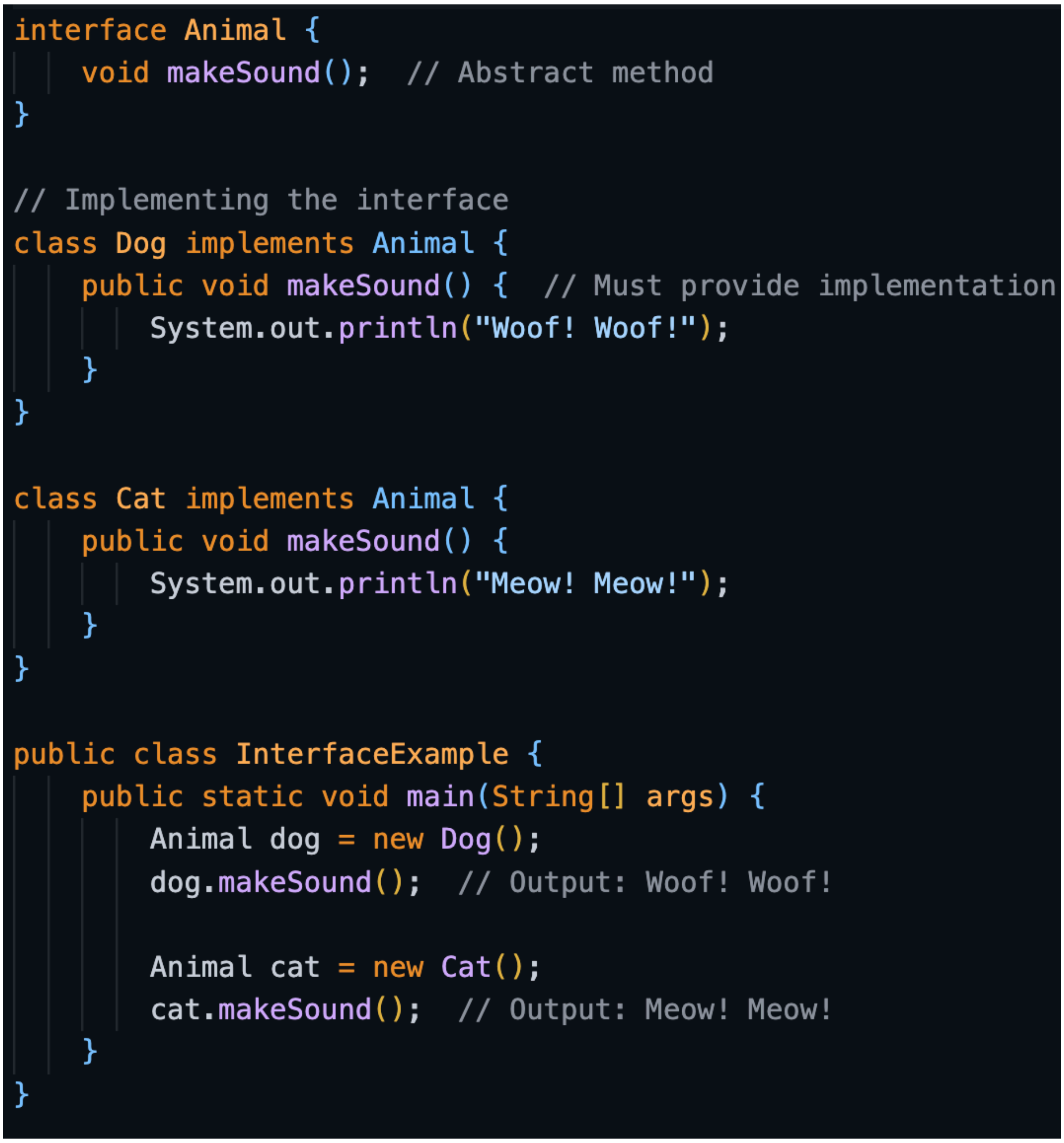
```
Enter a string 1:
aayush
Enter a string 2:
aayshu
Yes they are anagrams
aayushpatidar@Aayushs-MacBook-Air
Enter a string 1:
naman
Enter a string 2:
raman
Not anagrams for sure
```

Advanced Topics:

1.Explain the concept of interfaces and abstract classes with examples.

