VISVESVARAYA TECHNOLOGICAL UNIVERSITY

"JnanaSangama", Belgaum -590014, Karnataka.

LAB REPORT
On

OBJECT ORIENTED JAVA

Submitted by

TARANG RAJVANSHI

in partial fulfillment for the award of the degree of
BACHELOR OF ENGINEERING
in
COMPUTER SCIENCE AND ENGINEERING

B.M.S. COLLEGE OF ENGINEERING
(Autonomous Institution under VTU)
BENGALURU-560019
Dec 2023- March 2024

B. M. S. College of Engineering,
Bull Temple Road, Bangalore 560019
(Affiliated To Visvesvaraya Technological University, Belgaum)
Department of Computer Science and Engineering

This is to certify that the Lab work entitled **"OBJECT ORIENTED PROGRAMMING"** carried out by TARANG**(1BM22CS305)**, who is bonafide student of **B. M. S. College of Engineering**. It is in partial fulfillment for the award of **Bachelor of Engineering in Computer Science and Engineering** of the Visvesvaraya Technological University, Belgaum during the year 2023-24. The Lab report has been approved as it satisfies the academic requirements in respect of OBJECT ORIENTED JAVA Lab - **(23CS3PCOOJ)**work prescribed for the said degree.

Prof. Shravaya
Assistant Professor
Department of CSE
BMSCE, Bengaluru

Dr. Jyothi S Nayak
Professor and Head
Department of CSE
BMSCE, Bengaluru

Index Sheet

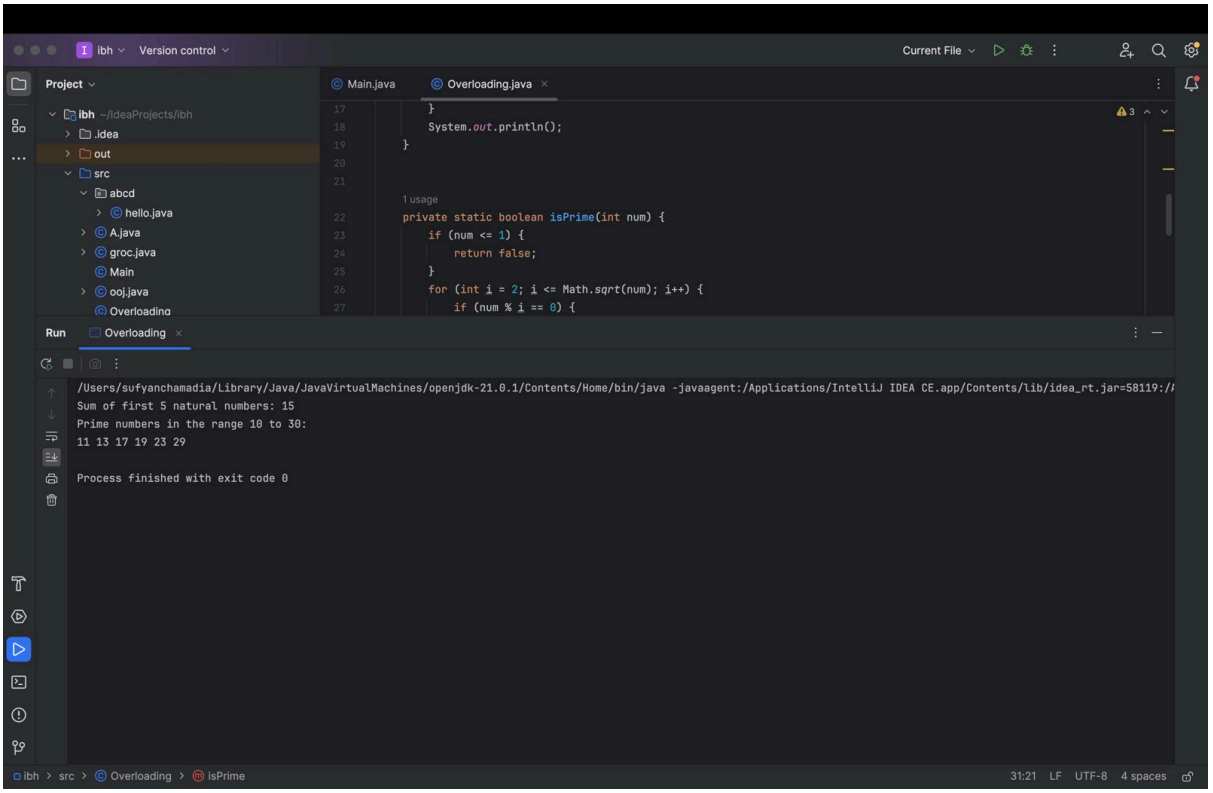
Course outcomes:

CO1	Apply the concept of linear and nonlinear data structures.
CO2	Analyze data structure operations for a given problem
CO3	Design and develop solutions using the operations of linear and nonlinear data structure for a given specification.
CO4	Conduct practical experiments for demonstrating the operations of different data structures.

