Name : Harshali Hemantkumar Patil
Class : SY B-Tech Div : B Roll no. : B-20
PRN : 23410087

GROUP- B

1. **Implement Student class using following Concepts**
   a. All types of Constructors
   b. Static variables and instance variables
   c. Static blocks and instance blocks
   d. Static methods and instance methods

   INPUT :

```
package Group_B;
import java.util.Scanner;

class Student
{
	//Instance variable
	private int roll_no;
	private String name , department,dob;
	private double marks;

	//Static variable
	private static int count;

	//Static Block
	static
	{
		count=0;
		System.out.println("Static Block excecuted");
	}

	//Instance Block
	{
		System.out.println("Instance Block excecuted");
		roll_no=0;
		name="";
		department="";
		dob="";
		marks=0.0;
	}

	public Student()
```

```
	{
		roll_no=0;
		name="";
		department="";
		dob="";
		marks=0.0;
	}

	public Student(int roll_no,String name,String department,String dob,double
marks)
	{
		this.roll_no=roll_no;
		this.name=name;
		this.department=department;
		this.dob=dob;
		this.marks=marks;
	}

	//Instance Method
	public void Percentage()
	{
		double percent=marks/5.0;
		System.out.println("Percentage : "+percent+" %");
	}

	//Static Method
	public static void StudentCount()
	{
		count=count+1;
		System.out.println("The number of Student : "+count);
	}

	public void display()
	{
		System.out.println("***Students details***");
		System.out.println("Roll no. : "+roll_no);
		System.out.println("Name : "+name);
		System.out.println("Department : "+department);
		System.out.println("DOB : "+dob);
		System.out.println("Marks obtained  : "+marks);
	}

}
public class Group_B_1 {
```

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

		System.out.print("Enter the roll number : ");
		int roll=sc.nextInt();
		System.out.print("Enter the name : ");
		String name = sc.next();
		System.out.print("Enter the department : ");
		String department=sc.next();
		System.out.print("Enter the DOB : ");
		String dob=sc.next();
		System.out.print("Enter the marks obtained : ");
		double marks=sc.nextDouble();

		System.out.println();

		Student s=new Student(roll,name,department,dob,marks);
		s.display();
		s.Percentage();
		Student.StudentCount();

		System.out.println();

		Student s1=new Student(2,"Hermonie","Computer","25-12-2005",499);
		s1.display();
		s.Percentage();
		Student.StudentCount();

		System.out.println();

		Student s3=new Student();
		System.out.println("Default constructor called");
		Student.StudentCount();

		sc.close();

	}

}
```

OUTPUT :

```
Enter the roll number : 1
Enter the name : Harry
Enter the department : Computer
Enter the DOB : 25-01-2005
Enter the marks obtained : 490

Static Block excecuted
Instance Block excecuted
***Students details***
Roll no. : 1
Name : Harry
Department : Computer
DOB : 25-01-2005
Marks obtained  : 490.0
Percentage : 98.0 %
The number of Student : 1

Instance Block excecuted
***Students details***
Roll no. : 2
Name : Hermonie
Department : Computer
DOB : 25-12-2005
Marks obtained  : 499.0
Percentage : 98.0 %
The number of Student : 2
```

```
Instance Block excecuted
Default constructor called
The number of Student : 3
```

1.
2.
3.
4.
5.
6.

Name : Harshali Hemantkumar Patil
Class : SY B-Tech Div : B Roll no. : B-20
PRN : 23410087

GROUP- B

Create a class with overloaded methods for performing mathematical operations (addition, subtraction, multiplication, and division) for both integers and doubles. Demonstrate these overloaded methods with different data types.

INPUT :

```
package Group_B;
import java.util.Scanner;
public class MathOperations {
        public int add(int a, int b) {
    return a + b;
  }
  public double add(double a, double b) {
    return a + b;
  }
  public int subtract(int a, int b) {
    return a - b;
  }
  public double subtract(double a, double b) {
    return a - b;
  }
  public int multiply(int a, int b) {
    return a * b;
  }
  public double multiply(double a, double b) {
    return a * b;
  }
  public int divide(int a, int b) {
    if (b == 0) {
      System.out.println("Error! Division by zero.");
      return 0;
    }
    return a / b;
  }
  public double divide(double a, double b) {
    if (b == 0) {
      System.out.println("Error! Division by zero.");
      return 0.0;
```