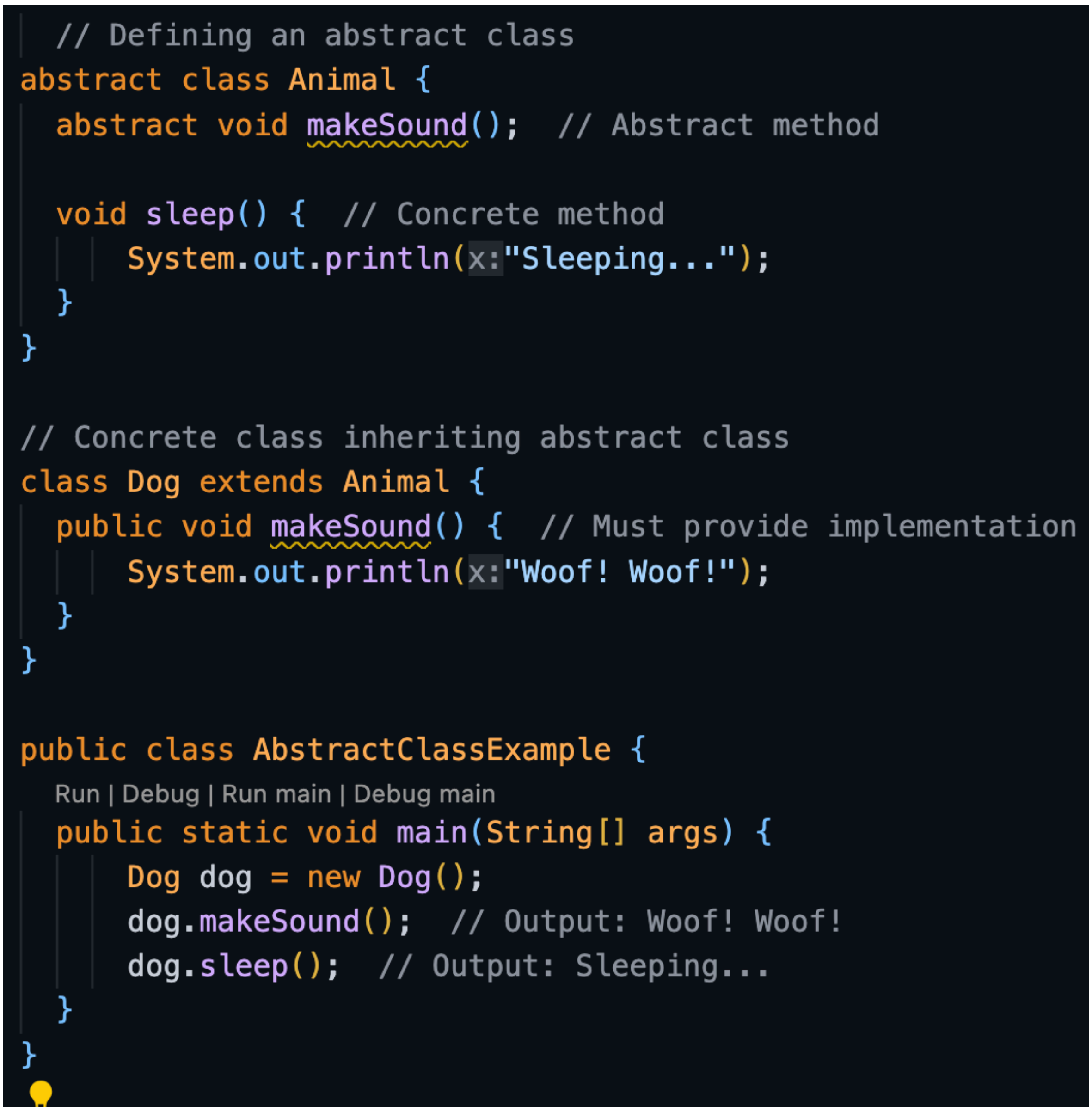
Both interfaces and abstract classes are used to achieve abstraction in Java.

