

/\*1> Create a class representing a Car with attributes like make, model, and year. Implement methods to set and get these attributes. \*/

```
class Cars {
    private String carMake; // car's make?
    private String carModel;// car's model

    private int carYear;// year of manufacture

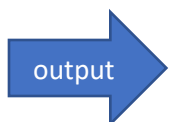
    public void setCar(String newMake, String newModel, int year) {
        this.carMake = newMake;
        this.carModel = newModel;
        this.carYear = year;
    }

    public String getCarMake() {
        return (this.carMake);
    }

    public String getCarModel() {
        return (this.carModel);
    }

    public int getCarYear() {
        return (this.carYear);
    }
}

public class _1_one {
    public static void main(String[] args) {
        Cars obj = new Cars();
        // setter
        obj.setCar("BMW", "BMW20230.2", 2023);
        System.out.println("The Make is: " + obj.getCarMake());
        System.out.println("The model is: " + obj.getCarModel());
        System.out.println("The yaer is: " + obj.getCarYear());
    }
}
```



```
The Make is: BMW
The model is: BMW20230.2
The yaer is: 2023
```

```

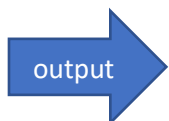
/* 2> Implement inheritance by creating a subclass of Car, such as
 * ElectricCar, with additional attributes and methods.*/
class Car {
    String name;
    String manufactured;
    int YearOfmanufactured;
}

class ElectricCar extends Car {
    boolean isElectric = true;
    boolean Driver;
    int numberOfBatteries;
    int chargingType;

    void printAboutElectricCar() {
        System.out.println("This car is electric" + " Non-Driver/Driver ");
    }
}

public class _2_two {
    public static void main(String[] args) {
        // Create an instance of the ElectricCar Class:
        ElectricCar eCar = new ElectricCar();
        /*
         * Set values for all variables in the ElectricCar object
         */
        eCar.name = "Tesla";
        eCar.YearOfmanufactured = 1986;
        eCar.chargingType = 30;
        if (eCar instanceof ElectricCar) {
            ((ElectricCar) (eCar)).printAboutElectricCar();
        } else {
            System.out.println("Not an electric car");
        }
        ;
        // Print out each variable value stored inside the ElectricCar Object:
        System.out.printf("%s %d",
            eCar.name,
            eCar.YearOfmanufactured);
        System.out.println("\n\n");
    }
}

```



```

This car is electric Non-Driver/Driver
Tesla 1986

```

//3> Demonstrate the concept of method overriding using a base class and a derived class.

```
class Shapes {
    public double calculateArea() {
        return 0.0;
    }
}

class Circle extends Shapes {
    private double radius;

    Circle(double r) {
        radius = r;
    }

    public double calculateArea() {
        return (3.14 * Math.pow((radius), 2));
    }
}

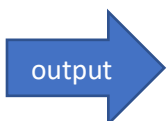
class Rectangle extends Shapes {
    private double length, width;

    Rectangle(double length, double width) {
        this.length = length;
        this.width = width;
    }

    public double calculateArea() {
        return length * width;
    }
}

public class _3_three {
    public static void main(String[] args) {
        Rectangle rectObj = new Rectangle(8.9, 8.6);
        Circle circObj = new Circle(5.57);
        System.out.println("Area of Rectangle is " + rectObj.calculateArea());

        System.out.println("Area of Circle is " + circObj.calculateArea());
    }
}
```



```
Area of Rectangle is 76.54
Area of Circle is 97.418186
```

```
//4> Write a java program to implement Abstract method (no implementation)
abstract class Shape {
    abstract double calculateArea();
    double calculatePerimeter() {
        return 0.0;
    }
}

class Circle extends Shape {
    private double radius;

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

    double calculateArea() {
        return Math.PI * radius * radius;
    }

    double calculatePerimeter() {
        return 2 * 3.14 * radius;
    }
}

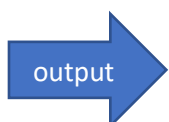
class Rectangle extends Shape {
    private double length;
    private double width;

    Rectangle(double length, double width) {
        this.length = length;
        this.width = width;
    }

    double calculateArea() {
        return length * width;
    }

    // i have never implement calculatePerimeter method because it is not be must
    // because it not be abstract method.
}

public class _4_four {
    public static void main(String[] args) {
        Circle circle = new Circle(5.0);
        Rectangle rectangle = new Rectangle(4.0, 6.0);
        System.out.println("Circle Area: " + circle.calculateArea());
        System.out.println("Circle Perimeter: " + circle.calculatePerimeter());
        System.out.println("Rectangle Area: " + rectangle.calculateArea());
        System.out.println("Rectangle Perimeter: " + rectangle.calculatePerimeter());
    }
}
```



```
Circle Area: 78.53981633974483
Circle Perimeter: 31.400000000000002
Rectangle Area: 24.0
Rectangle Perimeter: 0.0
```

```
//5> Define an interface Calculator with methods for addition, subtraction,
multiplication, and division. Implement this interface in a CalculatorImpl class.
// Define the Calculator interface
interface Calculator {
    double add(double num1, double num2);
    double subtract(double num1, double num2);
    double multiply(double num1, double num2);
    double divide(double num1, double num2);
}

class CalculatorImpl implements Calculator {
    @Override // this is not be must
    public double add(double num1, double num2) {
        return num1 + num2;
    }
    public double subtract(double num1, double num2) {
        return num1 - num2;
    }
    @Override
    public double multiply(double num1, double num2) {
        return num1 * num2;
    }
    @Override
    public double divide(double num1, double num2) {
        if (num2 == 0) {
            throw new IllegalArgumentException("Division by zero is not allowed.");
        }
        return num1 / num2;
    }
}

public class _5_five {
    public static void main(String[] args) {
        Calculator calculator = new CalculatorImpl();
        double num1 = 10.0;
        double num2 = 5.0;
        System.out.println("Addition: " + calculator.add(num1, num2));
        System.out.println("Subtraction: " + calculator.subtract(num1, num2));
        System.out.println("Multiplication: " + calculator.multiply(num1, num2));
        try {
            System.out.println("Division: " + calculator.divide(num1, num2));
        } catch (IllegalArgumentException e) {
            System.err.println(e.getMessage());
        }
    }
}
```

