**LAB-4**

1. Method Overloading: Write a class Calculator with overloaded methods add(). Implement add() methods that take:

- Two integers

- Two double values

- Three integers

- A variable number of integers

Program-

**package** demo;

**public** **class** Calc {

//Method to add two integers

**public** **int** add(**int** a, **int** b) {

**return** a+b;

}

//Method to add two double values

**public** **double** add(**double** a, **double** b) {

**return** a+b;

}

//Method to add three integers

**public** **int** add(**int** a, **int** b, **int** c) {

**return** a+b+c;

}

//Method to add a variable number of integers

**public** **int** add(**int**... numbers) {

**int** sum=0;

**for**(**int** number:numbers) {

sum +=number;

}

**return** sum;

}

**public** **static** **void** main(String[] args) {

// **TODO** Auto-generated method stub

Calc c=**new** Calc();

//Test add methods

System.***out***.println("Addition of two integers: " + c.add(10, 2));

System.***out***.println("Addition of two double values: " + c.add(1.5, 2.5));

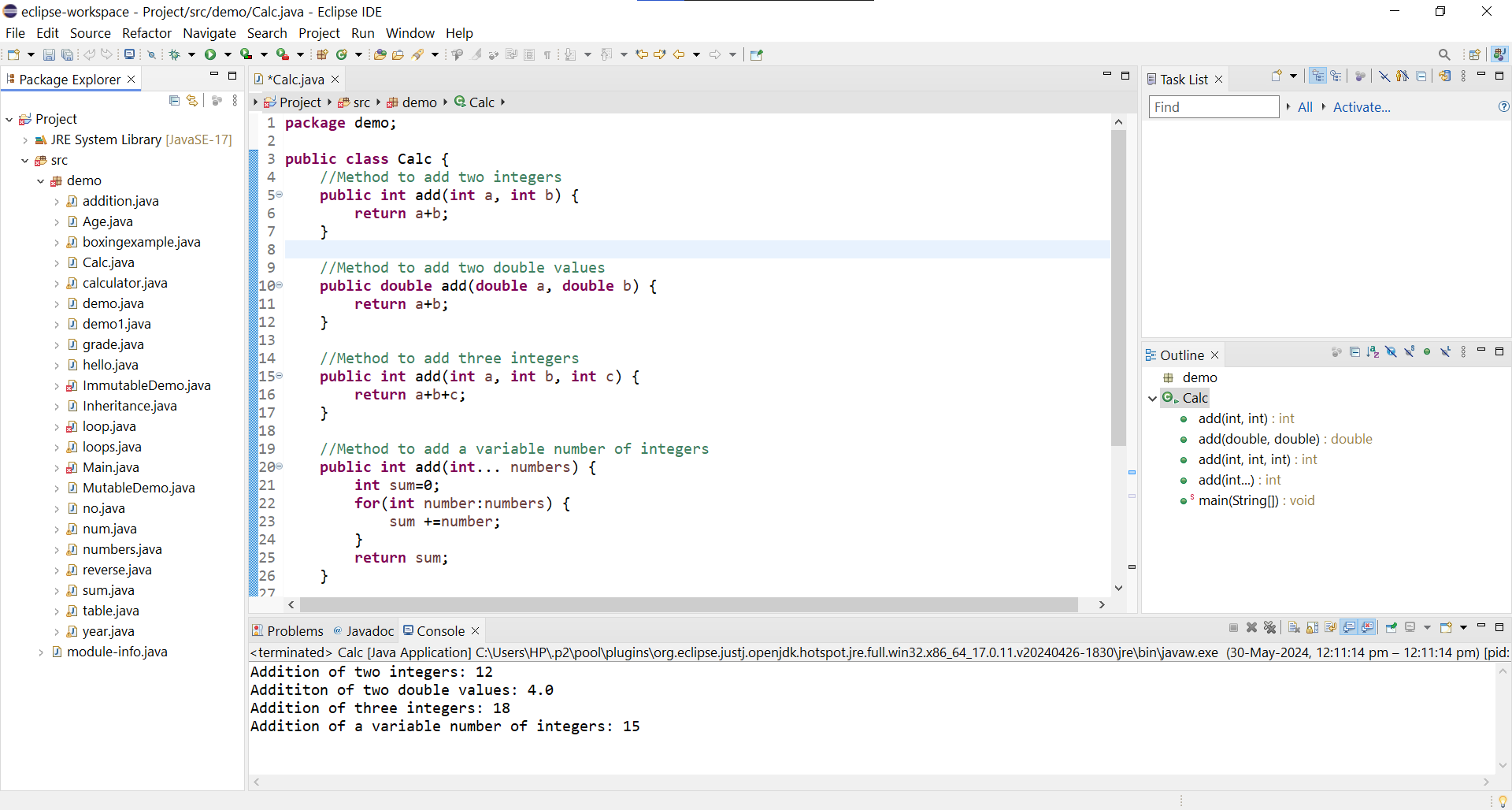
System.***out***.println("Addition of three integers: " + c.add(11, 4, 3));

System.***out***.println("Addition of a variable number of integers: " + c.add(1, 2, 3, 4, 5));

}

}

Output-



2.Super Keyword: Create a class Person with a constructor that accepts and sets name and age.

