Sara Fatma [Java Assignment-1]

Basic Java:

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

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

import java.util.*;
class Question1{
    public static void main(String []args){

        Scanner sc = new Scanner(System.in);

        System.out.println("Choose the shape to calculate area:");
        System.out.println("1. Circle");
        System.out.println("2. Rectangle");
        System.out.println("3. Triangle");
        System.out.print("Enter your choice (1/2/3): ");

        int choice = sc.nextInt();

       switch (choice) {
            case 1:
                System.out.print("Enter the radius of the circle: ");
                float radius = sc.nextFloat();
                float circleArea = 3.14f * radius * radius;
                System.out.println("Area of the circle:"+ circleArea);
                break;

            case 2:
                System.out.print("Enter the length of the rectangle: ");
                float length = sc.nextFloat();
                System.out.print("Enter the width of the rectangle: ");
                float width = sc.nextFloat();
                float rectangleArea = length * width;
                System.out.println("Area of the rectangle: "+ rectangleArea);
                break;

            case 3:
                System.out.print("Enter the base of the triangle: ");
                float base = sc.nextFloat();
                System.out.print("Enter the height of the triangle: ");
                float height = sc.nextFloat();
                float triangleArea = 0.5f * base * height;
                System.out.println("Area of the triangle: "+ triangleArea);
                break;

            default:
                System.out.println("Invalid choice. Please select 1, 2, or 3.");
        }

        sc.close();

    }
}
```

```
TERMINAL
PS D:\nucleusTeq\Training\java\JavaAssignment1-SaraFatma\BasicJavaQuestions> cd
Choose the shape to calculate area:
1. Circle
2. Rectangle
3. Triangle
Enter your choice (1/2/3): 1
Enter the radius of the circle: 4
Area of the circle:50.24
```

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

Question2.java

```java
//Create a program to check if a number is even or odd.

import java.util.*;
class Question2 {
    public static void main(String[] args) {
        Scanner sc = new Scanner(System.in);
        System.out.println("Enter a number to check if its even or odd: ");
        int number = sc.nextInt();

        if (number % 2 == 0){
            System.out.println("Number is even");
        }else{
            System.out.println("Number is odd");
        }

        sc.close();

    }
}
```

```
PROBLEMS  OUTPUT  DEBUG CONSOLE  TERMINAL
TERMINAL
PS D:\nucleusTeq\Training\java\JavaAssignment1-SaraFatma\BasicJavaQuestions> cd "d:\nucleusTeq\
Enter a number to check if its even or odd:
3
Number is odd
PS D:\nucleusTeq\Training\java\JavaAssignment1-SaraFatma\BasicJavaQuestions>
```

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

Question3.java

```java
// Implement a program to find the factorial of a given number.
import java.util.*;


public class Question3 {

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

        System.out.print("Enter a number whose factorial is to be printed: ");
        int number = sc.nextInt();
        long factorial = 1;

        for(int i = 1; i <= number; i++){
           factorial = factorial * i;
        }

        System.out.println("Factorial of " + number + " is " + factorial);
        sc.close();
    }
}
```

```
PROBLEMS  OUTPUT  DEBUG CONSOLE  TERMINAL
TERMINAL
PS D:\nucleusTeq\Training\java\JavaAssignment1-SaraFatma\BasicJavaQuestions> cd "d:\nucleusTeq\T
Enter a number whose factorial is to be printed: 5
Factorial of 5 is 120
PS D:\nucleusTeq\Training\java\JavaAssignment1-SaraFatma\BasicJavaQuestions>
```

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

Question4.java

```java
//Write a program to print the Fibonacci sequence up to a specified number.

import java.util.Scanner;

public class Question4 {
    public static void main(String[] args) {
        Scanner sc = new Scanner(System.in);
        System.out.print("Enter the number of terms: ");
        int n = sc.nextInt();
        sc.close();

        System.out.print("Fibonacci Series: ");
        for (int i = 0; i < n; i++) {
            System.out.print(fibonacci(i) + " ");
        }
    }

    public static int fibonacci(int n) {
        if (n <= 1)
            return n;
        return fibonacci(n - 1) + fibonacci(n - 2);
    }
}
```

```
PROBLEMS  OUTPUT  DEBUG CONSOLE  TERMINAL
TERMINAL
PS D:\nucleusTeq\Training\java\JavaAssignment1-SaraFatma\BasicJavaQuestions> cd "d:\nucleusTeq\Tra
Enter the number of terms: 10
Fibonacci Series: 0 1 1 2 3 5 8 13 21 34
PS D:\nucleusTeq\Training\java\JavaAssignment1-SaraFatma\BasicJavaQuestions>
```

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

Question5.java

```java
//  Use loops to print patterns like a triangle or square.
import java.util.*;
public class Question5
{
    public static void main(String[] args) {
        Scanner sc = new Scanner(System.in);
        System.out.print("Enter the size of the square: ");
        int n = sc.nextInt();
        sc.close();

        for (int i = 0; i < n; i++) {
            for (int j = 0; j < n; j++) {
                System.out.print("* ");
            }
            System.out.println();
        }
    }
}
```