output

```
Addition: 15.0
Subtraction: 5.0
Multiplication: 50.0
Division: 2.0
```

//6> create a package named "student" containing classes Student and Teacher. Import and use these classes in another package.

```
class InsufficientFundsException extends Exception {
    public InsufficientFundsException(String message) {
        super(message); // called exception class constructor
    }
}

class BankAccount {
    private double balance;
    public BankAccount(double initialBalance) {balance = initialBalance;}
    public void withdraw(double amount) throws InsufficientFundsException {
        if (amount > balance) {
            throw new InsufficientFundsException("Insufficient funds. Current
balance: " + balance);
        }
        balance -= amount;
        System.out.println("Withdrawal successful. Remaining balance: " + balance);
    }
}

public class _6_ {
    public static void main(String[] args) {
        BankAccount account = new BankAccount(1000.0);
        try {
            account.withdraw(500.0); // Valid withdrawal
            account.withdraw(700.0); // This will throw InsufficientFundsException
        } catch (InsufficientFundsException e) {
            System.err.println("Error: " + e.getMessage());
        }
    }
}
```

output

```
Withdrawal successful. Remaining balance: 500.0
Error: Insufficient funds. Current balance: 500.0
```

//7> Implement a try-catch block to handle an ArrayIndexOutOfBoundsException when accessing an out-of-bounds index in an array.

```
public class _7_seven {
    public static void main(String[] args) {
        int[] numbers = { 1, 2, 3 };
        try {
            int element = numbers[5]; // Attempt to access an out-of-bounds index
            System.out.println("Element: " + element);
        } catch (ArrayIndexOutOfBoundsException e) {
            // Handle the exception
            System.err.println("Error: " + e.getMessage());
        }
        System.out.println("The program continues after the exception handling.");
    }
}
```

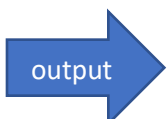
output

```
Error: Index 5 out of bounds for length 3
The program continues after the exception handling.
```