v

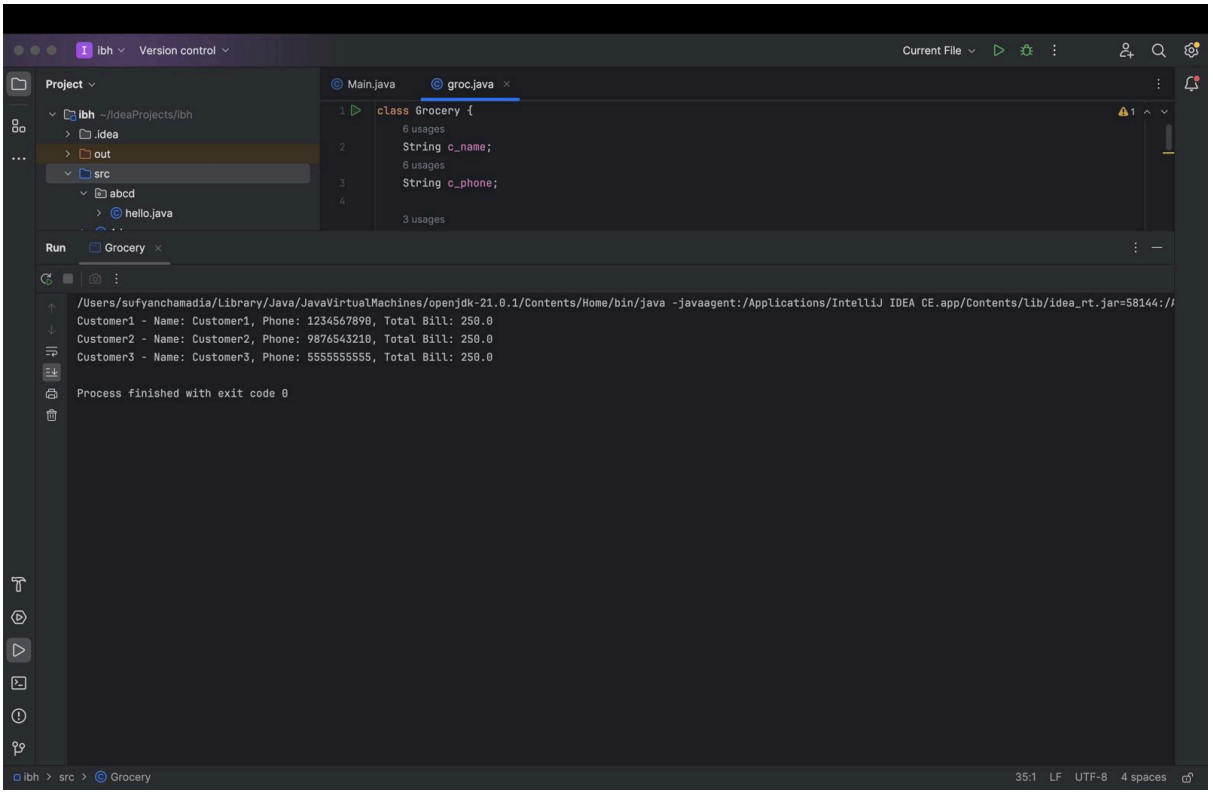
```
public class Overloading {
    static void print(int n) {
        int sum = 0;
        for (int i = 1; i <= n; i++) {
            sum += i;
        }
        System.out.println("Sum of first " + n + " natural numbers: " + sum);
    }



    static void print(int start, int end) {
        System.out.println("Prime numbers in the range " + start + " to " + end + ":");
        for (int num = start; num <= end; num++) {
            if (isPrime(num)) {
                System.out.print(num + " ");
            }
        }
        System.out.println();
    }



    private static boolean isPrime(int num) {
        if (num <= 1) {
            return false;
        }
        for (int i = 2; i <= Math.sqrt(num); i++) {
            if (num % i == 0) {
                return false;
            }
```

```
        }
        return true;
    }

    public static void main(String[] args) {

        print(5);


        print(10, 30);
    }
}
```



```
class Grocery {
    String c_name;
    String c_phone;

    double calculateTotalBill(int dalQty, int pulsesQty, int sugarQty) {
        double dalPrice = 50.0;   // Price per kg for dal
        double pulsesPrice = 40.0;   // Price per kg for pulses
        double sugarPrice = 30.0;   // Price per kg for sugar

        double totalBill = (dalQty * dalPrice) + (pulsesQty * pulsesPrice) + (sugarQty *
sugarPrice);
        return totalBill;
    }

    public static void main(String[] args) {
        Grocery customer1 = new Grocery();
        customer1.c_name = "Customer1";
        customer1.c_phone = "1234567890";
        double bill1 = customer1.calculateTotalBill(2, 3, 1);

        Grocery customer2 = new Grocery();
        customer2.c_name = "Customer2";
        customer2.c_phone = "9876543210";
        double bill2 = customer2.calculateTotalBill(1, 2, 4);

        Grocery customer3 = new Grocery();
        customer3.c_name = "Customer3";
        customer3.c_phone = "5555555555";
        double bill3 = customer3.calculateTotalBill(3, 1, 2);
```