```
PROBLEMS  OUTPUT  DEBUG CONSOLE  TERMINAL
TERMINAL
PS D:\nucleusTeq\Training\java\JavaAssignment1-SaraFatma\BasicJavaQuestions>
Enter the size of the square: 5
* * * * *
* * * * *
* * * * *
* * * * *
* * * * *
```

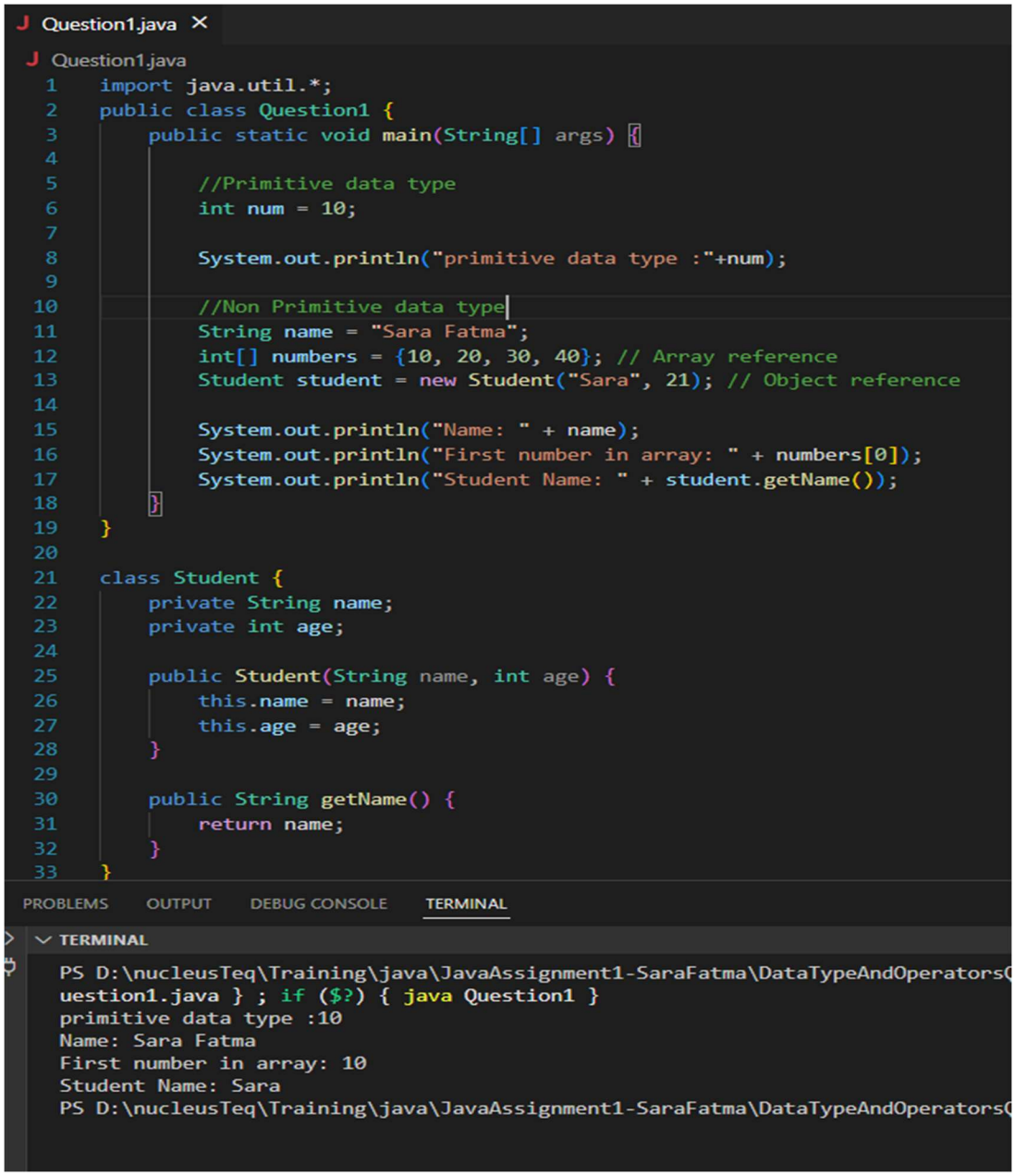
Data Types and Operators:

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

Feature	**Primitive Data Type**	**Reference Data Type**
Storage	Stores actual value	Stores memory address (reference)
Memory Location	Stored in **stack**	Object in **heap**, reference in stack
Default Value	0, false, '\u0000'	null
Methods Available?	No methods	Methods available

Feature	Primitive Data Type	Reference Data Type
Mutability	Immutable	Mutable (modifiable object properties)
Performance	Faster (direct access)	Slower (memory access via reference)

Example of Primitive DataType



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

Question2.java

```java
import java.util.*;

public class Question2 {
    public static void main(String[] args) {
        int[] arr = {1, 2, 3, 4, 5,
        // Arithmetic Operators
        int a = 10, b = 5;
        System.out.println("Arithmetic Operators:");
        System.out.println("a + b = " + (a + b)); // Addition
        System.out.println("a - b = " + (a - b)); // Subtraction
        System.out.println("a * b = " + (a * b)); // Multiplication
        System.out.println("a / b = " + (a / b)); // Division
        System.out.println("a % b = " + (a % b)); // Modulus

        // Relational Operators
        System.out.println("\nRelational Operators:");
        System.out.println("a == b: " + (a == b)); // Equal to
        System.out.println("a != b: " + (a != b)); // Not equal to
        System.out.println("a > b: " + (a > b)); // Greater than
        System.out.println("a < b: " + (a < b)); // Less than
        System.out.println("a >= b: " + (a >= b)); // Greater than or equal to
        System.out.println("a <= b: " + (a <= b)); // Less than or equal to

        // Logical Operators
        boolean x = true, y = false;
        System.out.println("\nLogical Operators:");
        System.out.println("x && y: " + (x && y)); // Logical AND
        System.out.println("x || y: " + (x || y)); // Logical OR
        System.out.println("!x: " + (!x)); // Logical NOT
    }
}
```

TERMINAL

```
PS D:\nucleusTeq\Training\java\JavaAssignment1-SaraFatma\DataTypeAndOperatorsQuestions> cd "d
uestion2.java } ; if ($?) { java Question2 }
Arithmetic Operators:
a + b = 15
a - b = 5
a * b = 50
a / b = 2
a % b = 0

Relational Operators:
a == b: false
a != b: true
a > b: true
a < b: false
a >= b: true
a <= b: false

Logical Operators:
x && y: false
x || y: true
!x: false
PS D:\nucleusTeq\Training\java\JavaAssignment1-SaraFatma\DataTypeAndOperatorsQuestions>
```

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

Question3.java

```java
import java.util.*;

public class Question3 {
    public static void main(String[] args) {

        Scanner sc = new Scanner(System.in);

        System.out.println("Temperature Converter");
        System.out.println("1: Celsius to Fahrenheit");
        System.out.println("2: Fahrenheit to Celsius");
        System.out.print("Choose an option (1 or 2): ");
        int choice = sc.nextInt();

        if (choice == 1) {
            System.out.print("Enter temperature in Celsius: ");
            float celsius = sc.nextFloat();
            float fahrenheit = (celsius * 9/5) + 32;
            System.out.println("Temperature in Fahrenheit: " + fahrenheit);
        } else if (choice == 2) {
            System.out.print("Enter temperature in Fahrenheit: ");
            float fahrenheit = sc.nextFloat();
            float celsius = (fahrenheit - 32) * 5/9;
            System.out.println("Temperature in Celsius: " + celsius);
        } else {
            System.out.println("Invalid choice!");
        }

        sc.close();
    }
}

```

PROBLEMS OUTPUT DEBUG CONSOLE TERMINAL

TERMINAL

```
Enter temperature in Celsius: 20
Temperature in Fahrenheit: 68.0
PS D:\nucleusTeq\Training\java\JavaAssignment1-SaraFatma\DataTypeAndOperatorsQuestions> cd "d:\nucleusTeq
uestion3.java } ; if ($?) { java Question3 }
Temperature Converter
1: Celsius to Fahrenheit
2: Fahrenheit to Celsius
Choose an option (1 or 2): 2
Enter temperature in Fahrenheit: 68
Temperature in Celsius: 20.0
PS D:\nucleusTeq\Training\java\JavaAssignment1-SaraFatma\DataTypeAndOperatorsQuestions>
```

Control Flow Statements:

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

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

Question2.java

```java
import java.util.*;

public class Question2 {
    public static void main(String[] args) {
        Scanner sc = new Scanner(System.in);