```
    }
    return a / b;
}

public static void main(String[] args) {
        Scanner scanner = new Scanner(System.in);
    MathOperations mathOps = new MathOperations();
    boolean continueOperations = true;
    while (continueOperations) {
    System.out.println("Choose the type of operation: ");
    System.out.println("1. Integer Operation");
    System.out.println("2. Double Operation");
    System.out.print("Enter your choice (1 or 2): ");
    int choice = scanner.nextInt();
    if (choice == 1) {

        System.out.print("Enter first integer: ");
        int num1 = scanner.nextInt();
        System.out.print("Enter second integer: ");
        int num2 = scanner.nextInt();
        System.out.println("\nInteger Operations:");
        System.out.println("Addition: " + mathOps.add(num1, num2));
        System.out.println("Subtraction: " + mathOps.subtract(num1, num2));
        System.out.println("Multiplication: " + mathOps.multiply(num1, num2));
        System.out.println("Division: " + mathOps.divide(num1, num2));
    } else if (choice == 2) {

        System.out.print("Enter first double: ");
        double num1 = scanner.nextDouble();
        System.out.print("Enter second double: ");
        double num2 = scanner.nextDouble();
        System.out.println("\nDouble Operations:");
        System.out.println("Addition: " + mathOps.add(num1, num2));
        System.out.println("Subtraction: " + mathOps.subtract(num1, num2));
        System.out.println("Multiplication: " + mathOps.multiply(num1, num2));
        System.out.println("Division: " + mathOps.divide(num1, num2));
    } else {
        System.out.println("Invalid choice. Please enter 1 or 2.");
        continue;
    }

    System.out.print("\nDo you want to perform another operation? (yes/no): ");
    String continueChoice = scanner.next();
    if (continueChoice.equalsIgnoreCase("no")) {
```

```
                continueOperations = false;
            }
            }
            System.out.println("Thank you for using the calculator. ");
            scanner.close();
        }
}
```

OUTPUT :

```
Choose the type of operation:
1. Integer Operation
2. Double Operation
Enter your choice (1 or 2): 1
Enter first integer: 10
Enter second integer: 20

Integer Operations:
Addition: 30
Subtraction: -10
Multiplication: 200
Division: 0

Do you want to perform another operation? (yes/no): yes
Choose the type of operation:
1. Integer Operation
2. Double Operation
Enter your choice (1 or 2): 2
Enter first double: 1.1
Enter second double: 2.5

Double Operations:
Addition: 3.6
Subtraction: -1.4
Multiplication: 2.75
Division: 0.44000000000000006

Do you want to perform another operation? (yes/no): no
Thank you for using the calculator.
```

Name : Harshali Hemantkumar Patil
Class : SY B-Tech Div : B Roll no. : B-20
PRN : 23410087

GROUP- B

Write Java program to calculate area of triangle, square and circle using function overloading. Function parameter accept from user (Use function Overloading concepts and Inheritance).

INPUT :

```
import java.util.Scanner;
public abstract class Shape {
    public abstract double calculateArea();
    public void display() {
        System.out.println("Area : "+calculateArea());
    }
    public static void main(String[] args) {
        Scanner scanner = new Scanner(System.in);
        System.out.print("Enter the radius of circle : ");
        double radius = scanner.nextDouble();
        Circle c = new Circle(radius);
        c.display();
        System.out.print("Enter the side of the square : ");
        double side = scanner.nextDouble();
        Square sq = new Square(side);
        sq.display();
        System.out.print("Enter the base of triangle : ");
        double base = scanner.nextDouble();
        System.out.print("Enter the height of triangle : ");
        double height = scanner.nextDouble();
        Triangle t = new  Triangle(base,height);
```