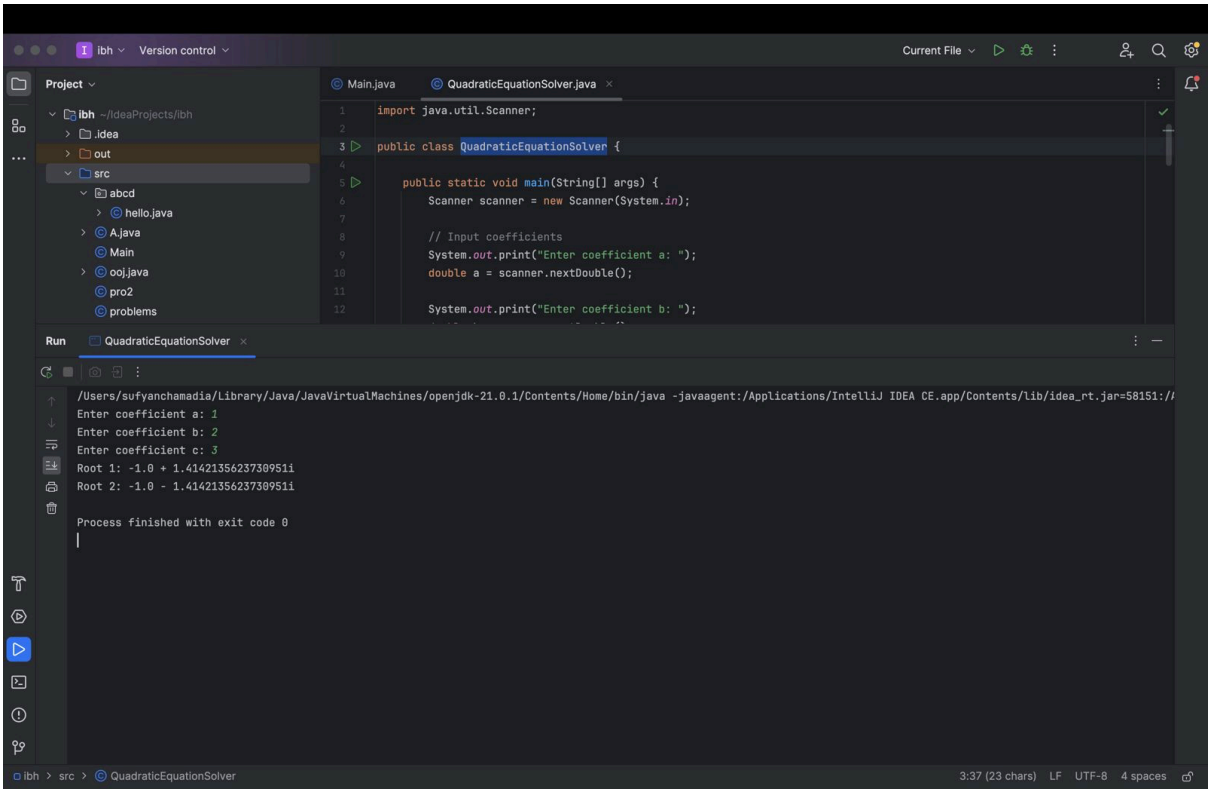
```
        System.out.println("Customer1 - Name: " + customer1.c_name + ", Phone: " +
customer1.c_phone + ", Total Bill: " + bill1);

        System.out.println("Customer2 - Name: " + customer2.c_name + ", Phone: " +
customer2.c_phone + ", Total Bill: " + bill2);

        System.out.println("Customer3 - Name: " + customer3.c_name + ", Phone: " +
customer3.c_phone + ", Total Bill: " + bill3);

    }

}
```




```
import java.util.Scanner;


public class QuadraticEquationSolver {


    public static void main(String[] args) {

        Scanner scanner = new Scanner(System.in);


        // Input coefficients

        System.out.print("Enter coefficient a: ");
```

```
        double a = scanner.nextDouble();

        System.out.print("Enter coefficient b: ");
        double b = scanner.nextDouble();

        System.out.print("Enter coefficient c: ");
        double c = scanner.nextDouble();

        // Calculate and display roots
        calculateAndDisplayRoots(a, b, c);

        scanner.close();
    }

    static void calculateAndDisplayRoots(double a, double b, double c) {
        // Calculate the discriminant
        double discriminant = b * b - 4 * a * c;

        // Check if the discriminant is non-negative
        if (discriminant >= 0) {
            // Calculate the roots
            double root1 = (-b + Math.sqrt(discriminant)) / (2 * a);
            double root2 = (-b - Math.sqrt(discriminant)) / (2 * a);

            // Display the roots
            System.out.println("Root 1: " + root1);
            System.out.println("Root 2: " + root2);
        } else {
            // If discriminant is negative, roots are imaginary
```

```
        double realPart = -b / (2 * a);
        double imaginaryPart = Math.sqrt(-discriminant) / (2 * a);

        // Display the roots as complex numbers
        System.out.println("Root 1: " + realPart + " + " + imaginaryPart + "i");
        System.out.println("Root 2: " + realPart + " - " + imaginaryPart + "i");
    }
  }
}
```

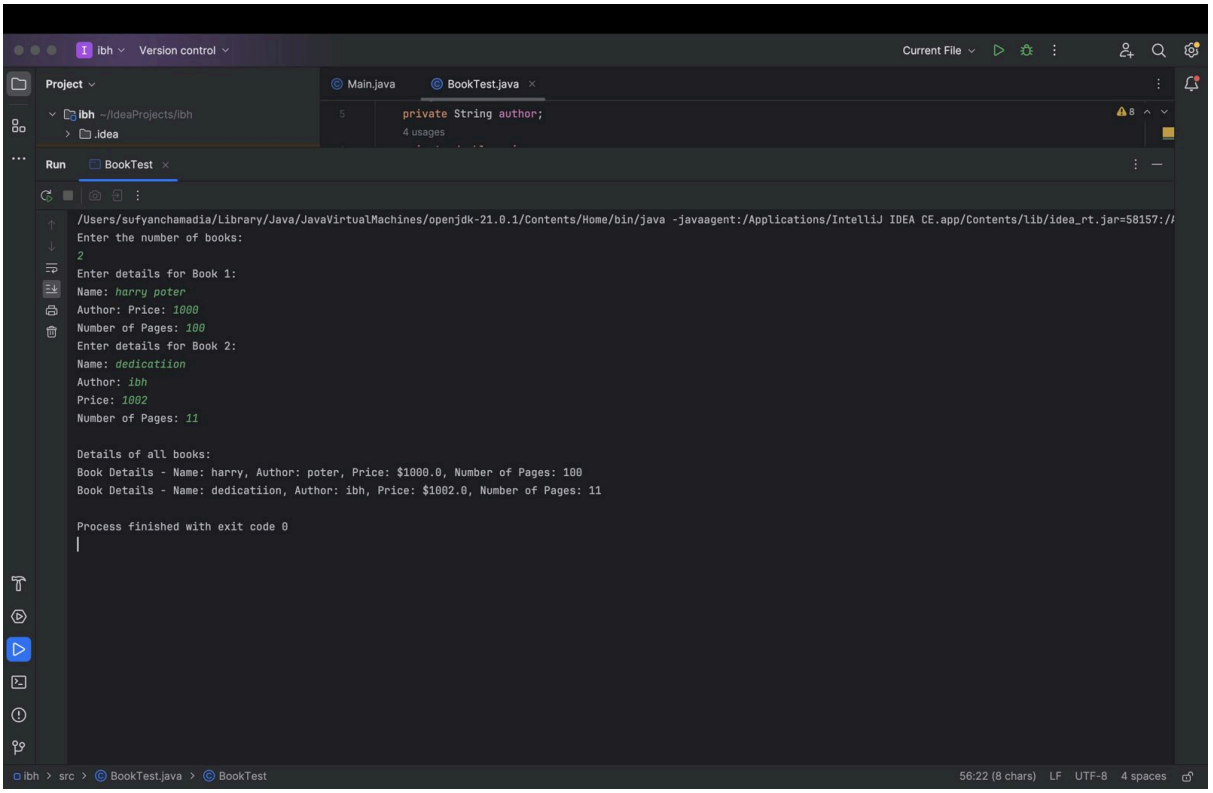



```
import java.util.Scanner;

class Book {
    private String name;
    private String author;
    private double price;
```

```
    private int num_pages;

    // Constructor to set the values for the members
    public Book(String name, String author, double price, int num_pages) {
        this.name = name;
        this.author = author;
        this.price = price;
        this.num_pages = num_pages;
    }

    // Methods to set and get the details of the objects
    public void setName(String name) {
        this.name = name;
    }

    public void setAuthor(String author) {
        this.author = author;
    }

    public void setPrice(double price) {
        this.price = price;
    }

    public void setNumPages(int num_pages) {
        this.num_pages = num_pages;
    }

    public String getName() {
        return name;
```

```
    }

    public String getAuthor() {
        return author;
    }

    public double getPrice() {
        return price;
    }

    public int getNumPages() {
        return num_pages;
    }

    // toString method to display the complete details of the book
    public String toString() {
        return "Book Details - Name: " + name + ", Author: " + author + ", Price: $" +
price + ", Number of Pages: " + num_pages;
    }
}

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

        System.out.println("Enter the number of books: ");
        int n = scanner.nextInt();

        Book[] books = new Book[n];
```