- Create a subclass Student that adds a grade property and initializes name and age using the super keyword in its constructor.

- Demonstrate the creation of Student objects and the usage of super to call the parent class constructor.

Program-

**package** demo;

**import** java.util.Scanner;

//Class Person

**class** Person {

//properties

String name;

**int** age;

//Constructor for Person

**public** Person(String name, **int** age) {

**this**.name = name;

**this**.age = age;

}

}

//Subclass Student extends Person

**class** Student **extends** Person {

String grade;

//Constructor for Student class

**public** Student(String name, **int** age, String grade) {

//Calling the constructor of the superclass

**super**(name, age);

**this**.grade = grade;

}

//Method to display information

**public** **void** display() {

System.***out***.println("Name: " + name + ", Age: " + age + ", Grade: " + grade);

}

**public** **static** **void** main(String[] args) {

Scanner scanner = **new** Scanner(System.***in***);

System.***out***.println("Enter student's name:");

String name = scanner.nextLine();

System.***out***.println("Enter student's age:");

**int** age = scanner.nextInt();

// Consume the newline character

scanner.nextLine();

System.***out***.println("Enter student's grade:");

String grade = scanner.nextLine();

Student student = **new** Student(name, age, grade);

student.display();

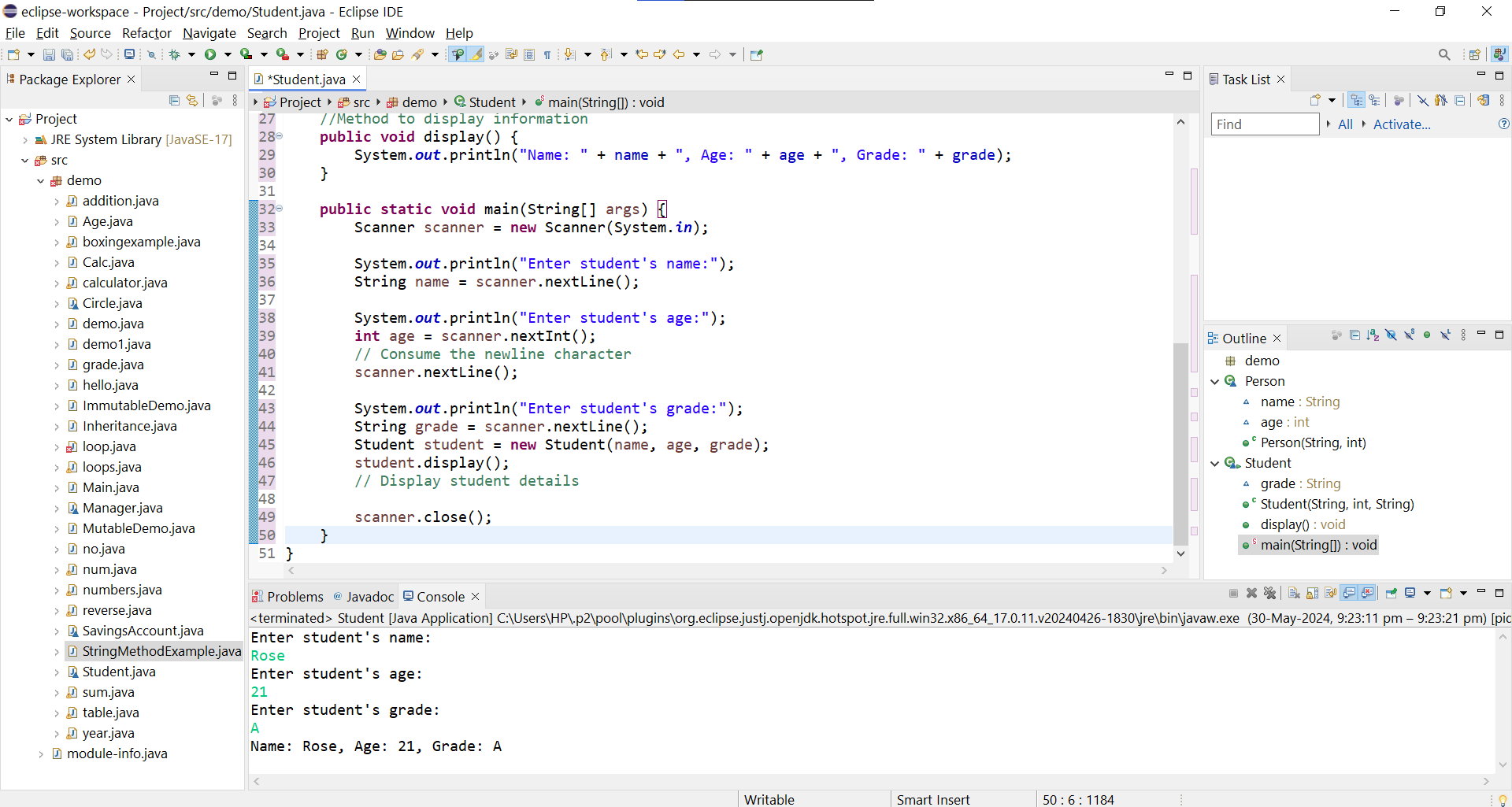
// Display student details

scanner.close();

}

}

Output-



3.Super Keyword: Create a base class Shape with a method draw() that prints "Drawing Shape".

- Create a subclass Circle that overrides draw() to print "Drawing Circle".

- Inside the draw() method of Circle, call the draw() method of the Shape class using super.draw().

- Write a main method to demonstrate calling draw() on a Circle object.

Program-

**package** demo;

//Define the base class Shape

**class** Shape {

//Method to draw the shape

**public** **void** draw() {

System.***out***.println(" Draw Shape ");

}

}

//Define the subclass Circle that extends Shape

**class** Circle **extends** Shape {

@Override

**public** **void** draw() {

**super**.draw();

// Calls the draw method in Shape class

System.***out***.println(" Draw Circle ");

}

**public** **static** **void** main(String[] args) {

Circle c = **new** Circle();

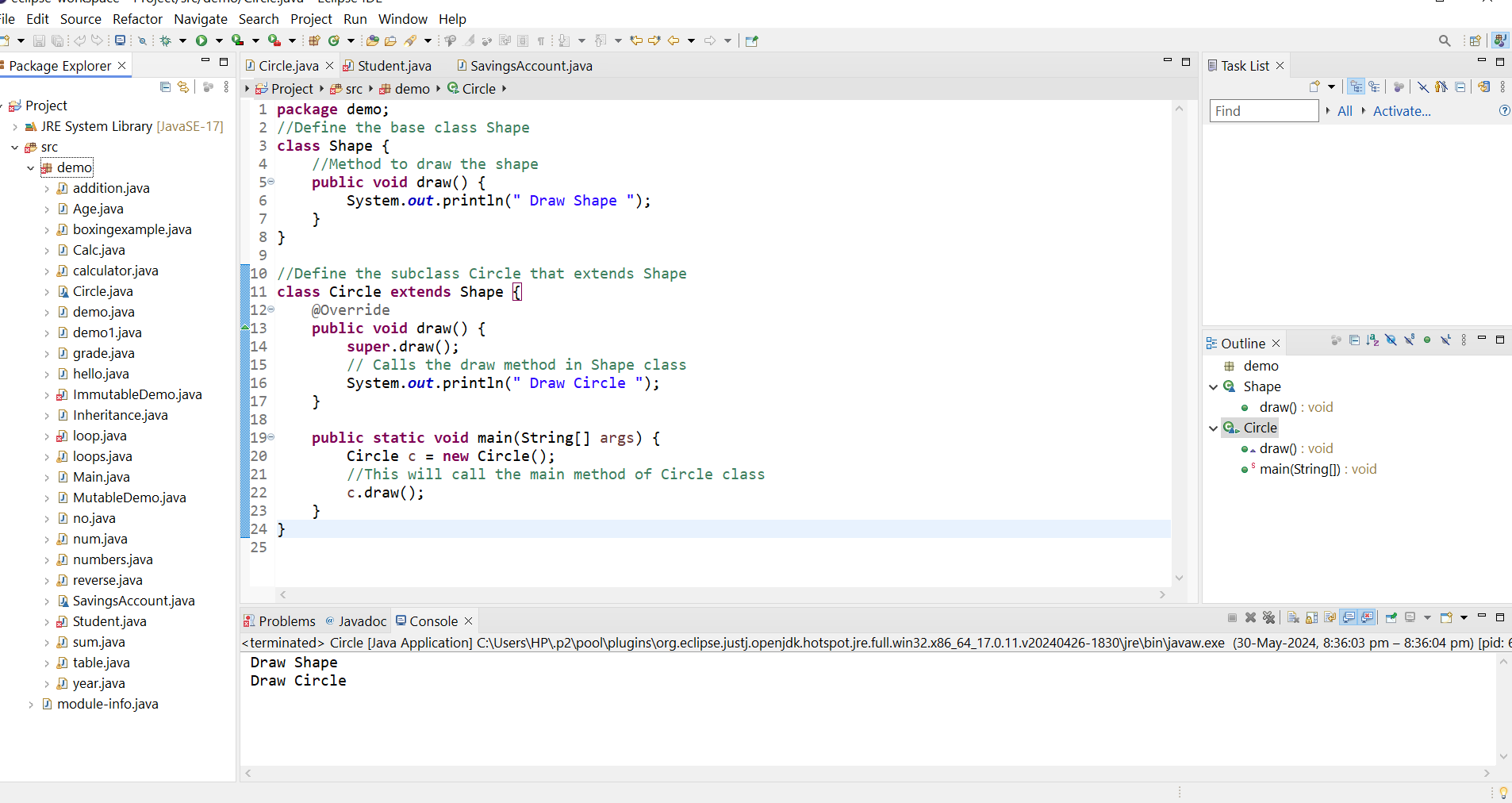
//This will call the main method of Circle class