        System.out.print("Enter first number: ");
        int num1 = sc.nextInt();

        System.out.print("Enter second number: ");
        int num2 = sc.nextInt();

        System.out.print("Enter third number: ");
        int num3 = sc.nextInt();

        sc.close();

        int largest;

        if (num1 >= num2 && num1 >= num3) {
            largest = num1;
        } else if (num2 >= num1 && num2 >= num3) {
            largest = num2;
        } else {
            largest = num3;
        }

        System.out.println("The largest number is: " + largest);
    }
}
```

PROBLEMS OUTPUT DEBUG CONSOLE TERMINAL

TERMINAL Code

```
Enter second number: 111
Enter third number: 4
The largest number is: 222
PS D:\nucleusTeq\Training\java\JavaAssignment1-SaraFatma\ControlFlowStat
ns> cd "d:\nucleusTeq\Training\java\JavaAssignment1-SaraFatma\ControlFlo
estions\" ; if ($?) { javac Question2.java } ; if ($?) { java Question2
Enter first number: -1
Enter second number: -3
Enter third number: 2
The largest number is: 2
```

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

Question3.java

```java
import java.util.*;

public class Question3 {
    public static void main(String[] args) {
        Scanner sc = new Scanner(System.in);
        System.out.print("Enter a number: ");
        int num = sc.nextInt();

        System.out.println("Multiplication Table of " + num);
        for (int i = 1; i <= 10; i++) {
            System.out.println(num + " x " + i + " = " + (num * i));
        }

        sc.close();
    }
}
```

PROBLEMS OUTPUT DEBUG CONSOLE TERMINAL

TERMINAL

```
9 x 1 = 9
9 x 2 = 18
9 x 3 = 27
9 x 4 = 36
9 x 5 = 45
9 x 6 = 54
9 x 7 = 63
9 x 8 = 72
9 x 9 = 81
9 x 10 = 90
PS D:\nucleusTeq\Training\java\JavaAssignment1-SaraFatma\ControlFlow
StatementQuestions>
```

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

Question4.java

```java
import java.util.*;
public class Question4 {
    public static void main(String[] args) {
        int num = 2, sum = 0;

        while (num <= 10) {
            sum += num;
            num += 2;
        }

        System.out.println("Sum of even numbers from 1 to 10: " + sum);
    }
}
```

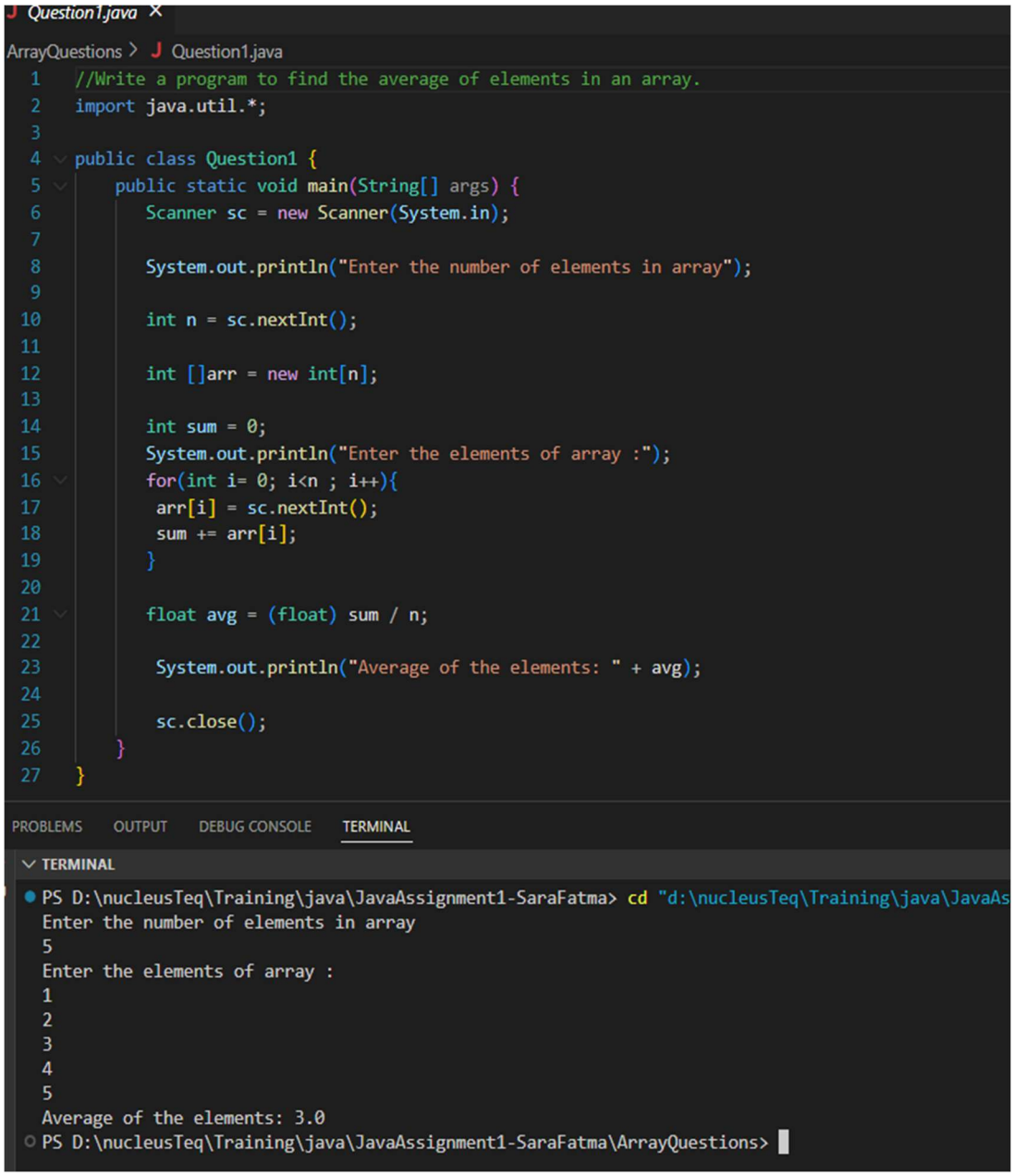
ROBLEMS OUTPUT DEBUG CONSOLE TERMINAL

TERMINAL Code

```
PS D:\nucleusTeq\Training\java\JavaAssignment1-SaraFatma\ControlFlowStatementQ
ns> cd "d:\nucleusTeq\Training\java\JavaAssignment1-SaraFatma\ControlFlowState
estions\" ; if ($?) { javac Question4.java } ; if ($?) { java Question4 }
Sum of even numbers from 1 to 10: 30
```

Arrays:

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



```java
//Write a program to find the average of elements in an array.
import java.util.*;

public class Question1 {
    public static void main(String[] args) {
        Scanner sc = new Scanner(System.in);