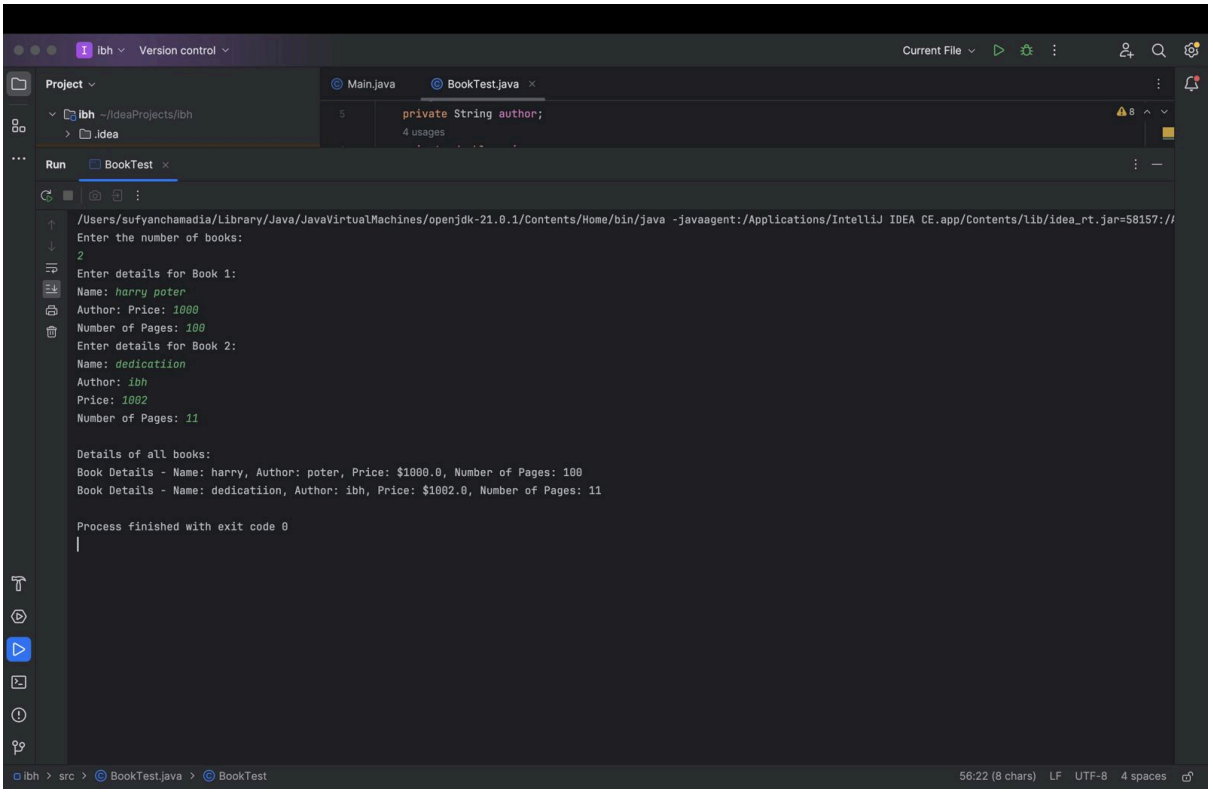
```
        for (int i = 0; i < n; i++) {
            System.out.println("Enter details for Book " + (i + 1) + ":");
            System.out.print("Name: ");
            String name = scanner.next();

            System.out.print("Author: ");
            String author = scanner.next();

            System.out.print("Price: ");
            double price = scanner.nextDouble();

            System.out.print("Number of Pages: ");
            int numPages = scanner.nextInt();

            books[i] = new Book(name, author, price, numPages);
        }

        // Display details of all books
        System.out.println("\nDetails of all books:");
        for (int i = 0; i < n; i++) {
            System.out.println(books[i].toString());
        }
    }
}
```




```
import java.util.Scanner;

abstract class Shape {
    protected int dimension1;
    protected int dimension2;

    public Shape(int dimension1, int dimension2) {
        this.dimension1 = dimension1;
        this.dimension2 = dimension2;
    }

    // Abstract method to be implemented by subclasses
    abstract void printArea();
}
```

```
class Rectangle extends Shape {
    public Rectangle(int length, int width) {
        super(length, width);
    }

    @Override
    void printArea() {
        int area = dimension1 * dimension2;
        System.out.println("Area of Rectangle: " + area);
    }
}

class Triangle extends Shape {
    public Triangle(int base, int height) {
        super(base, height);
    }

    @Override
    void printArea() {
        double area = 0.5 * dimension1 * dimension2;
        System.out.println("Area of Triangle: " + area);
    }
}

class Circle extends Shape {
    public Circle(int radius) {
        super(radius, 0);
    }
```