c.draw();

}

}

Output-



4.Create a base class BankAccount with a method deposit(amount) and a constructor that sets the initial balance.

- Create a subclass SavingsAccount that overrides deposit(amount) to add interest before depositing. Use the super keyword to call the deposit method of the base class.

- Write a main method to demonstrate creating a SavingsAccount and depositing an amount to see the effect of interest.

Program-

**package** demo;

//base class BankAccount

**class** BankAccount {

**protected** **double** balance;

**public** BankAccount(**double** initialBalance) {

**this**.balance = initialBalance;

}

**public** **void** deposit(**double** amount) {

balance += amount;

}

**public** **double** getBalance() {

**return** balance;

}

}

//Subclass SavingsAccount

**class** SavingsAccount **extends** BankAccount {

**private** **double** interestRate;

**public** SavingsAccount(**double** initialBalance, **double** interestRate) {

**super**(initialBalance);

**this**.interestRate = interestRate;

}

@Override

**public** **void** deposit(**double** amount) {

**double** interest = amount \* interestRate / 100;

//Call the deposit method of BankAccount class

**super**.deposit(amount + interest);

}

**public** **static** **void** main(String[] args) {

//Create a object of SavingsAccount

SavingsAccount savings = **new** SavingsAccount(1000, 10);

//Deposit 100 to the savings account

savings.deposit(100);