        System.out.println("Enter the number of elements in array");

        int n = sc.nextInt();

        int []arr = new int[n];

        int sum = 0;
        System.out.println("Enter the elements of array :");
        for(int i= 0; i<n ; i++){
         arr[i] = sc.nextInt();
         sum += arr[i];
        }

        float avg = (float) sum / n;

         System.out.println("Average of the elements: " + avg);

         sc.close();
    }
}
```

```
PS D:\nucleusTeq\Training\java\JavaAssignment1-SaraFatma> cd "d:\nucleusTeq\Training\java\JavaAs
Enter the number of elements in array
5
Enter the elements of array :
1
2
3
4
5
Average of the elements: 3.0
PS D:\nucleusTeq\Training\java\JavaAssignment1-SaraFatma\ArrayQuestions>
```

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

Question2.java

ArrayQuestions > Question2.java

```java
//implement a function to sort an array in ascending order using bubble sort
import java.util.*;

public class Question2 {
    public static void bubbleSort(int[] arr) {
        int n = arr.length;
        boolean swapped;

        for (int i = 0; i < n - 1; i++) {
            swapped = false;
            for (int j = 0; j < n - 1 - i; j++) {
                if (arr[j] > arr[j + 1]) {
                    // Swap arr[j] and arr[j+1]
                    int temp = arr[j];
                    arr[j] = arr[j + 1];
                    arr[j + 1] = temp;
                    swapped = true;
                }
            }
            // If no two elements were swapped in the inner loop, the array is sorted
            if (!swapped) break;
        }
    }

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

        System.out.print("Enter the number of elements: ");
        int n = sc.nextInt();

        int[] arr = new int[n];

        System.out.println("Enter the elements:");
        for (int i = 0; i < n; i++) {
            arr[i] = sc.nextInt();
        }

        bubbleSort(arr);

        System.out.println("Sorted array in ascending order:");
        for (int num : arr) {
            System.out.print(num + " ");
        }

        sc.close();
    }
}
```

PROBLEMS OUTPUT DEBUG CONSOLE TERMINAL

TERMINAL

```
PS D:\nucleusTeq\Training\java\JavaAssignment1-SaraFatma> cd "d:\nucleusTeq\Training\j
Enter the number of elements: 10
Enter the elements:
12
3
111
765
233
0
-3
55
234
975
Sorted array in ascending order:
-3 0 3 12 55 111 233 234 765 975
PS D:\nucleusTeq\Training\java\JavaAssignment1-SaraFatma\ArrayQuestions>
```

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

Question3.java

ArrayQuestions > Question3.java

```java
//Create a program to search for a specific element within an array using linear search

import java.util.*;

public class Question3 {
    public static int linearSearch(int[] arr, int key) {
        for (int i = 0; i < arr.length; i++) {
            if (arr[i] == key) {
                return i;
            }
        }
        return -1;
    }

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

        System.out.print("Enter the number of elements: ");
        int n = sc.nextInt();

        int[] arr = new int[n];

        System.out.println("Enter the elements:");
        for (int i = 0; i < n; i++) {
            arr[i] = sc.nextInt();
        }

        System.out.print("Enter the element to search: ");
        int key = sc.nextInt();

        int index = linearSearch(arr, key);

        if (index != -1) {
            System.out.println("Element found at index: " + index);
        } else {
            System.out.println("Element not found in the array.");
        }

        sc.close();
    }
}
```

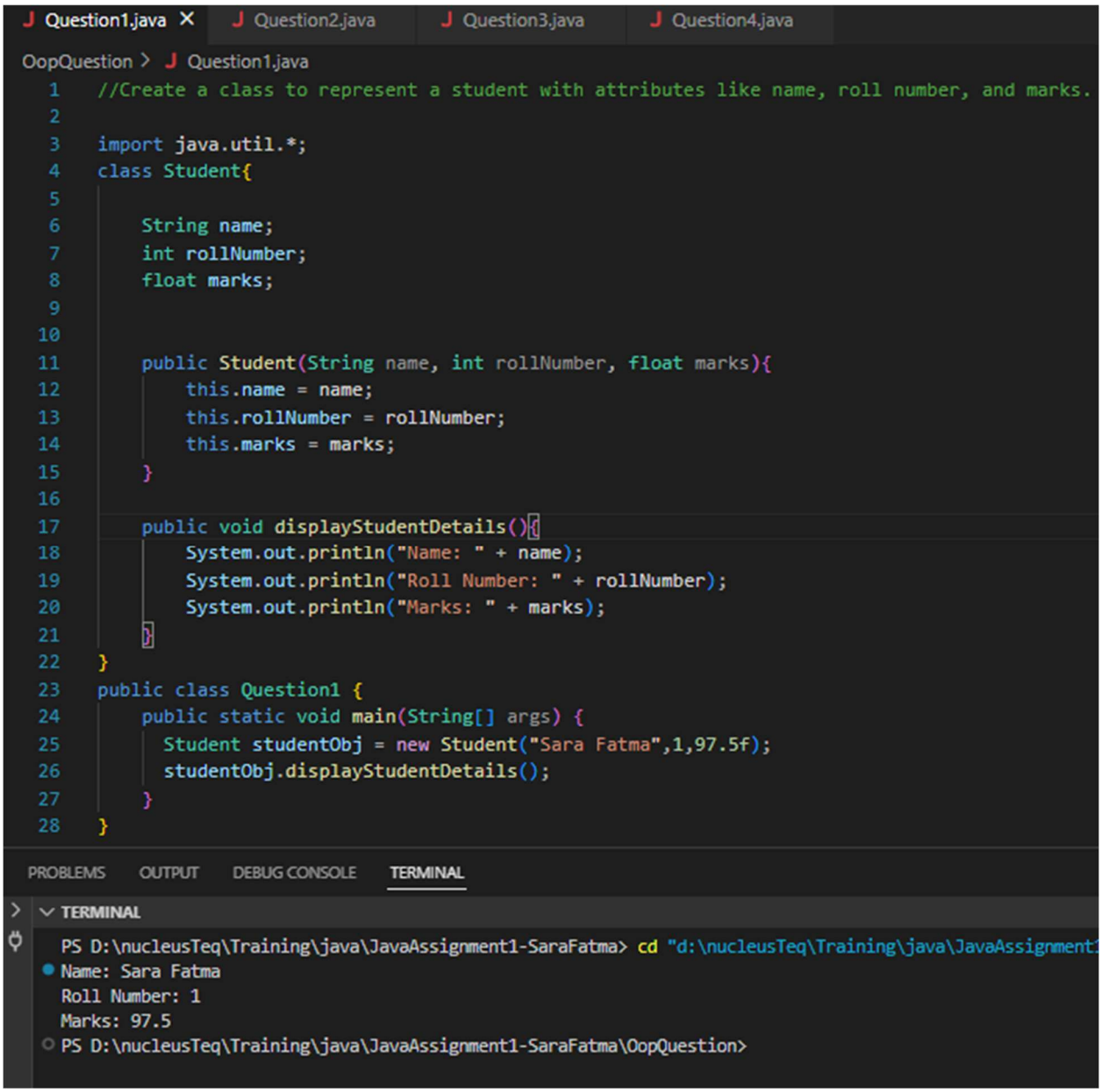
PROBLEMS OUTPUT DEBUG CONSOLE TERMINAL

TERMINAL