//8>Create two threads that print even and odd numbers using synchronized methods.

```
class NumberPrinter {
    private int count = 1;
    private int maxCount;
    public NumberPrinter(int maxCount) {
        this.maxCount = maxCount;
    }
    public synchronized void printEven() {
        while (count <= maxCount) {
            if (count % 2 == 0) {
                System.out.println("Even: " + count);
                count++;
                notify(); // Notify the other thread to run
            } else {
                try {
                    wait(); // Wait for the other thread to notify
                } catch (InterruptedException e) {
                    Thread.currentThread().interrupt();
                }
            }
        }
    }
    public synchronized void printOdd() {
        while (count <= maxCount) {
            if (count % 2 != 0) {
                System.out.println("Odd: " + count);
                count++;
                notify(); // Notify the other thread to run
            } else {
                try {
                    wait(); // Wait for the other thread to notify
                } catch (InterruptedException e) {
                    Thread.currentThread().interrupt();
                }
            }
        }
    }
}

public class _8_Threads {
    public static void main(String[] args) {
        NumberPrinter numberPrinter = new NumberPrinter(5);
        Thread evenThread = new Thread(() -> {numberPrinter.printEven();});
        Thread oddThread = new Thread(() -> {numberPrinter.printOdd();});
        evenThread.start();
        oddThread.start();
    }
}
```

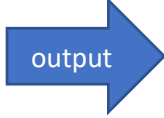


Odd: 1  
Even: 2  
Odd: 3  
Even: 4  
Odd: 5

//9> Build a simple console-based calculator application that takes user input for two numbers and an operator, then performs the operation.

```
class calculator {
    public int add(int n, int m) {return n + m;}
    public int sub(int n, int m) {return n - m;}
    public int mult(int n, int m) {
        return n * m;
    }
    public double div(double n, double m) {
        return n / m;
    }
}

public class _9_Calculator {
    public static void main(String[] args) {
        calculator calculate = new calculator();
        System.out.println("add: " + calculate.add(8, 9));
        System.out.println("sub: " + calculate.sub(15, 9));
        System.out.println("div: " + calculate.div(15, 2));
        System.out.println("mult: " + calculate.mult(7, 5));
    }
}
```



```
add: 17
sub: 6
div: 7.5
mult: 35
```

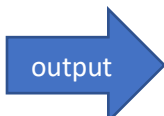
//10> Wap to create a file in java

```
import java.io.File;
import java.io.IOException;

public class _10_FileHandling {
    public static void main(String[] args) {
        String filePath = "temp.txt"; // Specify the file path and name

        try {
            File file = new File(filePath);

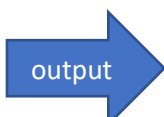
            if (file.createNewFile()) {
                System.out.println("File created successfully.");
            } else {
                System.out.println("File already exists.");
            }
        } catch (IOException e) {
            System.err.println("Error creating the file: " + e.getMessage());
        }
    }
}
```



```
File created successfully.
PS C:\Users\C Kumar\Desktop\CollegePractical\Java>
```



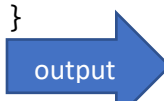
```
//11> Write a java program to write data in file and read it from file.
import java.io.BufferedReader;
import java.io.BufferedWriter;
import java.io.FileInputStream;
import java.io.FileOutputStream;
import java.io.IOException;
import java.io.InputStreamReader;
import java.io.OutputStreamWriter;
public class _11_ReadAndWrite {
    public static void main(String[] args) {
        String filePath = "student.txt"; // Specify the file path and name
        String dataToWrite = "Hello guys, I am student of Munger University";
        try (FileOutputStream fos = new FileOutputStream(filePath);
            OutputStreamWriter osw = new OutputStreamWriter(fos);
            BufferedWriter writer = new BufferedWriter(osw)) {
            writer.write(dataToWrite);
            System.out.println("Data written to the file successfully.");
        } catch (IOException e) {
            System.err.println("Error writing to the file: " + e.getMessage());
        }
        // Read file
        try (FileInputStream fis = new FileInputStream(filePath);
            InputStreamReader isr = new InputStreamReader(fis);
            BufferedReader reader = new BufferedReader(isr)) {
            String line;
            while ((line = reader.readLine()) != null) {
                System.out.println("File Data: " + line);
            }
        } catch (IOException e) {
            System.err.println("Error reading from the file: " + e.getMessage());
        }
    }
}
```

 **output** Data written to the file successfully.  
File Data: Hello guys, I am student of Munger University  
PS C:\Users\C Kumar\Desktop\CollegePractical\Java> █

```
//12> write a java program to generate random number between 0 to 100
import java.util.Random;
```

```
public class _12_Random {
    public static void main(String[] args) {
        Random random = new Random();

        // Generate a random integer between 0 and 100
        int randomNumber = random.nextInt(100);
        System.out.println("Random Number: " + randomNumber);
    }
}
```