Interfaces:
> supports multiple inheritance
>100 % abstraction
>does not contain concrete methods only abstract methods



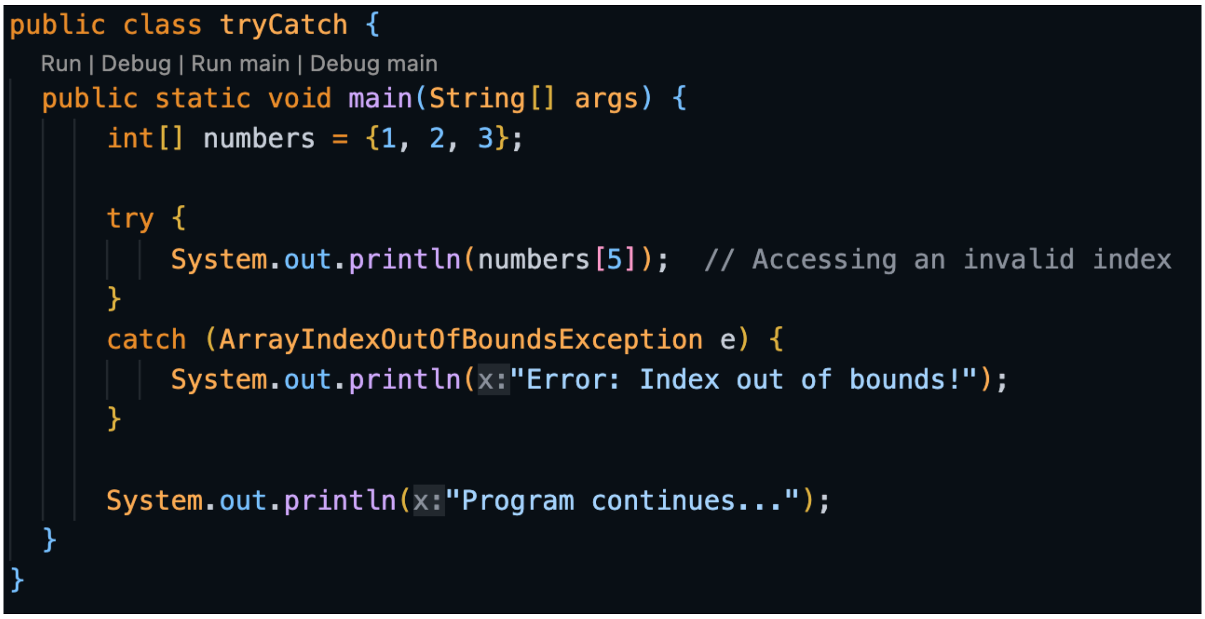
```java
interface Animal {
    void makeSound();  // Abstract method
}

// Implementing the interface
class Dog implements Animal {
    public void makeSound() {  // Must provide implementation
        System.out.println("Woof! Woof!");
    }
}

class Cat implements Animal {
    public void makeSound() {
        System.out.println("Meow! Meow!");
    }
}

public class InterfaceExample {
    public static void main(String[] args) {
        Animal dog = new Dog();
        dog.makeSound();  // Output: Woof! Woof!

        Animal cat = new Cat();
        cat.makeSound();  // Output: Meow! Meow!
    }
}
```

Abstract:
> does not support multiple inheritance
> contains both concrete and abstract methods



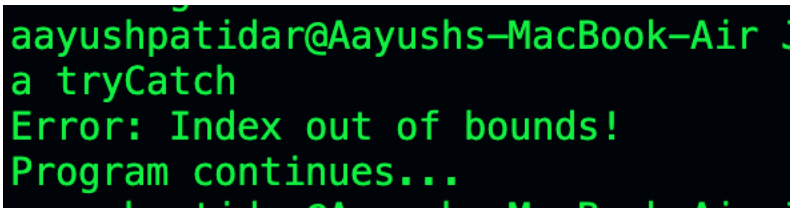
```java
// Defining an abstract class
abstract class Animal {
    abstract void makeSound();  // Abstract method

    void sleep() {  // Concrete method
        System.out.println(x:"Sleeping...");
    }
}

// Concrete class inheriting abstract class
class Dog extends Animal {
    public void makeSound() {  // Must provide implementation
        System.out.println(x:"Woof! Woof!");
    }
}

public class AbstractClassExample {
    Run | Debug | Run main | Debug main
    public static void main(String[] args) {
        Dog dog = new Dog();
        dog.makeSound();  // Output: Woof! Woof!
        dog.sleep();  // Output: Sleeping...
    }
}
```

2.Create a program to handle exceptions using try-catch blocks.