```
PS D:\nucleusTeq\Training\java\JavaAssignment1-SaraFatma> cd "d:\nucleus
Enter the number of elements: 5
Enter the elements:
12
75
7
90
23
Enter the element to search: 75
Element found at index: 1
PS D:\nucleusTeq\Training\java\JavaAssignment1-SaraFatma\ArrayQuestions
```

Object Oriented Programming (OOP):

1. Create a class to represent a student with attributes like name, roll number, and marks.



2. Implement inheritance to create a "GraduateStudent" class that extends the "Student" class with additional features.

Question1.java | Question2.java | Question3.java | Question4.java

OopQuestion > Question2.java

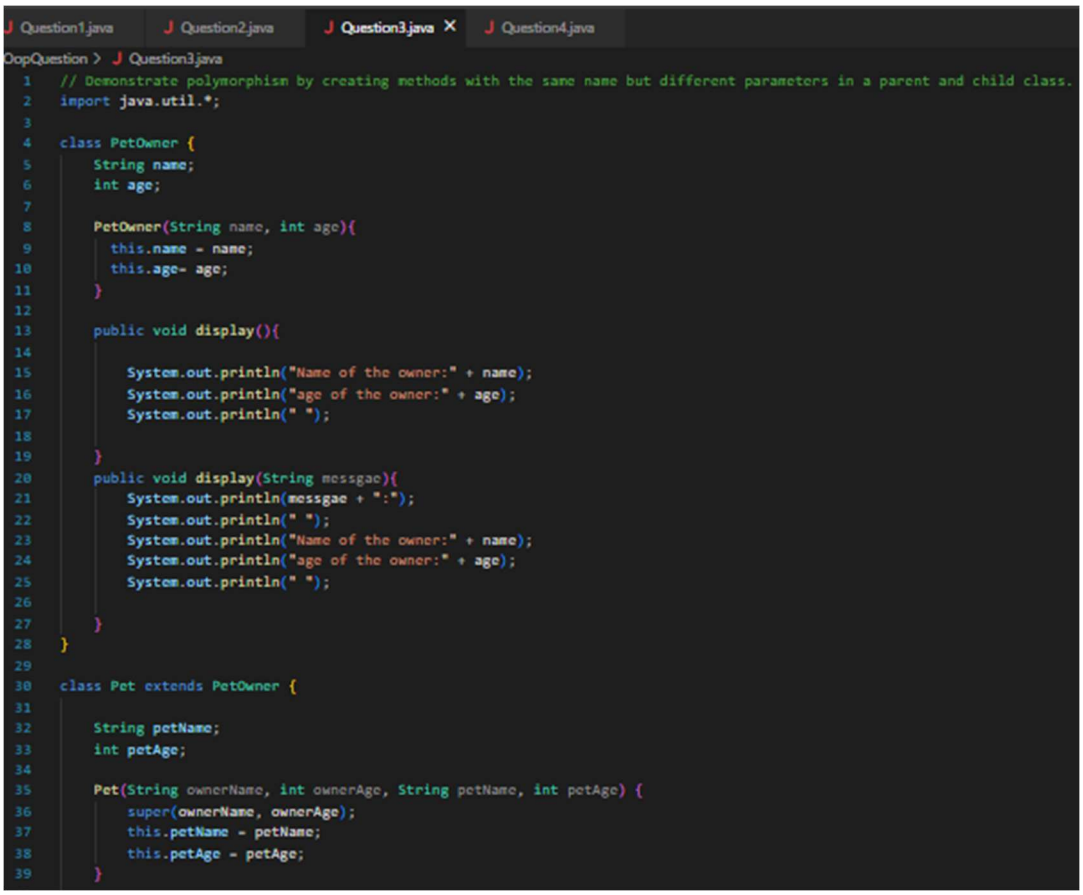
```java
import java.util.*;

class Student {
    String name;
    int rollNumber;
    float marks;

    Student(String name, int rollNumber, float marks) {
        this.name = name;
        this.rollNumber = rollNumber;
        this.marks = marks;
    }

    public void displayStudentDetails() {
        System.out.println("Name: " + name);
        System.out.println("Roll Number: " + rollNumber);
        System.out.println("Marks: " + marks);
    }
}

class GraduateStudent extends Student {
    String course;

    GraduateStudent(String name, int rollNumber, float marks, String course) {
        super(name, rollNumber, marks);
        this.course = course;
    }

    @Override
    public void displayStudentDetails() {
        super.displayStudentDetails();
        System.out.println("Course: " + course);
    }

    }

class Question2 {
    public static void main(String[] args) {
        GraduateStudent gradStudent = new GraduateStudent("Sara Fatma", 1, 97.5f, "AI & ML");
        gradStudent.displayStudentDetails();
    }
}
```

PROBLEMS | OUTPUT | DEBUG CONSOLE | TERMINAL

TERMINAL

```
PS D:\nucleusTeq\Training\java\JavaAssignment1-SaraFatma\OopQuestion> cd "d:\nucleusTeq\Training\java\JavaAssign
Name: Sara Fatma
Roll Number: 1
Marks: 97.5
Course: AI & ML
PS D:\nucleusTeq\Training\java\JavaAssignment1-SaraFatma\OopQuestion>
```

3. Demonstrate polymorphism by creating methods with the same name but different parameters in a parent and child class.



```java
// Demonstrate polymorphism by creating methods with the same name but different parameters in a parent and child class.
import java.util.*;

class PetOwner {
    String name;
    int age;

    PetOwner(String name, int age){
      this.name = name;
      this.age= age;
    }

    public void display(){

        System.out.println("Name of the owner:" + name);
        System.out.println("age of the owner:" + age);
        System.out.println(" ");

    }
    public void display(String messgae){
        System.out.println(messgae + ":");
        System.out.println(" ");
        System.out.println("Name of the owner:" + name);
        System.out.println("age of the owner:" + age);
        System.out.println(" ");

    }
}

class Pet extends PetOwner {

    String petName;
    int petAge;

    Pet(String ownerName, int ownerAge, String petName, int petAge) {
        super(ownerName, ownerAge);
        this.petName = petName;
        this.petAge = petAge;
    }
```

```java
    @Override
    public void display(){

        System.out.println("Name of the owner:" + name);
        System.out.println("age of the owner:" + age);
        System.out.println("Name of the pet:" + petName);
        System.out.println("age of the pet:" + petAge);