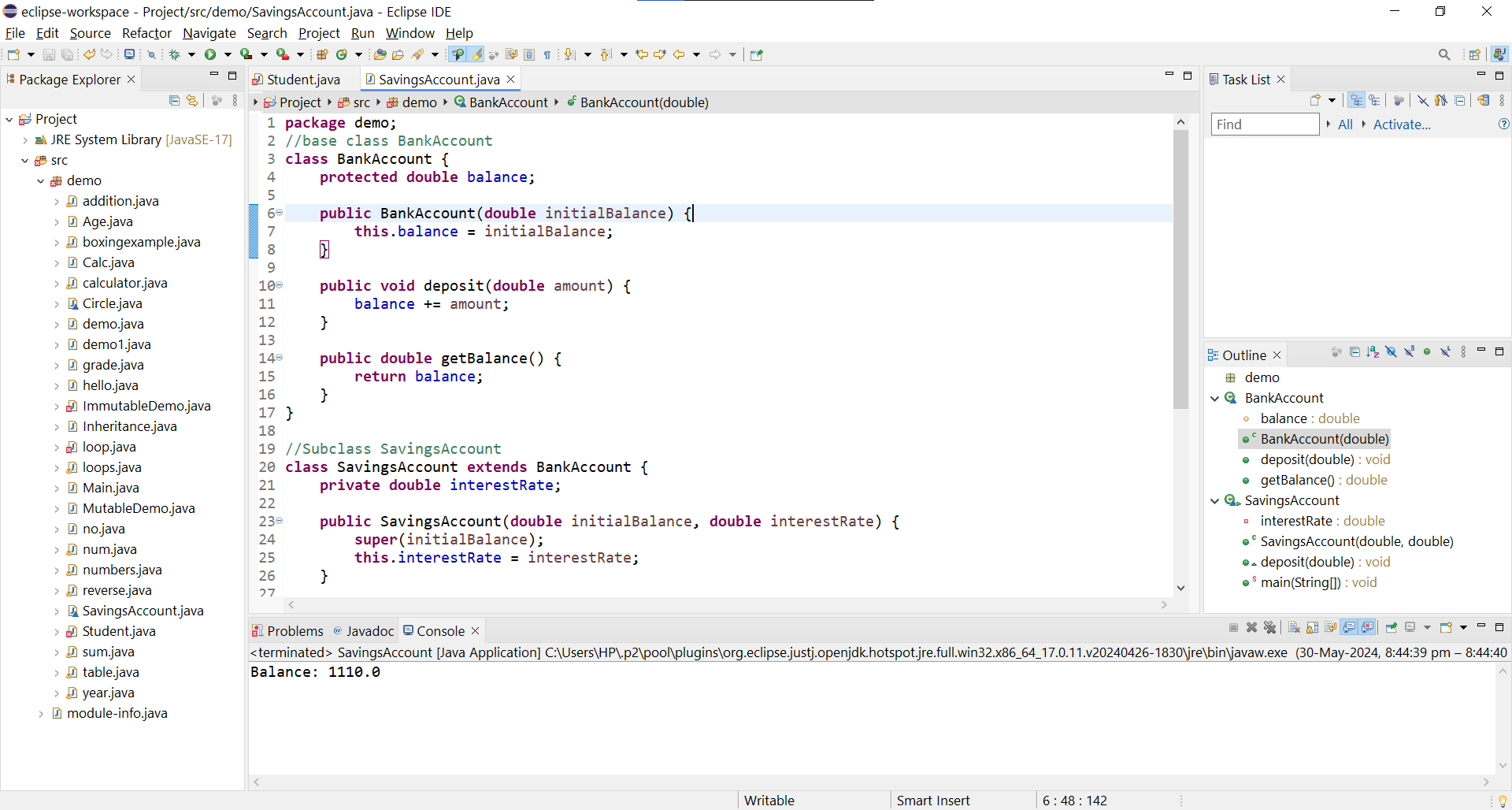
//Print the new balance

System.***out***.println("Balance: " + savings.getBalance());

}

}

Output-



5.Define a class Employee with properties name and salary and a method displayDetails().

- Create a subclass Manager that adds a property department and overrides displayDetails() to include department details. Use the super keyword to call the displayDetails() method of Employee within Manager.

- In the main method, create objects of Employee and Manager and call displayDetails() to show the details.

Program-

**import** java.util.Scanner;

**class** Employee {

String name;

**double** salary;

**public** Employee(String name, **double** salary) {

**this**.name = name;

**this**.salary = salary;

}

**public** **void** displayDetails() {

System.***out***.println("Name: " + name + ", Salary: " + salary);

}

}

**class** Manager **extends** Employee {

String department;

**public** Manager(String name, **double** salary, String department) {

**super**(name, salary);

**this**.department = department;

}

@Override

**public** **void** displayDetails() {

**super**.displayDetails();

System.***out***.println("Department: " + department);

}

**public** **static** **void** main(String[] args) {

Scanner scanner = **new** Scanner(System.***in***);

// Input for Employee

System.***out***.println("Enter Employee details:");

System.***out***.print("Name: ");

String empName = scanner.nextLine();

System.***out***.print("Salary: ");

**double** empSalary = scanner.nextDouble();

scanner.nextLine(); // Consume the newline character

Employee emp = **new** Employee(empName, empSalary);

// Input for Manager

System.***out***.println("Enter Manager details:");

System.***out***.print("Name: ");

String mgrName = scanner.nextLine();

System.***out***.print("Salary: ");

**double** mgrSalary = scanner.nextDouble();

scanner.nextLine(); // Consume the newline character

System.***out***.print("Department: ");

String mgrDepartment = scanner.nextLine();

Manager mgr = **new** Manager(mgrName, mgrSalary, mgrDepartment);

// Display details

System.***out***.println("\nEmployee Details:");

emp.displayDetails();

System.***out***.println("\nManager Details:");