```
            t.display();
        }
    }
    class Circle extends Shape{
        private double radius;

        Circle(double radius){
            this.radius=radius;
        }

        public double calculateArea() {
            return Math.PI*radius*radius;
        }
    }
    class Square extends Shape{
        private double side;

        Square(double side){
            this.side=side;
        }

        public double calculateArea() {
            return side*side;
        }
    }
    class Triangle extends Shape{
        private double base;
        private double height;

        Triangle(double base,double height){
            this.base=base;
            this.height=height;
        }

        public double calculateArea() {
            return 0.5*base*height;
        }
    }
```

OUTPUT :

```
Enter the radius of circle : 5.0
Area : 78.53981633974483
Enter the side of the square : 10.0
Area : 100.0
Enter the base of triangle : 2.1
Enter the height of triangle : 1.1
Area : 1.1550000000000002
```

Name : Harshali Hemantkumar Patil
Class : SY B-Tech Div : B Roll no. : B-20
PRN : 23410087

GROUP- B

Design an abstract class "Shape" with abstract methods for calculating area and perimeter. Create subclasses for specific shapes (e.g., Circle, Rectangle, Triangle) that extend the abstract class and implement these Methods.

INPUT :

```
import java.util.Scanner;
abstract class Shape {
  abstract double getArea();
  abstract double getPerimeter();
}
class Circle extends Shape {
  private double radius;
  public Circle(double radius) {
    this.radius = radius;
  }
  double getArea() {
    return Math.PI * radius * radius;
  }
  double getPerimeter() {
    return 2 * Math.PI * radius;
  }
}
class Rectangle extends Shape {
  private double length, width;
  public Rectangle(double length, double width) {
    this.length = length;
    this.width = width;
  }
  double getArea() {
    return length * width;
  }
  double getPerimeter() {
    return 2 * (length + width);
  }
}
class Triangle extends Shape {
  private double sideA, sideB, sideC;
```

```
  public Triangle(double sideA, double sideB, double sideC) {
    this.sideA = sideA;
    this.sideB = sideB;
    this.sideC = sideC;
  }
  double getArea() {
    double s = (sideA + sideB + sideC) / 2; // Semi-perimeter
    return Math.sqrt(s * (s - sideA) * (s - sideB) * (s - sideC));
  }
  double getPerimeter() {
    return sideA + sideB + sideC;
  }
}
public class Main {
  public static void main(String[] args) {
    Scanner scanner = new Scanner(System.in);

    System.out.print("Enter radius of circle: ");
    double radius = scanner.nextDouble();
    Shape circle = new Circle(radius);
    System.out.println("Circle - Area: " + circle.getArea() + "\nPerimeter: " +
circle.getPerimeter());

    System.out.print("Enter length and width of rectangle: ");
    double length = scanner.nextDouble();
    double width = scanner.nextDouble();
    Shape rectangle = new Rectangle(length, width);
    System.out.println("Rectangle - Area: " + rectangle.getArea() + "\nPerimeter:
" + rectangle.getPerimeter());

    System.out.print("Enter sides of triangle: ");
    double sideA = scanner.nextDouble();
    double sideB = scanner.nextDouble();
    double sideC = scanner.nextDouble();
    Shape triangle = new Triangle(sideA, sideB, sideC);
    System.out.println("Triangle - Area: " + triangle.getArea() + "\nPerimeter: " +
triangle.getPerimeter());

    scanner.close();
  }
}
```

OUTPUT :

```
Enter radius of circle: 10
Circle - Area: 314.1592653589793
Perimeter: 62.83185307179586
Enter length and width of rectangle: 5 4
Rectangle - Area: 20.0
Perimeter: 18.0
Enter sides of triangle: 3 4 5
Triangle - Area: 6.0
Perimeter: 12.0
```

Name : Harshali Hemantkumar Patil
Class : SY B-Tech Div : B Roll no. : B-20
PRN : 23410087

GROUP- B

Define an interface Employee with methods for calculating salary and displaying employee information. Create classes that implement this interface for different types of employees (e.g., FullTimeEmployee, PartTimeEmployee).

INPUT :