 **output** ndom.java } ; if (\$?) { java \_12\_Random }  
Random Number: 74

```
//13> Write a java program to implement Dice Rolling Game
import java.util.Random;
import java.util.Scanner;

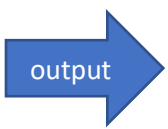
public class _13_DiceGame {
    public static void main(String[] args) {
        Scanner scanner = new Scanner(System.in);
        Random random = new Random();

        int totalRolls = 0;
        int totalWins = 0;

        System.out.println("Welcome to the Dice Rolling Game!");
        System.out.println("Enter the number of times you want to roll the dice: ");
        int numRolls = scanner.nextInt();

        for (int i = 0; i < numRolls; i++) {
            // Generate a random number between 1 and 6
            int diceRoll = random.nextInt(6) + 1;

            System.out.println("Roll " + (i + 1) + ": You rolled a " + diceRoll);
            // Check if the roll is a win (e.g., rolling a 6)
            if (diceRoll == 6) {
                System.out.println("Congratulations! You win!");
                totalWins++;
            }
            totalRolls++;
        }
        double winPercentage = (double) totalWins / totalRolls * 100;
        System.out.println("Total rolls: " + totalRolls);
        System.out.println("Total wins: " + totalWins);
        System.out.println("Win percentage: " + winPercentage + "%");
        scanner.close();
    }
}
```

 **output**

```
Welcome to the Dice Rolling Game!
Enter the number of times you want to roll the dice:
5
Roll 1: You rolled a 3
Roll 2: You rolled a 5
Roll 3: You rolled a 6
Congratulations! You win!
Roll 4: You rolled a 6
Congratulations! You win!
Roll 5: You rolled a 5
Total rolls: 5
Total wins: 2
Win percentage: 40.0%
PS C:\Users\C Kumar\Desktop\CollegePractical\Java> █
```

```
//14> write a java program to print current time, year & month using Date & calendar.
import java.util.Date;
import java.util.Calendar;

public class _14_DateAndTime {
    public static void main(String[] args) {
        Date currentDate = new Date();

        System.out.println("Current Date and Time (using java.util.Date): " +
currentDate);

        Calendar calendar = Calendar.getInstance();

        int year = calendar.get(Calendar.YEAR);
        int month = calendar.get(Calendar.MONTH); // Note: Months start from 0
(January)
        int dayOfMonth = calendar.get(Calendar.DAY_OF_MONTH);
        int hourOfDay = calendar.get(Calendar.HOUR_OF_DAY);
        int minute = calendar.get(Calendar.MINUTE);
        int second = calendar.get(Calendar.SECOND);

        System.out.println("\nUsing Calendar:");
        System.out.println("Year: " + year);
        System.out.println("Month: " + (month + 1)); // Adding 1 to get the correct
month
        System.out.println("Day of Month: " + dayOfMonth);
        System.out.println("Hour of Day: " + hourOfDay);
        System.out.println("Minute: " + minute);
        System.out.println("Second: " + second);

        // Adding 10 days to the current date using Calendar
        calendar.add(Calendar.DAY_OF_MONTH, 10);
        Date futureDate = calendar.getTime();

        System.out.println("\nFuture Date (after adding 10 days): "+futureDate+"\n");
    }
}
```

output

Current Date and Time (using java.util.Date): Sun Sep 10 07:55:26 IST 2023

Using Calendar:

Year: 2023

Month: 9

Day of Month: 10

Hour of Day: 7

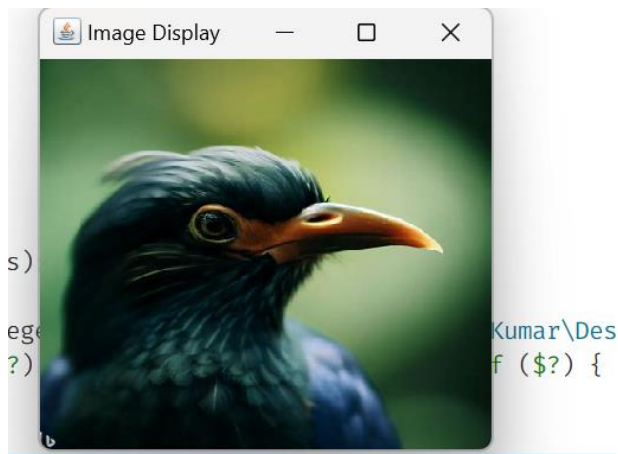
Minute: 55

Second: 27

Future Date (after adding 10 days): Wed Sep 20 07:55:27 IST 2023

```
//15> Create an AWT application that loads and displays an image on the screen.
import java.awt.*;
import java.awt.image.BufferedImage;
import javax.swing.*;
import java.io.File;
import java.io.IOException;
import javax.imageio.ImageIO;
public class _15_AwtImage {
    public static void main(String[] args) {
        SwingUtilities.invokeLater(() -> {
            JFrame frame = new JFrame("Image Display");
            frame.setDefaultCloseOperation(JFrame.EXIT_ON_CLOSE);
            frame.setSize(400, 400);
            // Load the image
            BufferedImage image = loadImage("C:\\Users\\C
Kumar\\Pictures\\Birds.jpg"); // Replace with your image
file
            // path
            // Create a custom JPanel to display the image
            JPanel imagePanel = new JPanel() {
                @Override
                protected void paintComponent(Graphics g) {
                    super.paintComponent(g);
                    if (image != null) {
                        g.drawImage(image, 0, 0, getWidth(), getHeight(), this);
                    }
                }
            };
            frame.add(imagePanel);
            frame.setVisible(true);
        });
    }
    // Load an image from a file
    private static BufferedImage loadImage(String filePath) {
        try {
            return ImageIO.read(new File(filePath));
        } catch (IOException e) {
            e.printStackTrace();
            return null;
        }
    }
}
```

output



//16> in this , i will show uses of this operator and Math.PI, where Math is predefine obj and PI is a data attributes of that Math class.

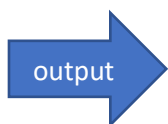
```
class Circle {
    int radius;

    public Circle(int radius) {
        this.radius = radius;
    }

    public void DisplayArea() {
        System.out.println("The area is " + (3.14 * radius * radius));
    }

    public void parimeter() {
        System.out.println("The paremeter is " + (2 * Math.PI * radius));
    }
}

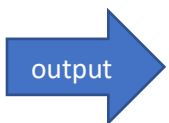
public class _16_ {
    public static void main(String[] args) {
        Circle c = new Circle(8);
        c.DisplayArea();
        c.parimeter();
    }
}
```



```
Java> ; 11 (\?) \ javac _10_.java ; 11 (\?) \ java _1
The area is 200.96
The paremeter is 50.26548245743669
PS C:\Users\C Kumar\Desktop\CollegePractical\Java> █
```

//17> in this, use Scanner class with input types like int, char, string with in also introduce & fix buffere problem

```
import java.util.Scanner;
public class _17_ {
    public static void main(String[] args) {
        Scanner input = new Scanner(System.in);
        /* =====nextInt===== */
        System.out.print("Enter a number: ");
        int num = input.nextInt();
        System.out.println("Enter number is " + num);
        /* =====nextCharAt===== */
        System.out.print("Enter a character: ");
        char ch = input.next().charAt(0);
        System.out.println("Entered character is " + ch);
        /* =====nextLine===== */
        input.nextLine();
        System.out.print("What is your name: ");
        String str = input.nextLine();
        System.out.println("You entered name is : " + str);
        /* =====next===== */
        System.out.print("Enter hobbies: ");
        String hobbies = input.next();
        System.out.println("Entered hobbies is " + hobbies);
        /* =====ReadMultipleLines===== */
        input.nextLine();
        System.out.println("Enter multiple lines string: ");
        String multLines = "";
        while (true) {
            String temp = input.nextLine();
            if (temp == "") {
                break;
            }
            multLines += temp;
        }
        System.out.println("Enter strings are " + multLines);
        input.close();
    }
}
```



```
Enter a number: 895
Enter number is 895
Enter a character: c
Entered character is c
What is your name: chhotu
You entered name is :chhotu
Enter hobbies: programming
Entered hobbies is programming
Enter multiple lines string:
hii,
i am chhotu.
i am very passionate for programming & Development
thank u.
```

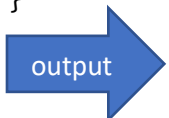
Enter strings are hii,i am chhotu.i am very passionate for programming & Developmentthank u.

```

/*18>Write a program that reverses a given string without using any library functions
or additional data structures.Utilize the String class from the java.lang package. */
public class _18_ReverseString {
    public static void main(String[] args) {
        String inputString = "Ramadhin College Sheikhpura";
        char[] charArray = inputString.toCharArray();
        int left = 0,int right = charArray.length - 1;
        while (left < right) {
            char temp = charArray[left];
            charArray[left] = charArray[right];
            charArray[right] = temp;
            // Move the indices towards the center
            left++;
            right--;
        }
        // Convert the character array back to a string
        String reversedString = new String(charArray);

        System.out.println("Original String: " + inputString);
        System.out.println("Reversed String: " + reversedString);
    }
}