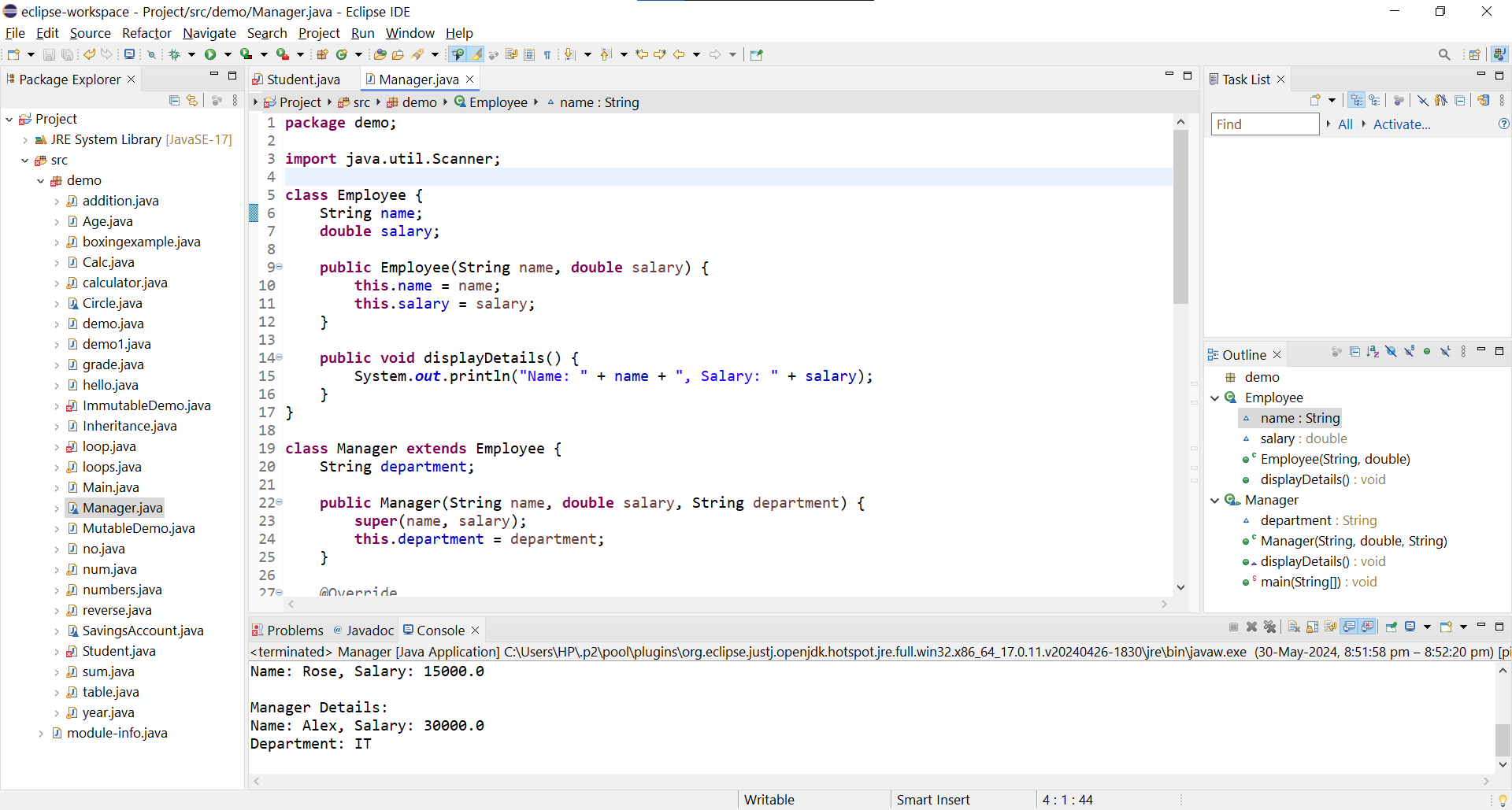
mgr.displayDetails();

scanner.close();

}

}

Output-



6.Write the same programme for the class ImmutableExample, to achieve object value ‘Hi’.

Program-

**package** demo;

**public** **final** **class** ImmutableDemo {

**private** String name;

ImmutableDemo (String name)