```
interface Employee {

  double calculateSalary();
  void displayInfo();
}
class FullTimeEmployee implements Employee {
  private String name;
  private double annualSalary;
  public FullTimeEmployee(String name, double annualSalary) {
     this.name = name;
     this.annualSalary = annualSalary;
  }
  public double calculateSalary() {
     return annualSalary;
  }
  public void displayInfo() {
     System.out.println("Full-Time Employee");
     System.out.println("Name: " + name);
     System.out.println("Annual Salary: Rs " + annualSalary);
  }
}
class PartTimeEmployee implements Employee {
  private String name;
  private double hourlyRate;
  private double hoursWorked;
  public PartTimeEmployee(String name, double hourlyRate, double
hoursWorked) {
     this.name = name;
     this.hourlyRate = hourlyRate;
     this.hoursWorked = hoursWorked;
  }
  public double calculateSalary() {
```

```
        return hourlyRate * hoursWorked;
    }
    public void displayInfo() {
        System.out.println("Part-Time Employee");
        System.out.println("Name: " + name);
        System.out.println("Hourly Rate: Rs " + hourlyRate);
        System.out.println("Hours Worked: " + hoursWorked);
    }
}
public class EmployeeTest {
    public static void main(String[] args) {

        Employee fullTimeEmployee = new FullTimeEmployee("Alice", 60000);
        Employee partTimeEmployee = new PartTimeEmployee("Bob", 20, 25);
        System.out.println("Employee 1: ");
        fullTimeEmployee.displayInfo();
        System.out.println("Salary: Rs " + fullTimeEmployee.calculateSalary());
        System.out.println("\nEmployee 2: ");
        partTimeEmployee.displayInfo();
        System.out.println("Salary: Rs " + partTimeEmployee.calculateSalary());
    }
}
```

OUTPUT :

```
Employee 1:
Full-Time Employee
Name: Alice
Annual Salary: Rs 60000.0
Salary: Rs 60000.0

Employee 2:
Part-Time Employee
Name: Bob
Hourly Rate: Rs 20.0
Hours Worked: 25.0
Salary: Rs 500.0
```

Name : Harshali Hemantkumar Patil
Class : SY B-Tech Div : B Roll no. : B-20
PRN : 23410087

GROUP- B

Write a program for following exception, develop a suitable scenario in which the following exceptions occur:
a. divide by zero
b. Array index out of bounds exception
c. Null pointer Exception

INPUT :

```
public class RestaurantOrderSystem {
  public static void main(String[] args) {
    try {
      double totalRevenue = 1200;
      int numberOfCustomers = 0;

      if (numberOfCustomers == 0) {
        System.out.println("No customers, cannot calculate average
spending.");
      } else {
        double avgSpending = calculateAverageSpending(totalRevenue,
numberOfCustomers);
        System.out.println("Average spending per customer: " + avgSpending);
      }
    } catch (ArithmeticException e) {
      System.out.println("Exception occurred: " + e.toString() + " - Cannot divide
by zero (No customers).");
    }
    try {
      String[] orders = {"Order 1: Pizza", "Order 2: Burger", "Order 3: Pasta"};
      int index = 5;
      if (index >= 0 && index < orders.length) {
        String order = orders[index];
        System.out.println("Accessing Order: " + order);
      } else {
        System.out.println("Invalid order index.");
      }
    } catch (ArrayIndexOutOfBoundsException e) {
      System.out.println("Exception occurred: " + e.toString() + " - Invalid order
index.");
    }
```

```
        try {
            Order order = null;

            if (order != null) {
                System.out.println("Order details: " + order.getDetails());
            } else {
                System.out.println("No order placed.");
            }
        } catch (NullPointerException e) {
            System.out.println("Exception occurred: " + e.toString() + " - No order
placed.");
        }
    }
    public static double calculateAverageSpending(double totalRevenue, int
numberOfCustomers) {
        return totalRevenue / numberOfCustomers;
    }
    static class Order {
        private String orderID;
        private String foodItem;
        public Order(String orderID, String foodItem) {
            this.orderID = orderID;
            this.foodItem = foodItem;
        }
        public String getDetails() {
            return "Order ID: " + orderID + ", Food Item: " + foodItem;
        }
    }
}
```

OUTPUT :