```



```

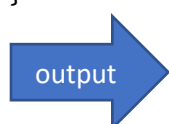
Original String: Ramadhin College Sheikhpura
Reversed String: aruphkiehS egelloC nihdamaR

```

```

/*19> Create a custom exception class that extends java.lang.Exception. Then, write a
program that demonstrate how to throw and catch instances of this custom exception */
class CustomException extends Exception {
    // Constructor with a custom error message
    public CustomException(String message) {super(message);}
}
public class _19_CustomException {
    public static void main(String[] args) {
        try {
            int age = -5; // Assuming age cannot be negative
            if (age < 0) {
                throw new CustomException("Age cannot be negative.");
            }
            System.out.println("Age is valid: " + age);
        } catch (CustomException e) {
            System.out.println("Custom Exception Caught: " + e.getMessage());
        } catch (Exception e) {
            System.out.println("An unexpected error occurred: " + e.getMessage());
        }
    }
}

```



```

Custom Exception Caught: Age cannot be negative.

```

```

/*20> Define an enumeration in Java using the enum keyword. Write a program that
utilizes this enumeration to represent days of the week and perform operations like
finding the next day or checking if a given day is a weekday or weekend. */
// Define the DaysOfWeek enumeration
enum DaysOfWeek {

    SUNDAY, MONDAY, TUESDAY, WEDNESDAY, THURSDAY, FRIDAY, SATURDAY;

    public DaysOfWeek getNextDay() {
        DaysOfWeek[] days = DaysOfWeek.values();
        int nextIndex = (this.ordinal() + 1) % days.length;
        return days[nextIndex];
    }

    public boolean isWeekday() {
        return this != SATURDAY && this != SUNDAY;
    }

    public boolean isWeekend() {
        return this == SATURDAY || this == SUNDAY;
    }
}

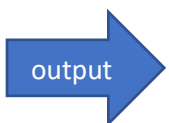
public class _20_Enum {

    public static void main(String[] args) {
        // Use the DaysOfWeek enumeration
        DaysOfWeek today = DaysOfWeek.WEDNESDAY;

        // Find the next day
        DaysOfWeek nextDay = today.getNextDay();
        System.out.println("Next day after " + today + " is " + nextDay);

        // Check if a given day is a weekday or weekend day
        System.out.println(today + " is a weekday: " + today.isWeekday());
        System.out.println(today + " is a weekend day: " + today.isWeekend());
    }
}

```



```

Next day after WEDNESDAY is THURSDAY
WEDNESDAY is a weekday: true
WEDNESDAY is a weekend day: false

```

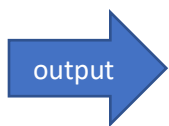


/\*21> Write a Java program that creates and starts multiple threads. You can use the Thread class from the java.lang package to achieve this. Ensure that the threads perform some concurrent tasks. \*/

```
public class _21_MultipleThreads {
    public static void main(String[] args) {
        // Create and start multiple threads
        for (int i = 1; i <= 3; i++) {
            Thread thread = new NumberPrintingThread("Thread " + i);
            thread.start();
        }
    }
}

class NumberPrintingThread extends Thread {
    public NumberPrintingThread(String name) {
        super(name);
    }

    public void run() {
        for (int i = 1; i <= 3; i++) {
            System.out.println(getName() + ": " + i);
            // Introduce a slight delay to simulate concurrent execution
            try {
                Thread.sleep(100);
            } catch (InterruptedException e) {
                e.printStackTrace();
            }
        }
    }
}
```



```
Thread 3: 1
Thread 1: 1
Thread 2: 1
Thread 2: 2
Thread 3: 2
Thread 1: 2
Thread 3: 3
Thread 1: 3
Thread 2: 3
```