{

**this**.name = name;

}

**public** String getName()

{

**return** name;

}

**public** **static** **void** main(String[] args) {

// **TODO** Auto-generated method stub

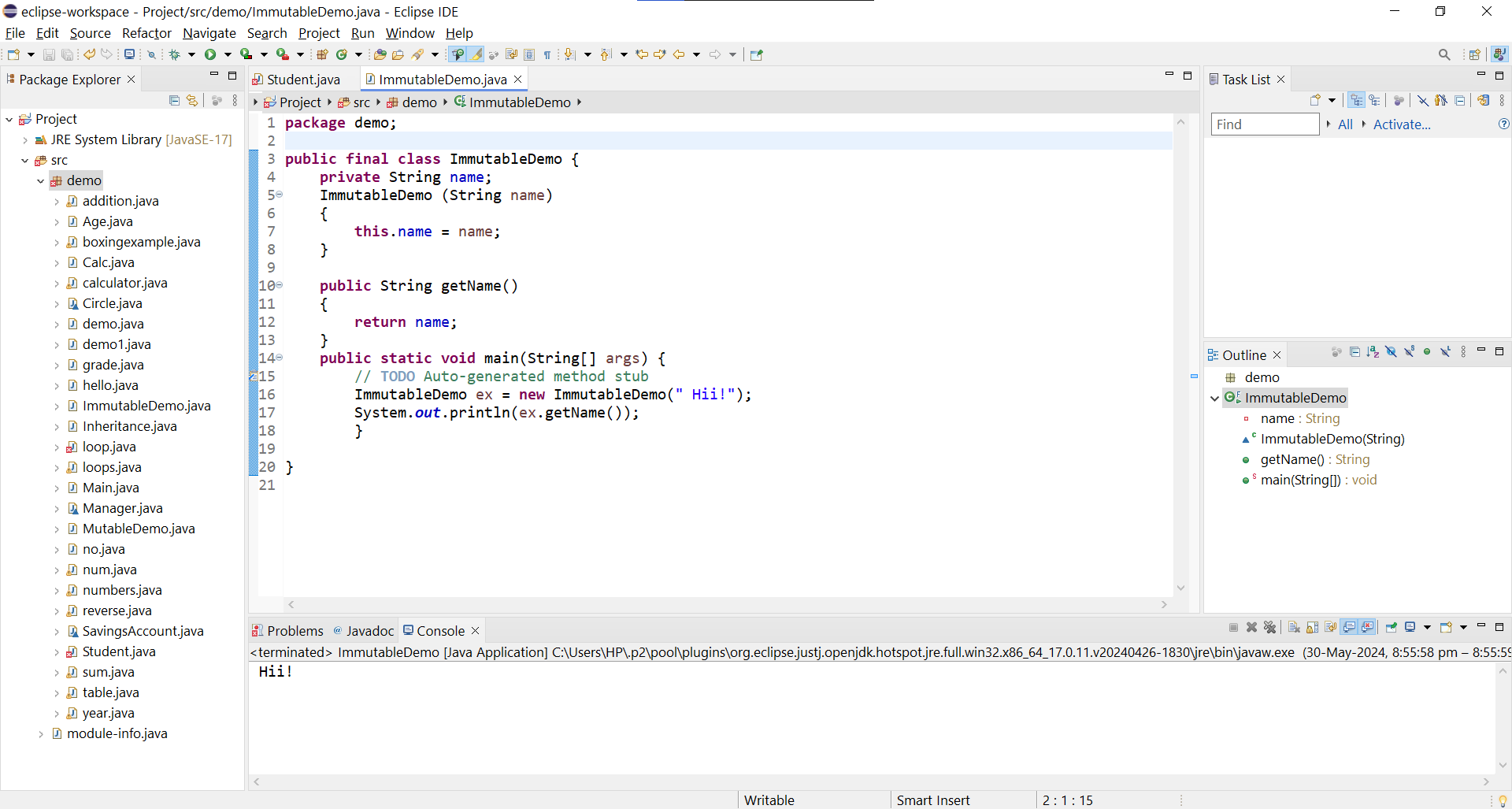
ImmutableDemo ex = **new** ImmutableDemo(" Hii!");

System.***out***.println(ex.getName());

}

}

Output-



1. Write the same programme for the class MutableExample, to output the object values ‘hello 2’ and ‘hello3’

Program-

**package** demo;

**public** **class** MutableDemo {

**private** String name;

MutableDemo(String name) {

**this**.name = name;

}

**public** String getName()

{

**return** name;

}

**public** **void** setName(String name)

{

**this**.name = name;

}

**public** **static** **void** main(String[] args) {

// **TODO** Auto-generated method stub

//Create a MutableDemo object

MutableDemo obj = **new** MutableDemo(" hello2 ");

System.***out***.println(obj.getName());