```
No customers, cannot calculate average spending.
Invalid order index.
No order placed.
```

Name : Harshali Hemantkumar Patil
Class : SY B-Tech Div : B Roll no. : B-20
PRN : 23410087

GROUP- B

Write a program for following exception, develop a suitable scenario in which the following exceptions occur:
a. divide by zero
b. Array index out of bounds exception
c. Null pointer Exception

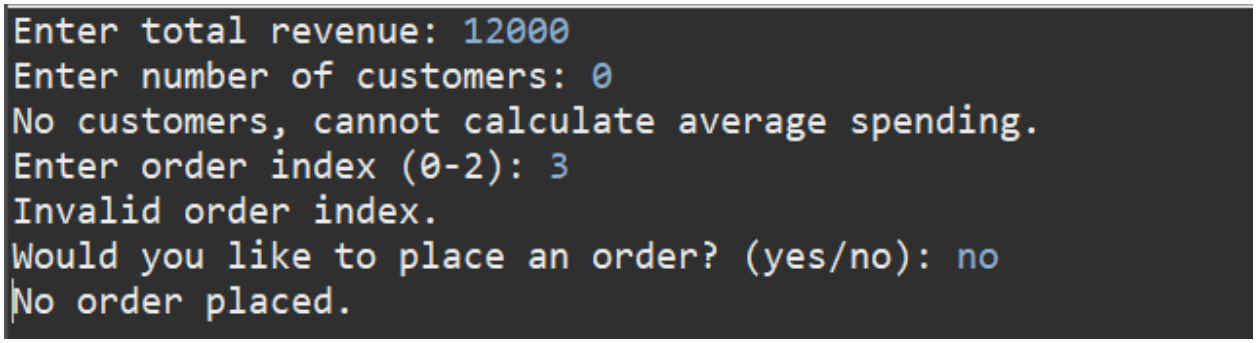
INPUT :

```
import java.util.Scanner;
public class RestaurantOrderSystem {
  public static void main(String[] args) {
     Scanner scanner = new Scanner(System.in);

     try {
        System.out.print("Enter total revenue: ");
        double totalRevenue = scanner.nextDouble();

        System.out.print("Enter number of customers: ");
        int numberOfCustomers = scanner.nextInt();

        if (numberOfCustomers == 0) {
           System.out.println("No customers, cannot calculate average
spending.");
        } else {
           double avgSpending = calculateAverageSpending(totalRevenue,
numberOfCustomers);
           System.out.println("Average spending per customer: " + avgSpending);
        }
     } catch (ArithmeticException e) {
        System.out.println("Exception occurred: " + e.toString() + " - Cannot divide
by zero (No customers).\n");
     }

     try {
        String[] orders = {"Order 1: Pizza", "Order 2: Burger", "Order 3: Pasta"};

        System.out.print("Enter order index (0-2): ");
        int index = scanner.nextInt();
```

```
            if (index >= 0 && index < orders.length) {
                String order = orders[index];
                System.out.println("Accessing Order: " + order);
            } else {
                System.out.println("Invalid order index.");
            }
        } catch (ArrayIndexOutOfBoundsException e) {
            System.out.println("Exception occurred: " + e.toString() + " - Invalid order
index.\n");
        }

        try {
            Order order = null;

            System.out.print("Would you like to place an order? (yes/no): ");
            scanner.nextLine();
            String choice = scanner.nextLine();

            if (choice.equalsIgnoreCase("yes")) {
                System.out.print("Enter order ID: ");
                String orderID = scanner.nextLine();
                System.out.print("Enter food item: ");
                String foodItem = scanner.nextLine();
                order = new Order(orderID, foodItem);
            }

            if (order != null) {
                System.out.println("Order details: " + order.getDetails());
            } else {
                System.out.println("No order placed.");
            }
        } catch (NullPointerException e) {
            System.out.println("Exception occurred: " + e.toString() + " - No order
placed.");
        }

        scanner.close();
    }

    public static double calculateAverageSpending(double totalRevenue, int
numberOfCustomers) {
        return totalRevenue / numberOfCustomers;
    }
```