    }

    public void display(String messgae){
        System.out.println(messgae + ":");
        System.out.println(" ");
        System.out.println("Name of the owner:" + name);
        System.out.println("age of the owner:" + age);
        System.out.println("Name of the pet:" + petName);
        System.out.println("age of the pet:" + petAge);

    }
}

public class Question3 {
    public static void main(String[] args) {

        PetOwner ownerObj1 = new PetOwner("John", 25);
        ownerObj1.display("Overloading display() in class PetOwner");

        System.out.println("After Overriding display() method in Pet class");
        System.out.println(" ");

        Pet petObj1 = new Pet("John", 25, "Buddy", 2);
        petObj1.display("Overloading Display() method in Pet class");
    }
}
```

```
PROBLEMS  OUTPUT  DEBUG CONSOLE  TERMINAL
TERMINAL
PS D:\nucleusTeq\Training\java\JavaAssignment1-SaraFatma> cd "d:\nucleusT
Overloading display() in class PetOwner:

Name of the owner:John
age of the owner:25

After Overriding display() method in Pet class

Overloading Display() method in Pet class:

Name of the owner:John
age of the owner:25
Name of the pet:Buddy
age of the pet:2
```

4. Explain the concept of encapsulation with a suitable example.

Question1.java | Question2.java | Question3.java | Question4.java

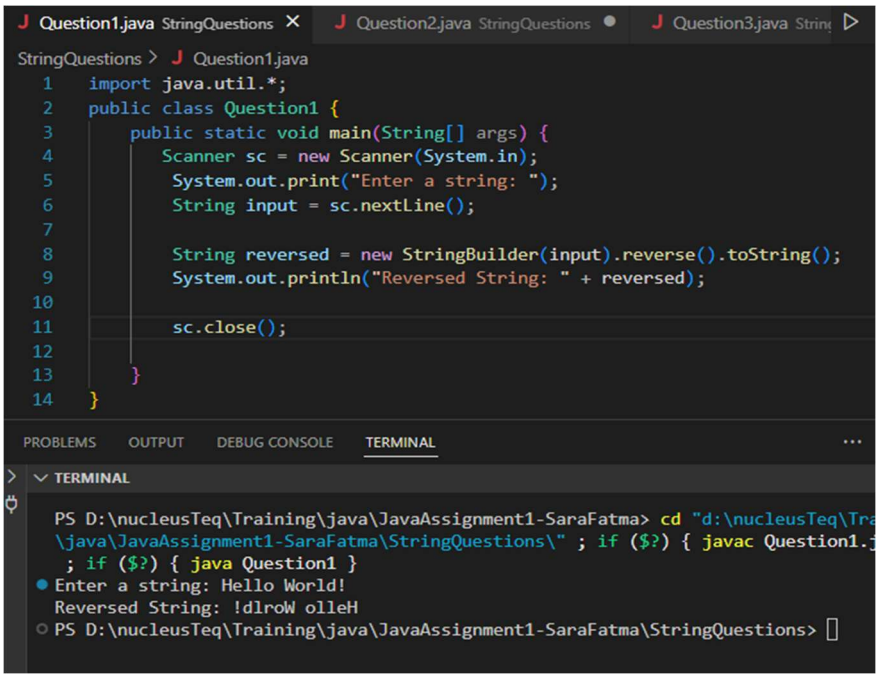
OopQuestion > Question4.java

```java
//Explain the concept of encapsulation with a suitable example.
import java.util.*;

class Person {
    // (data hiding)
    private String name;


    public void setName(String name) {
        this.name = name;
    }


    public String getName() {
        return name;
    }
}
public class Question4 {
    public static void main(String[] args) {
        Person p = new Person();
        p.setName("Sara Fatma");
        System.out.println("Name: " + p.getName());
    }
}
```

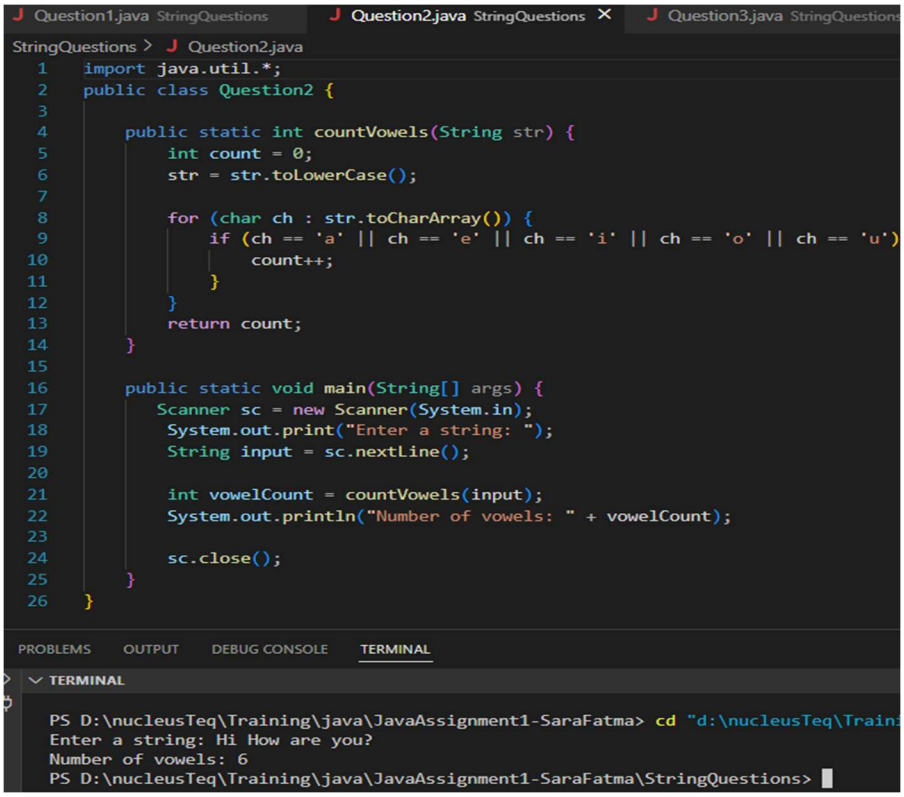
```
PROBLEMS  OUTPUT  DEBUG CONSOLE  TERMINAL
TERMINAL
PS D:\nucleusTeq\Training\java\JavaAssignment1-SaraFatma> cd "d:\nucleusTeq\Training\
Name: Sara Fatma
PS D:\nucleusTeq\Training\java\JavaAssignment1-SaraFatma\OopQuestion>
```

String Manipulation:

1. Write a program to reverse a given string.



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



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

Question1.java StringQuestions | Question2.java StringQuestions | Question3.java StringQuestions

StringQuestions > Question3.java