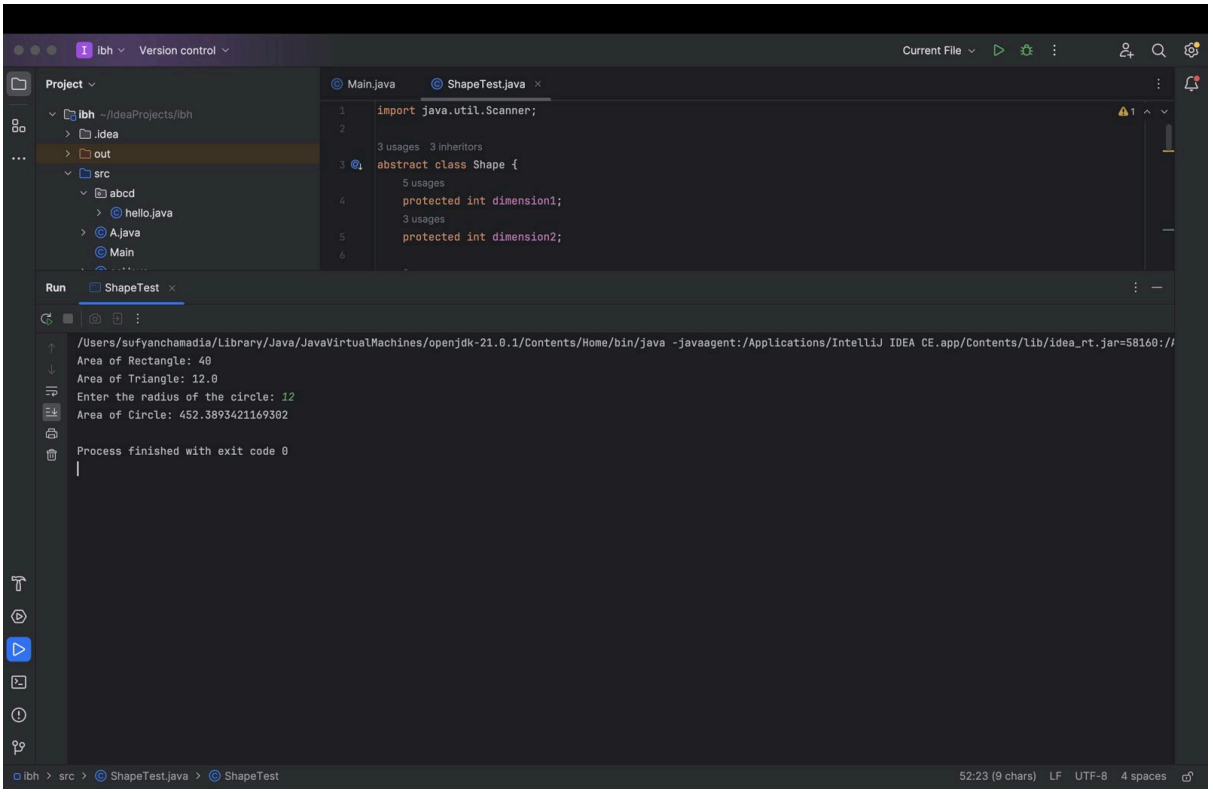
```
    @Override
    void printArea() {
        double area = Math.PI * dimension1 * dimension1;
        System.out.println("Area of Circle: " + area);
    }
}

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

        // Example for Rectangle
        Rectangle rectangle = new Rectangle(5, 8);
        rectangle.printArea();

        // Example for Triangle
        Triangle triangle = new Triangle(4, 6);
        triangle.printArea();

        // Example for Circle
        System.out.print("Enter the radius of the circle: ");
        int radius = scanner.nextInt();
        Circle circle = new Circle(radius);
        circle.printArea();
    }
}
```




```
import java.util.Scanner;

abstract class Shape {
    protected int dimension1;
    protected int dimension2;

    public Shape(int dimension1, int dimension2) {
        this.dimension1 = dimension1;
        this.dimension2 = dimension2;
    }

    // Abstract method to be implemented by subclasses
    abstract void printArea();
}

class Rectangle extends Shape {
```

```
    public Rectangle(int length, int width) {
        super(length, width);
    }

    @Override
    void printArea() {
        int area = dimension1 * dimension2;
        System.out.println("Area of Rectangle: " + area);
    }
}

class Triangle extends Shape {
    public Triangle(int base, int height) {
        super(base, height);
    }

    @Override
    void printArea() {
        double area = 0.5 * dimension1 * dimension2;
        System.out.println("Area of Triangle: " + area);
    }
}

class Circle extends Shape {
    public Circle(int radius) {
        super(radius, 0);
    }

    @Override
```

```
    void printArea() {
        double area = Math.PI * dimension1 * dimension1;
        System.out.println("Area of Circle: " + area);
    }
}

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

        // Example for Rectangle
        Rectangle rectangle = new Rectangle(5, 8);
        rectangle.printArea();

        // Example for Triangle
        Triangle triangle = new Triangle(4, 6);
        triangle.printArea();

        // Example for Circle
        System.out.print("Enter the radius of the circle: ");
        int radius = scanner.nextInt();
        Circle circle = new Circle(radius);
        circle.printArea();
    }
}
```




```
import java.util.Scanner;


class Account {
    String customerName;
    long accountNumber;
    String accountType;
    double balance;

    public Account(String customerName, long accountNumber, String accountType,
double balance) {
        this.customerName = customerName;
        this.accountNumber = accountNumber;
        this.accountType = accountType;
        this.balance = balance;
    }

    void deposit(double amount) {
```