```
    static class Order {
        private String orderID;
        private String foodItem;

        public Order(String orderID, String foodItem) {
            this.orderID = orderID;
            this.foodItem = foodItem;
        }

        public String getDetails() {
            return "Order ID: " + orderID + ", Food Item: " + foodItem;
        }
    }
}
```

OUTPUT :



Name : Harshali Hemantkumar Patil
Class : SY B-Tech Div : B Roll no. : B-20
PRN : 23410087

GROUP- B

Build a class hierarchy for bank accounts (e.g., SavingsAccount, CheckingAccount) with a common base class BankAccount. Override the calculateInterest() method in the derived classes to calculate interest based on specific account types.

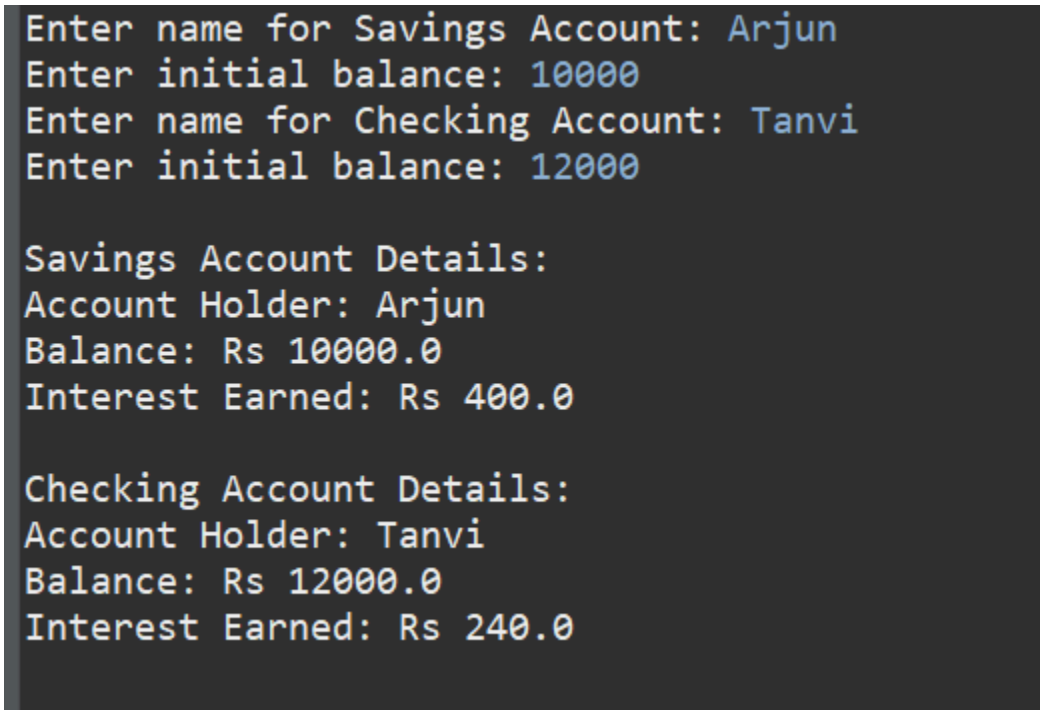
INPUT :

```
import java.util.Scanner;
abstract class BankAccount {
  protected String accountHolder;
  protected double balance;

  public BankAccount(String accountHolder, double balance) {
     this.accountHolder = accountHolder;
     this.balance = balance;
  }
  public abstract double calculateInterest();

  public void deposit(double amount) {
     balance += amount;
     System.out.println("Deposited: Rs " + amount);
  }

  public void withdraw(double amount) {
     if (amount > balance) {
        System.out.println("Insufficient balance!");
     } else {
        balance -= amount;
        System.out.println("Withdrawn: Rs " + amount);
     }
  }

  public void displayAccountInfo() {
     System.out.println("Account Holder: " + accountHolder);
     System.out.println("Balance: Rs " + balance);
     System.out.println("Interest Earned: Rs " + calculateInterest());
  }
}
class SavingsAccount extends BankAccount {
```