```java
import java.util.*;
public class Question3 {
    public static boolean areAnagrams(String str1, String str2) {
        str1 = str1.replaceAll("\\s", "").toLowerCase();
        str2 = str2.replaceAll("\\s", "").toLowerCase();
        if (str1.length() != str2.length()) {
            return false;
        }
        char[] arr1 = str1.toCharArray();
        char[] arr2 = str2.toCharArray();
        Arrays.sort(arr1);
        Arrays.sort(arr2);
        return Arrays.equals(arr1, arr2);
    }
    public static void main(String[] args) {
        Scanner sc = new Scanner(System.in);
        System.out.print("Enter first string: ");
        String str1 = sc.nextLine();
        System.out.print("Enter second string: ");
        String str2 = sc.nextLine();
        if (areAnagrams(str1, str2)) {
            System.out.println("The strings are anagrams.");
        } else {
            System.out.println("The strings are NOT anagrams.");
        }

        sc.close();
    }
}
```

PROBLEMS OUTPUT DEBUG CONSOLE TERMINAL

TERMINAL

```
PS D:\nucleusTeq\Training\java\JavaAssignment1-SaraFatma\StringQuestions> cd "d:\nucleusTeq\Tra
estion3 }
Enter first string: arc
Enter second string: car
The strings are anagrams.
PS D:\nucleusTeq\Training\java\JavaAssignment1-SaraFatma\StringQuestions>
```

Advanced Topics:

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

Both **interfaces** and **abstract classes** are used for abstraction in Java, but they have key differences in how they work and are used.