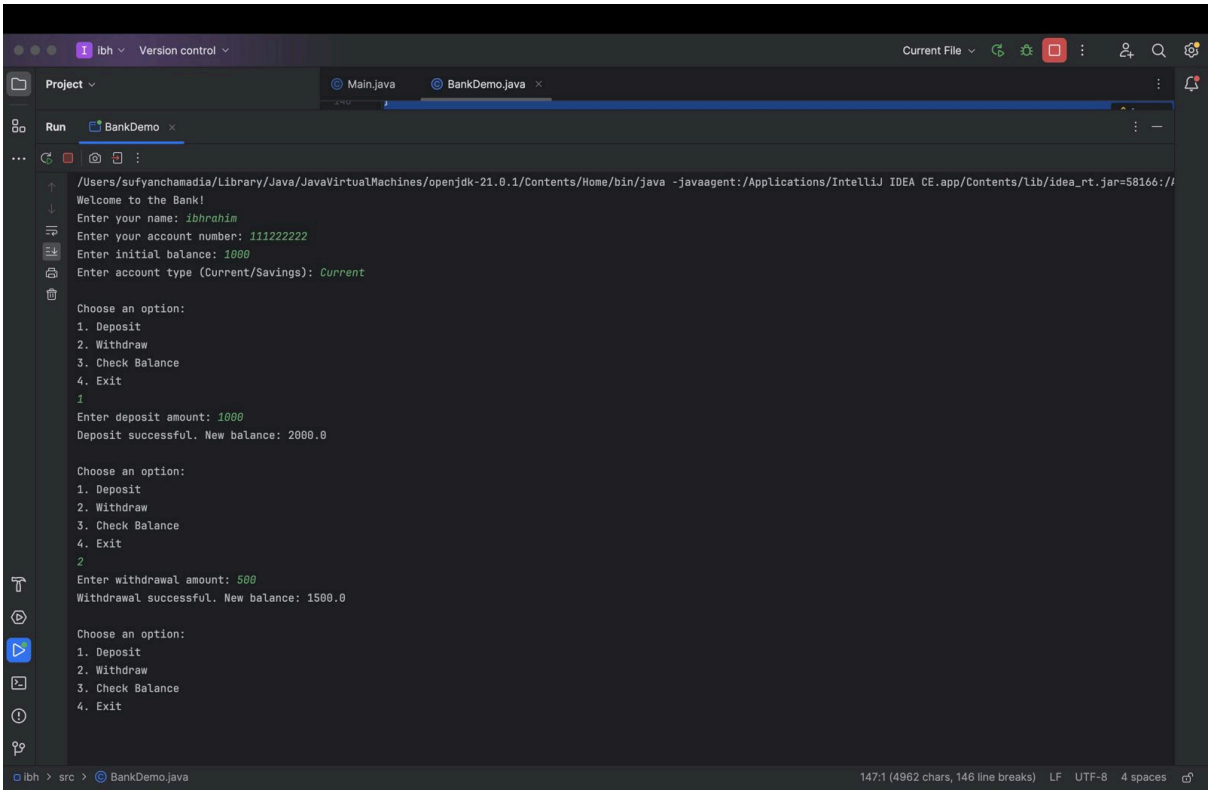
```
        balance += amount;
        System.out.println("Deposit successful. New balance: " + balance);
    }

    void displayBalance() {
        System.out.println("Account Number: " + accountNumber);
        System.out.println("Balance: " + balance);
    }

    void computeInterest() {
        // Default implementation for accounts with no interest
        System.out.println("This account type does not earn interest.");
    }

    void withdraw(double amount) {
        if (balance >= amount) {
            balance -= amount;
            System.out.println("Withdrawal successful. New balance: " + balance);
        } else {
            System.out.println("Insufficient funds. Withdrawal failed.");
        }
    }
}

class CurrentAccount extends Account {
    double minimumBalance;
    double serviceCharge;

    public CurrentAccount(String customerName, long accountNumber, double balance)
{
```

```
        super(customerName, accountNumber, "Current", balance);
        this.minimumBalance = 500; // Example minimum balance
        this.serviceCharge = 20; // Example service charge
    }

    @Override
    void computeInterest() {
        // Current account does not earn interest
        System.out.println("Current account does not earn interest.");
    }

    @Override
    void withdraw(double amount) {
        if (balance - amount >= minimumBalance) {
            balance -= amount;
            System.out.println("Withdrawal successful. New balance: " + balance);
        } else {
            System.out.println("Insufficient funds. Service charge of $" + serviceCharge + "
applied.");
            balance -= serviceCharge;
            System.out.println("New balance after service charge: " + balance);
        }
    }
}

class SavingsAccount extends Account {
    double interestRate;

    public SavingsAccount(String customerName, long accountNumber, double balance)
{
```

```
        super(customerName, accountNumber, "Savings", balance);
        this.interestRate = 0.05; // Example interest rate (5%)
    }

    @Override
    void computeInterest() {
        double interest = balance * interestRate;
        balance += interest;
        System.out.println("Interest computed and added. New balance: " + balance);
    }
}

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

        System.out.println("Welcome to the Bank!");

        System.out.print("Enter your name: ");
        String customerName = scanner.nextLine();

        System.out.print("Enter your account number: ");
        long accountNumber = scanner.nextLong();

        System.out.print("Enter initial balance: ");
        double initialBalance = scanner.nextDouble();

        System.out.print("Enter account type (Current/Savings): ");
        String accountType = scanner.next();
```

```
        // Choose the account type based on user input
        Account userAccount;
        if (accountType.equalsIgnoreCase("Current")) {
            userAccount = new CurrentAccount(customerName, accountNumber,
initialBalance);
        } else if (accountType.equalsIgnoreCase("Savings")) {
            userAccount = new SavingsAccount(customerName, accountNumber,
initialBalance);
        } else {
            System.out.println("Invalid account type. Exiting.");
            return;
        }

        while (true) {
            System.out.println("\nChoose an option:");
            System.out.println("1. Deposit");
            System.out.println("2. Withdraw");
            System.out.println("3. Check Balance");
            System.out.println("4. Exit");

            int choice = scanner.nextInt();

            switch (choice) {
                case 1:
                    System.out.print("Enter deposit amount: ");
                    double depositAmount = scanner.nextDouble();
                    userAccount.deposit(depositAmount);
                    break;
                case 2:
```

```
                System.out.print("Enter withdrawal amount: ");
                double withdrawalAmount = scanner.nextDouble();
                userAccount.withdraw(withdrawalAmount);
                break;
            case 3:
                userAccount.displayBalance();
                break;
            case 4:
                System.out.println("Thank you for using the Bank. Exiting.");
                return;
            default:
                System.out.println("Invalid option. Please choose again.");
        }
    }
}
```



```
package CIE;

public class Student {
    String usn;
    String name;
    int sem;

    public Student(String usn, String name, int sem) {
        this.usn = usn;
        this.name = name;
        this.sem = sem;
    }
}
package CIE;

public class Internals extends Student {
    int[] internalMarks;

    public Internals(String usn, String name, int sem, int[] internalMarks) {
        super(usn, name, sem);
        this.internalMarks = internalMarks;
    }
}
package SEE;

import CIE.Student;

public class External extends Student {
    int[] externalMarks;
```

```
    public External(String usn, String name, int sem, int[] externalMarks) {
        super(usn, name, sem);
        this.externalMarks = externalMarks;
    }
}
import CIE.Internals;
import SEE.External;

public class Main {
    public static void main(String[] args) {
        // Example usage
        int[] internalMarks = {75, 80, 85, 90, 95};
        Internals studentCIE = new Internals("123", "John Doe", 3, internalMarks);

        int[] externalMarks = {85, 90, 75, 88, 92};
        External studentSEE = new External("123", "John Doe", 3, externalMarks);

        // Calculate final marks and display
        displayFinalMarks(studentCIE, studentSEE);
    }

    static void displayFinalMarks(Internals cie, External see) {
        System.out.println("Student Details:");
        System.out.println("USN: " + cie.usn);
        System.out.println("Name: " + cie.name);
        System.out.println("Semester: " + cie.sem);

        System.out.println("\nInternal Marks:");
```