```
    private static final double INTEREST_RATE = 0.04; // 4% annual interest

    public SavingsAccount(String accountHolder, double balance) {
        super(accountHolder, balance);
    }

    public double calculateInterest() {
        return balance * INTEREST_RATE;
    }
}
class CheckingAccount extends BankAccount {
    private static final double INTEREST_RATE = 0.02; // 2% annual interest

    public CheckingAccount(String accountHolder, double balance) {
        super(accountHolder, balance);
    }

    public double calculateInterest() {
        return balance * INTEREST_RATE;
    }
}
public class BankApp {
    public static void main(String[] args) {
        Scanner scanner = new Scanner(System.in);

        System.out.print("Enter name for Savings Account: ");
        String savName = scanner.nextLine();
        System.out.print("Enter initial balance: ");
        double savBalance = scanner.nextDouble();
        SavingsAccount savings = new SavingsAccount(savName, savBalance);

        System.out.print("Enter name for Checking Account: ");
        scanner.nextLine();
        String chkName = scanner.nextLine();
        System.out.print("Enter initial balance: ");
        double chkBalance = scanner.nextDouble();
        CheckingAccount checking = new CheckingAccount(chkName, chkBalance);

        System.out.println("\nSavings Account Details:");
        savings.displayAccountInfo();

        System.out.println("\nChecking Account Details:");
        checking.displayAccountInfo();
```

```
        scanner.close();
    }
}
```

OUTPUT :



Name : Harshali Hemantkumar Patil
Class : SY B-Tech Div : B Roll no. : B-20
PRN : 23410087

GROUP- B

Design a Java program for a library catalog system. Create overloaded methods for adding and searching for books in the library. Overload the methods to accept various search criteria (e.g., title, author, ISBN) and demonstrate their usage in managing the library catalog.

INPUT :

```
import java.util.Scanner;
class Book {
 String title;
 String author;
 int ISBN;
 public Book() {
    this.title = "";
    this.author = "";
    this.ISBN = 0;
 }
 public Book(String title, String author, int ISBN) {
    this.title = title;
    this.author = author;
    this.ISBN = ISBN;
 }
 public void display() {
    System.out.println("Title: " + title);
    System.out.println("Author: " + author);
    System.out.println("ISBN: " + ISBN);
```

```
}
public static void main(String[] args) {
    Scanner scanner = new Scanner(System.in);
    Library library = new Library(5);
    library.addBook(new Book("Let us C", "Yashwant Kanitkar", 101));
    library.addBook("Let us C++", "Yashwant Kanitkar", 102);
    library.addBook("Harry Potter", "J K Rowling", 103);
    boolean running = true;
    while (running) {
        System.out.println("\nLibrary Catalog System Menu:");
        System.out.println("1. Add a new book");
        System.out.println("2. Search books by title");
        System.out.println("3. Search books by author");
        System.out.println("4. Search books by ISBN");
        System.out.println("5. Display all books");
        System.out.println("6. Exit");
        System.out.print("Enter your choice: ");
        int choice = scanner.nextInt();
        scanner.nextLine();
        switch (choice) {
            case 1:
                System.out.print("Enter book title: ");
                String title = scanner.nextLine();
                System.out.print("Enter author name: ");
                String author = scanner.nextLine();
                System.out.print("Enter ISBN: ");
```