//Modify the values

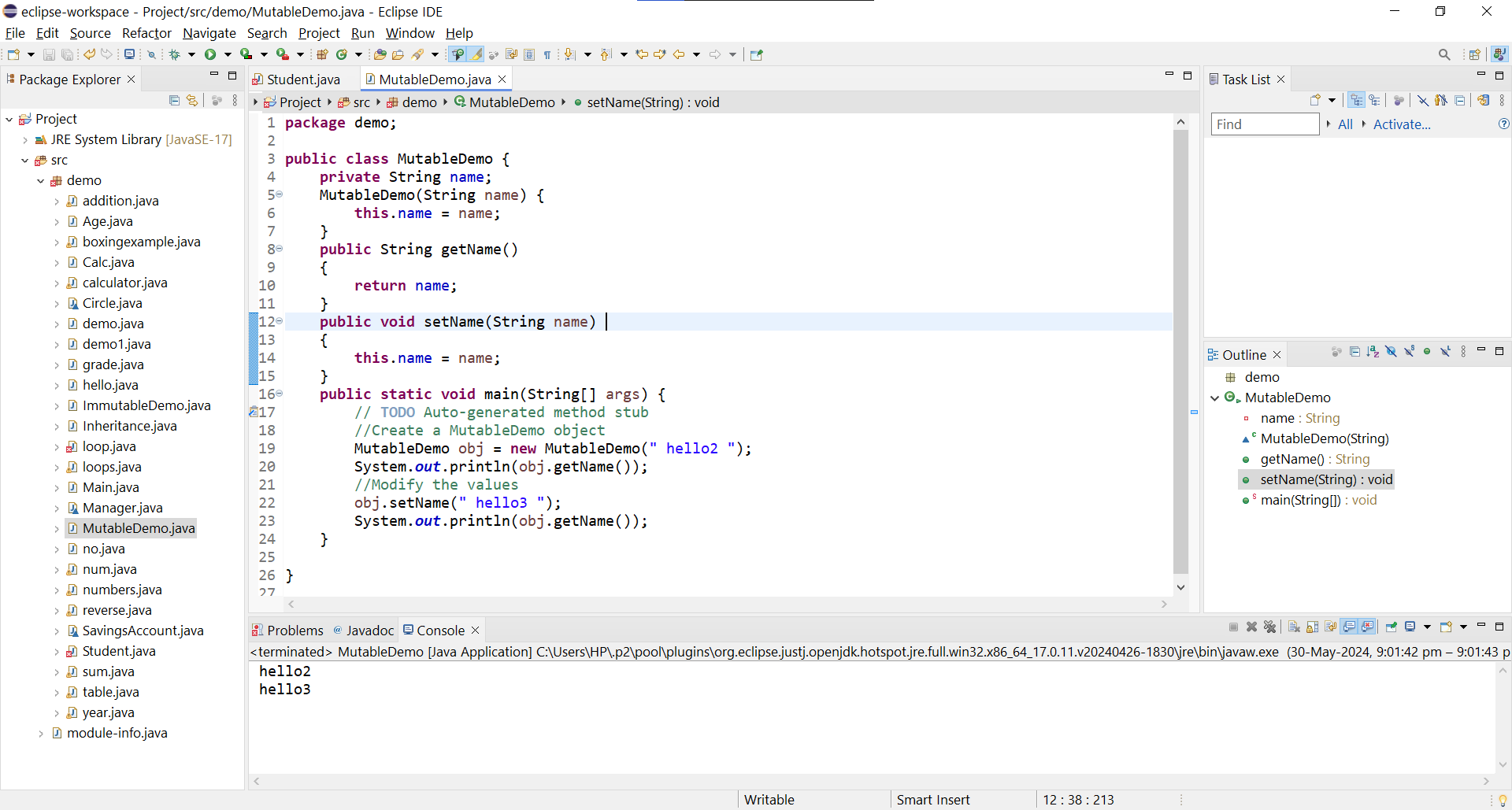
obj.setName(" hello3 ");

System.***out***.println(obj.getName());

}

}

Output-



8.Write a java class to implement any 10 string methods:

● replace ● contains ● replaceAll ● indexOf ● substring ● Equals ● lastIndexOf ● startsWith

● endsWith ● EqualsIgnoreCase ● toLowerCase ● toUpperCase ● isEmpty ● Length ● split

Program-

**package** demo;

**public** **class** StringMethodExample {

**public** **static** **void** main(String[] args) {

String str = "Hello, World!!";

// replace method

String replaced = str.replace('o', '0');

System.***out***.println("Replace 'o' with '0': " + replaced);

// contains method

**boolean** contains = str.contains("World");

System.***out***.println("Contains 'World': " + contains);

// replaceAll method

String replacedAll = str.replaceAll("[aeiou]", "\*");

System.***out***.println("Replace all vowels with '\*': " + replacedAll);

// indexOf method

**int** index = str.indexOf('o');

System.***out***.println("Index of 'o': " + index);

// substring method

String substring = str.substring(7);

System.***out***.println("Substring from index 7: " + substring);

// equals method

String str2 = "Hello, World!";

**boolean** equals = str.equals(str2);

System.***out***.println("Equals to 'Hello, World!': " + equals);

// lastIndexOf method

**int** lastIndex = str.lastIndexOf('o');

System.***out***.println("Last Index of 'o': " + lastIndex);

// startsWith method

**boolean** startsWith = str.startsWith("Hello");

System.***out***.println("Starts with 'Hello': " + startsWith);

// endsWith method

**boolean** endsWith = str.endsWith("World!");

System.***out***.println("Ends with 'World!': " + endsWith);

// equalsIgnoreCase method

**boolean** equalsIgnoreCase = str.equalsIgnoreCase("HELLO, WORLD!");

System.***out***.println("Equals ignore case 'HELLO, WORLD!': " + equalsIgnoreCase);

// toLowerCase method

String lowerCase = str.toLowerCase();

System.***out***.println("To lower case: " + lowerCase);

// toUpperCase method

String upperCase = str.toUpperCase();

System.***out***.println("To upper case: " + upperCase);

// isEmpty method

**boolean** isEmpty = str.isEmpty();

System.***out***.println("Is empty: " + isEmpty);

// length method

**int** length = str.length();

System.***out***.println("Length: " + length);

// split method

String[] parts = str.split(",");

System.***out***.println("Split by ',': ");

**for** (String part : parts) {

System.***out***.println(part.trim());

}

}

}

Output-