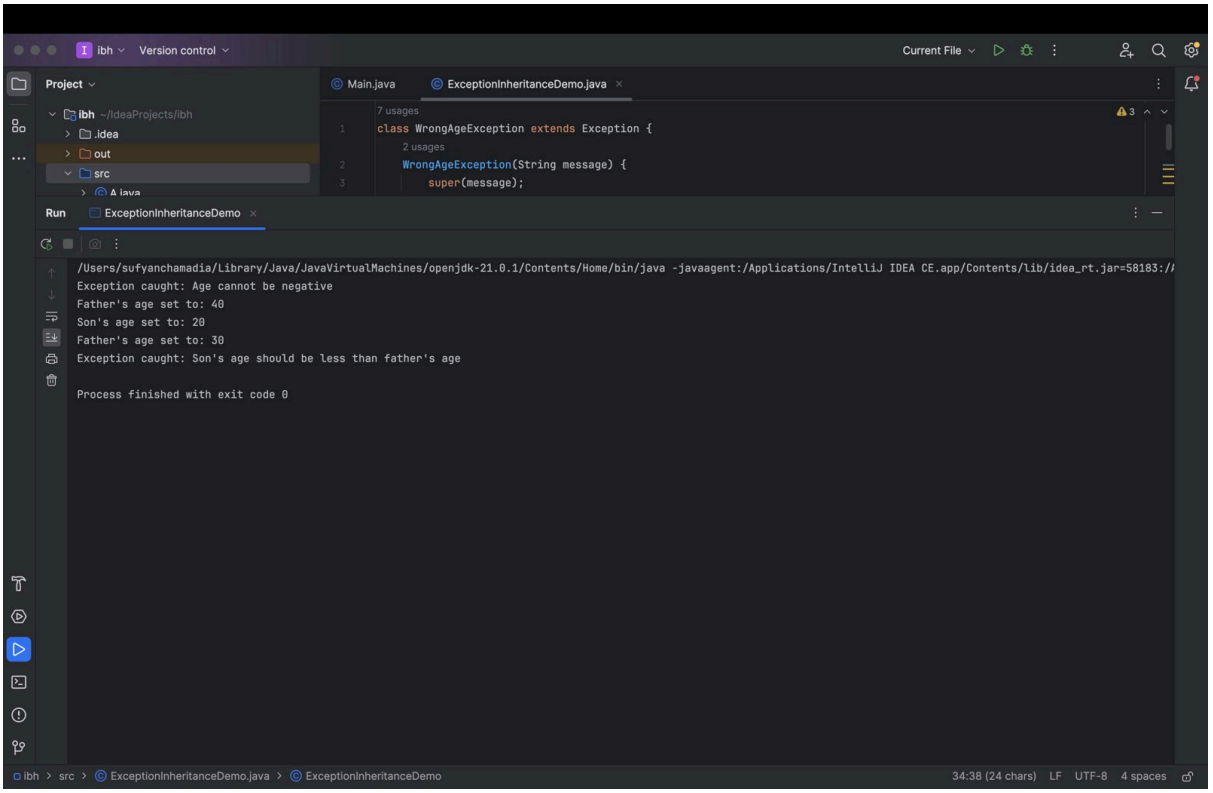
```
        for (int i = 0; i < cie.internalMarks.length; i++) {
            System.out.println("Course " + (i + 1) + ": " + cie.internalMarks[i]);
        }

        System.out.println("\nExternal Marks:");
        for (int i = 0; i < see.externalMarks.length; i++) {
            System.out.println("Course " + (i + 1) + ": " + see.externalMarks[i]);
        }

        // Calculate and display final marks
        int[] finalMarks = calculateFinalMarks(cie.internalMarks, see.externalMarks);
        System.out.println("\nFinal Marks:");
        for (int i = 0; i < finalMarks.length; i++) {
            System.out.println("Course " + (i + 1) + ": " + finalMarks[i]);
        }
    }

    static int[] calculateFinalMarks(int[] internalMarks, int[] externalMarks) {
        int[] finalMarks = new int[internalMarks.length];
        for (int i = 0; i < finalMarks.length; i++) {
            // Assuming a simple average for final marks
            finalMarks[i] = (internalMarks[i] + externalMarks[i]) / 2;
        }
        return finalMarks;
    }
}
```

```
Student Details:
USN: 1BM123
Name: John Doe
Semester: 3

Internal Marks:
80 75 85 90 88

External Marks:
75 70 80 85 78
```

```
class WrongAgeException extends Exception {
    WrongAgeException(String message) {
        super(message);
    }
}

class Father {
    int fatherAge;

    Father(int age) throws WrongAgeException {
        if (age < 0) {
            throw new WrongAgeException("Age cannot be negative");
        }
        fatherAge = age;
        System.out.println("Father's age set to: " + fatherAge);
    }
}

class Son extends Father {
    int sonAge;
```