```
            int ISBN = scanner.nextInt();
            library.addBook(title, author, ISBN);
            System.out.println("Book added successfully!");
            break;
        case 2:
            System.out.print("Enter the title of the book: ");
            String searchTitle = scanner.nextLine();
            library.searchBookByTitle(searchTitle);
            break;
        case 3:
            System.out.print("Enter the author's name: ");
            String searchAuthor = scanner.nextLine();
            library.searchBookByAuthor(searchAuthor);
            break;
        case 4:
            System.out.print("Enter the ISBN of the book: ");
            int searchISBN = scanner.nextInt();
            library.searchBookByISBN(searchISBN);
            break;
        case 5:
            library.displayAllBooks();
            break;
        case 6:
            System.out.println("Exiting Library Catalog System.");
            running = false;
            break;
```

```
            default:
                System.out.println("Invalid choice. Please try again.");
        }
    }
    scanner.close();
 }
}
class Library {
 private Book[] catalog;
 private int bookCount;
 public Library(int size) {
    catalog = new Book[size];
    bookCount = 0;
 }
 public void addBook(Book book) {
    if (bookCount < catalog.length) {
        catalog[bookCount] = book;
        bookCount++;
    } else {
        System.out.println("Library catalog is full. Cannot add more books.");
    }
 }
 public void addBook(String title, String author, int ISBN) {
    if (bookCount < catalog.length) {
        catalog[bookCount] = new Book(title, author, ISBN);
        bookCount++;
```

```
    } else {
        System.out.println("Library catalog is full. Cannot add more books.");
    }
}
public void searchBookByTitle(String title) {
    boolean found = false;
    for (int i = 0; i < bookCount; i++) {
        if (catalog[i].title.equalsIgnoreCase(title)) {
            System.out.println("\nBook found:");
            catalog[i].display();
            found = true;
        }
    }
    if (!found) {
        System.out.println("No book found with the title: " + title);
    }
}
public void searchBookByAuthor(String author) {
    boolean found = false;
    for (int i = 0; i < bookCount; i++) {
        if (catalog[i].author.equalsIgnoreCase(author)) {
            System.out.println("\nBook found:");
            catalog[i].display();
            found = true;
        }
    }
```

```
    if (!found) {
        System.out.println("No books found by author: " + author);
    }
}
public void searchBookByISBN(int ISBN) {
    boolean found = false;
    for (int i = 0; i < bookCount; i++) {
        if (catalog[i].ISBN == ISBN) {
            System.out.println("\nBook found:");
            catalog[i].display();
            found = true;
        }
    }
    if (!found) {
        System.out.println("No book found with ISBN: " + ISBN);
    }
}
public void displayAllBooks() {
    if (bookCount == 0) {
        System.out.println("The catalog is empty.");
    } else {
        System.out.println("\nLibrary Catalog:");
        for (int i = 0; i < bookCount; i++) {
            catalog[i].display();
            System.out.println("--------------------");
        }
```

```
    }
  }
}
```

OUTPUT :

```
Library Catalog System Menu:
1. Add a new book
2. Search books by title
3. Search books by author
4. Search books by ISBN
5. Display all books
6. Exit
Enter your choice: 5

Library Catalog:
Title: Let us C
Author: Yashwant Kanitkar
ISBN: 101
--------------------
Title: Let us C++
Author: Yashwant Kanitkar
ISBN: 102
--------------------
Title: Harry Potter
Author: J K Rowling
ISBN: 103
--------------------
```

```
Library Catalog System Menu:
1. Add a new book
2. Search books by title
3. Search books by author
4. Search books by ISBN
5. Display all books
6. Exit
Enter your choice: 2
Enter the title of the book: Let us C

Book found:
Title: Let us C
Author: Yashwant Kanitkar
ISBN: 101

Library Catalog System Menu:
1. Add a new book
2. Search books by title
3. Search books by author
4. Search books by ISBN
5. Display all books
6. Exit
Enter your choice: 4
Enter the ISBN of the book: 101
```

```
Book found:
Title: Let us C
Author: Yashwant Kanitkar
ISBN: 101

Library Catalog System Menu:
1. Add a new book
2. Search books by title
3. Search books by author
4. Search books by ISBN
5. Display all books
6. Exit
Enter your choice: 6
Exiting Library Catalog System.
```