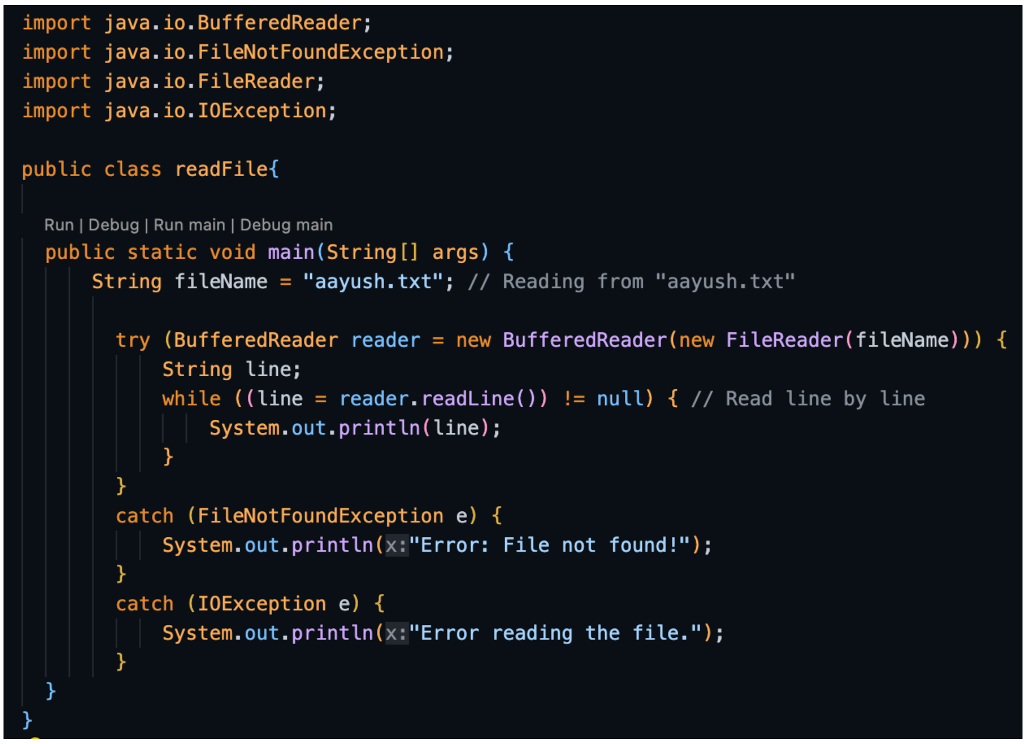
```java
public class tryCatch {
  Run | Debug | Run main | Debug main
  public static void main(String[] args) {
      int[] numbers = {1, 2, 3};

      try {
          System.out.println(numbers[5]);  // Accessing an invalid index
      }
      catch (ArrayIndexOutOfBoundsException e) {
          System.out.println(x:"Error: Index out of bounds!");
      }

      System.out.println(x:"Program continues...");
  }
}
```

O/P Example



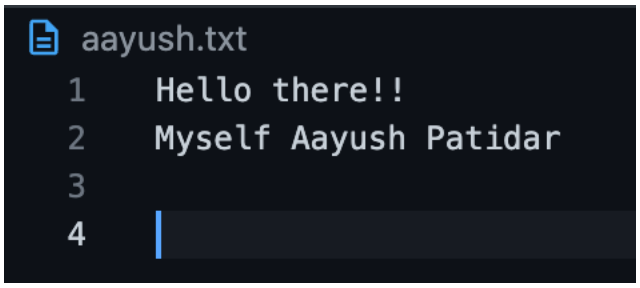
```
aayushpatidar@Aayushs-MacBook-Air
a tryCatch
Error: Index out of bounds!
Program continues...
```

3.Implement a simple file I/O operation to read data from a text file.