1. Abstract Class (Partial Abstraction)

An **abstract class**:

- Can have **both abstract (unimplemented) and concrete (implemented) methods**.
- Can have **constructors, instance variables, and static methods**.
- Supports **inheritance (extends keyword)**.
- Cannot be instantiated directly.

2. Interface (Full Abstraction)

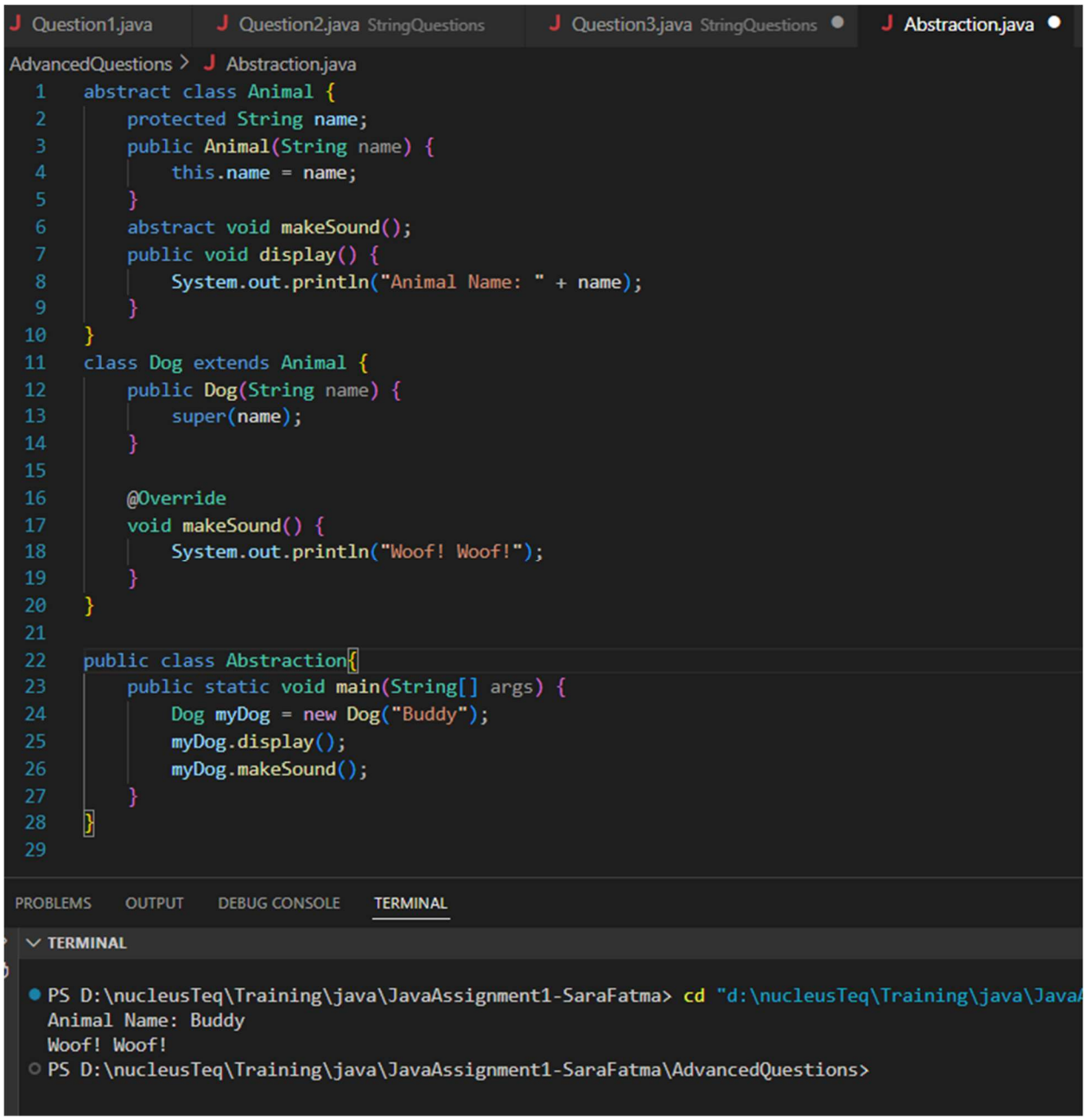
An **interface**:

- **Only contains abstract methods (before Java 8)**.
- Can contain **default methods and static methods** (from Java 8).
- **Cannot have constructors or instance variables**.
- Supports **multiple inheritance** (unlike abstract classes).

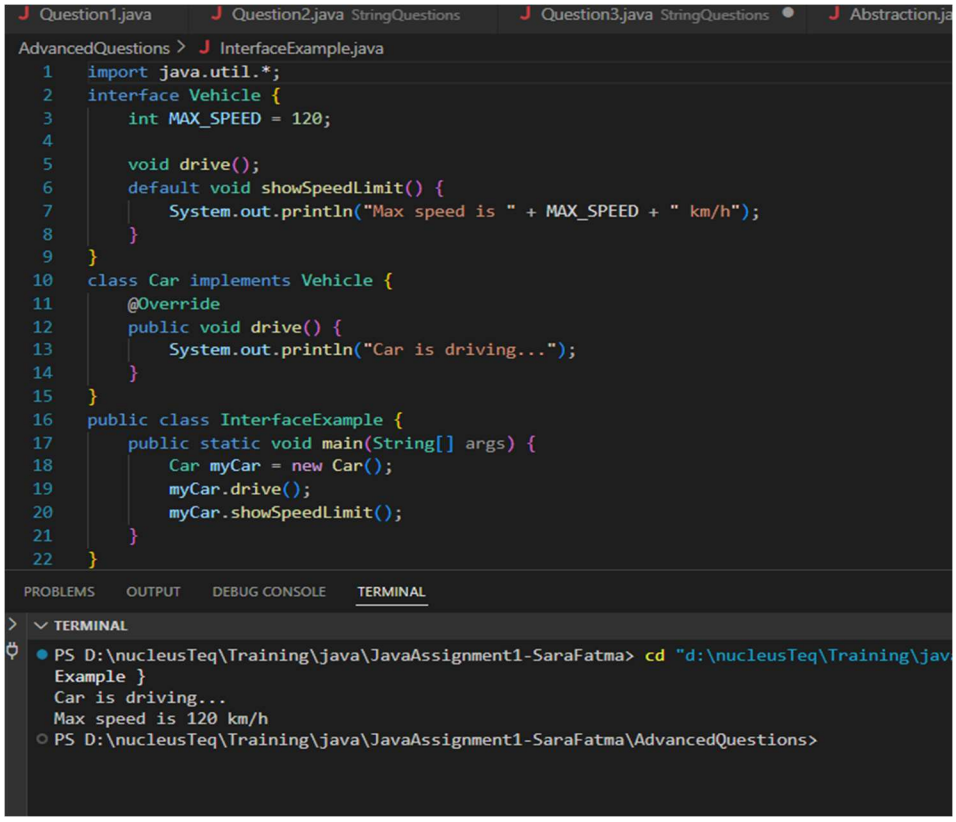
Abstract Class vs Interface:

Feature	Abstract Class	Interface
Methods	Can have both abstract & concrete methods	Only abstract methods (before Java 8)
Variables	Can have instance variables	Only public, static, and final constants
Constructors	Can have constructors	Cannot have constructors
Multiple Inheritance	Not supported	Supported
Access Modifiers	Methods can be `public`, `protected`, or `private`	Methods are `public` by default
Use Case	Used when classes share **common behavior**	Used for **defining a contract** that multiple classes can follow

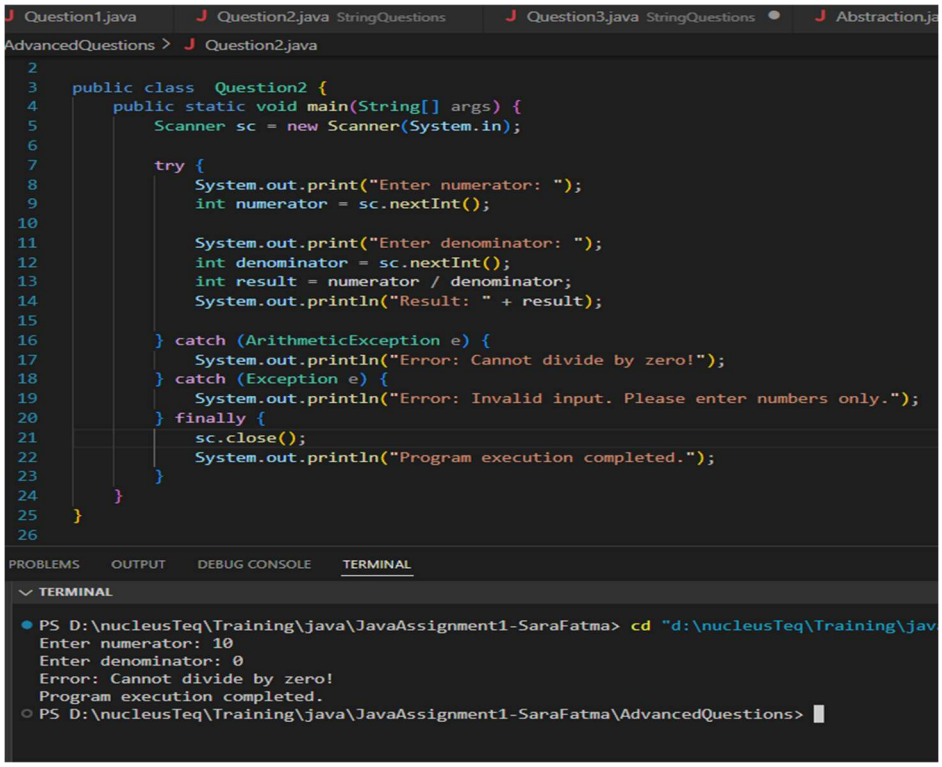
Abstraction example



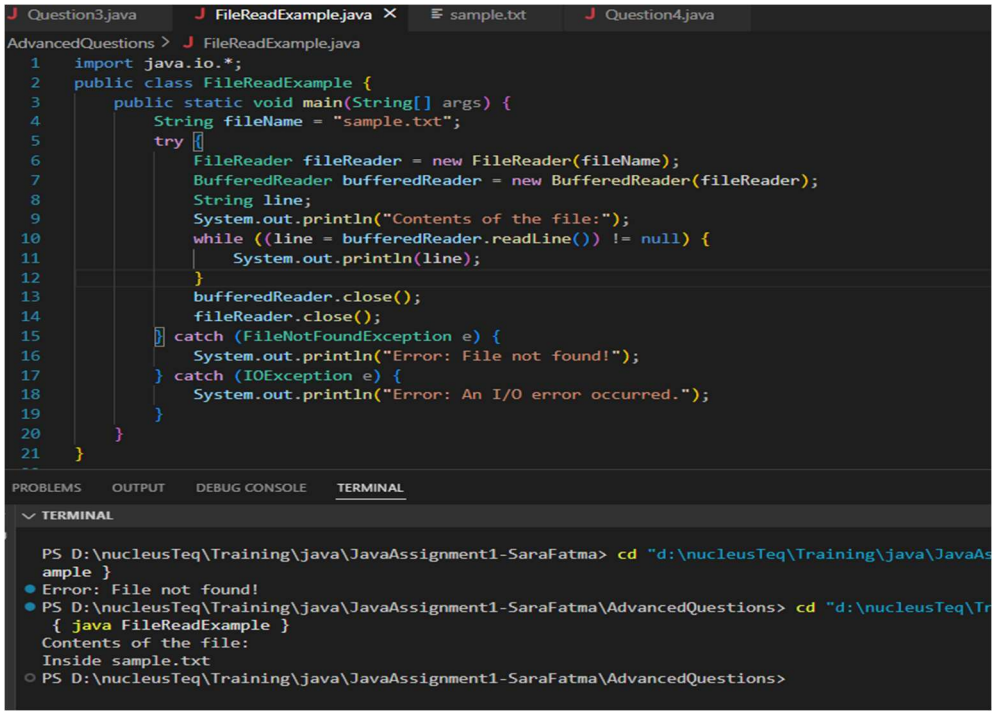
Interface Example



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



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



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