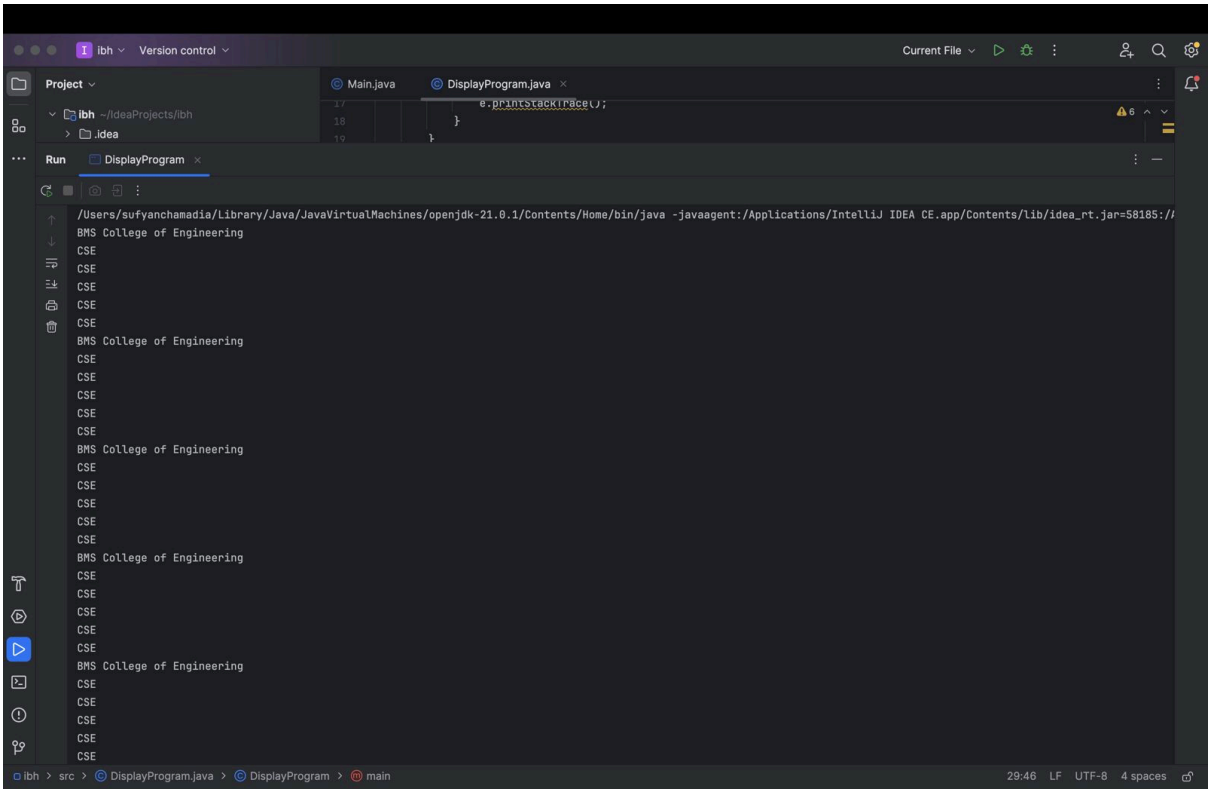
```
    Son(int fatherAge, int sonAge) throws WrongAgeException {
        super(fatherAge);

        if (sonAge >= fatherAge) {
            throw new WrongAgeException("Son's age should be less than father's age");
        }

        this.sonAge = sonAge;
        System.out.println("Son's age set to: " + sonAge);
    }
}

public class ExceptionInheritanceDemo {
    public static void main(String[] args) {
        try {
            // Creating a Father object with negative age (will throw an exception)
            Father father = new Father(-50);
        } catch (WrongAgeException e) {
            System.out.println("Exception caught: " + e.getMessage());
        }

        try {
            // Creating a Son object with valid ages (no exception)
            Son son = new Son(40, 20);
        } catch (WrongAgeException e) {
            System.out.println("Exception caught: " + e.getMessage());
        }
```

```
        try {
            // Creating a Son object with son's age greater than or equal to father's age (will throw an exception)
            Son sonWithWrongAge = new Son(30, 35);
        } catch (WrongAgeException e) {
            System.out.println("Exception caught: " + e.getMessage());
        }
    }
}
```




```
class DisplayThread extends Thread {
    private String message;
    private int sleepTime;

    public DisplayThread(String message, int sleepTime) {
        this.message = message;
        this.sleepTime = sleepTime;
```

```
    }

    @Override
    public void run() {
        while (true) {
            System.out.println(message);
            try {
                Thread.sleep(sleepTime * 1000); // Convert seconds to milliseconds
            } catch (InterruptedException e) {
                e.printStackTrace();
            }
        }
    }
}

public class DisplayProgram {
    public static void main(String[] args) {
        // Create and start the first thread
        DisplayThread thread1 = new DisplayThread("BMS College of Engineering", 10);
        thread1.start();

        // Create and start the second thread
        DisplayThread thread2 = new DisplayThread("CSE", 2);
        thread2.start();
    }
}
```


ibh
Version control
Current File
Project
Main.java
DisplayProgram.java
ibh ~/IdeaProjects/ibh
.idea
e.printStackTrace();
}
}
Run
DisplayProgram
/Users/sufyanchamadia/Library/Java/JavaVirtualMachines/openjdk-21.0.1/Contents/Home/bin/java -javaagent:/Applications/IntelliJ IDEA CE.app/Contents/lib/idea_rt.jar=58185:/
BMS College of Engineering
CSE
CSE
CSE
CSE
CSE
BMS College of Engineering
CSE
CSE
CSE
CSE
CSE
BMS College of Engineering
CSE
CSE
CSE
CSE
CSE
BMS College of Engineering
CSE
CSE
CSE
CSE
CSE
BMS College of Engineering
CSE
CSE
CSE
CSE
CSE
ibh > src > DisplayProgram.java > DisplayProgram > main
29:46 LF UTF-8 4 spaces