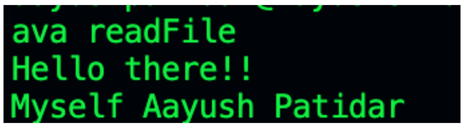
```java
import java.io.BufferedReader;
import java.io.FileNotFoundException;
import java.io.FileReader;
import java.io.IOException;

public class readFile{

  Run | Debug | Run main | Debug main
  public static void main(String[] args) {
      String fileName = "aayush.txt"; // Reading from "aayush.txt"

        try (BufferedReader reader = new BufferedReader(new FileReader(fileName))) {
            String line;
            while ((line = reader.readLine()) != null) { // Read line by line
                System.out.println(line);
            }
        }
        catch (FileNotFoundException e) {
            System.out.println(x:"Error: File not found!");
        }
        catch (IOException e) {
            System.out.println(x:"Error reading the file.");
        }
  }
}
```

Aayush.txt



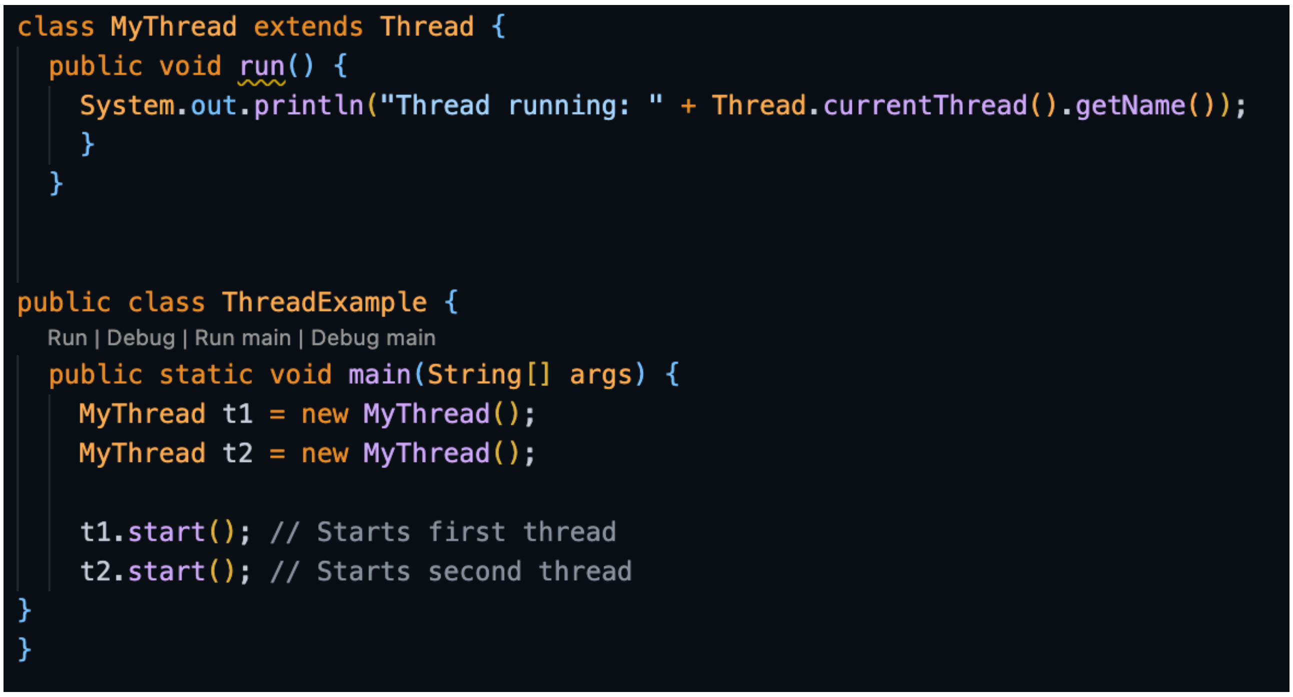
```
aayush.txt
Hello there!!
Myself Aayush Patidar

```

O/P Example



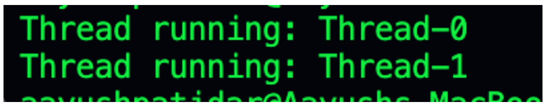
```
ava readFile
Hello there!!
Myself Aayush Patidar
```

4.Explore multithreading in Java to perform multiple tasks concurrently.



```java
class MyThread extends Thread {
    public void run() {
        System.out.println("Thread running: " + Thread.currentThread().getName());
        }
    }


public class ThreadExample {
    Run | Debug | Run main | Debug main
    public static void main(String[] args) {
        MyThread t1 = new MyThread();
        MyThread t2 = new MyThread();

        t1.start(); // Starts first thread
        t2.start(); // Starts second thread
}
}
```

O/P Example



```
Thread running: Thread-0
Thread running: Thread-1
```