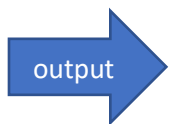
```
/*22. Develop a program that performs various string manipulation tasks using the
StringBuilder class from the java.lang package. Tasks may include concatenation,
insertion, deletion, and reversing.*/
public class _22_StringManipulation {
    public static void main(String[] args) {
        StringBuilder stringBuilder = new StringBuilder("Hello, ");

        stringBuilder.append("World!");
        System.out.println("Concatenation Result: " + stringBuilder.toString());

        stringBuilder.insert(6, "Java ");
        System.out.println("Insertion Result: " + stringBuilder.toString());

        stringBuilder.delete(0, 6);
        System.out.println("Deletion Result: " + stringBuilder.toString());

        stringBuilder.reverse();
        System.out.println("Reversed Result: " + stringBuilder.toString());
    }
}
```



```
Concatenation Result: Hello, World!
Insertion Result: Hello,Java  World!
Deletion Result: Java  World!
Reversed Result: !dlroW  avaJ
```

```

/*23. Create a Java program that demonstrates autoboxing and unboxing of primitive
data types using classes from the java.lang package, such as Integer, Double, and
Boolean. */
public class _23_AutoBoxingAndUnBoxing {
    public static void main(String[] args) {
        /* =====AutoBoxing===== */
        int intValue = 42;
        Integer integerObject = intValue; // Autoboxing int to Integer class
        System.out.println("Autoboxing: int to Integer: " + integerObject);

        double doubleValue = 3.14;
        Double doubleObject = doubleValue; // Autoboxing double to Double class
        System.out.println("Autoboxing: double to Double: " + doubleObject);

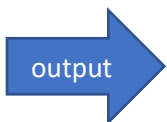
        boolean booleanValue = true;
        Boolean booleanObject = booleanValue; // Autoboxing boolean to Boolean class
        System.out.println("Autoboxing: boolean to Boolean: " + booleanObject);

        /* =====Unboxing===== */
        Integer intWrapper = 99;
        int unboxedInt = intWrapper; // Unboxing Integer to int type
        System.out.println("Unboxing: Integer to int: " + unboxedInt);

        Double doubleWrapper = 2.718;
        double unboxedDouble = doubleWrapper; // Unboxing Double to double type
        System.out.println("Unboxing: Double to double: " + unboxedDouble);

        Boolean booleanWrapper = Boolean.FALSE;
        boolean unboxedBoolean = booleanWrapper; // Unboxing Boolean to boolean type
        System.out.println("Unboxing: Boolean to boolean: " + unboxedBoolean);
    }
}

```



```

Autoboxing: int to Integer: 42
Autoboxing: double to Double: 3.14
Autoboxing: boolean to Boolean: true
Unboxing: Integer to int: 99
Unboxing: Double to double: 2.718
Unboxing: Boolean to boolean: false

```

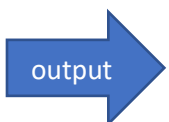
/\*24> demonstrating hierarchical inheritance, we have a base class Vehicle and two derived classes, Car and Bike. Each class has its own methods, and the program creates objects of both Car and Bike \*/

```
class Vehicle {
    String type;

    Vehicle(String type) {
        this.type = type;
    }
    void start() {
        System.out.println(type + " is starting.");
    }
    void stop() {
        System.out.println(type + " is stopping.");
    }
}
class Car extends Vehicle {
    Car() {
        super("Car");
    }
    void drive() {
        System.out.println("Car is driving.");
    }
}
class Bike extends Vehicle {
    Bike() {
        super("Bike");
    }
    void ride() {
        System.out.println("Bike is riding.");
    }
}
public class _24_Hirarchical {
    public static void main(String[] args) {
        Car myCar = new Car();
        Bike myBike = new Bike();

        myCar.start();
        myCar.drive();
        myCar.stop();

        myBike.start();
        myBike.ride();
        myBike.stop();
    }
}
```



```
Car is starting.
Car is driving.
Car is stopping.
Bike is starting.
Bike is riding.
Bike is stopping.
```

//25> Develop a program using AWT components to create a simple calculator interface.

```
import java.awt.*;
import java.awt.event.ActionEvent;
import java.awt.event.ActionListener;

public class _25_Calculator extends Frame implements ActionListener {
    private TextField displayField;
    private String operator;
    private double operand1, operand2, result;

    public _25_Calculator() {
        operator = "";
        operand1 = operand2 = result = 0.0;

        setTitle("Simple Calculator");
        setSize(300, 400);
        setLayout(new BorderLayout());

        displayField = new TextField();
        displayField.setEditable(false);
        add(displayField, BorderLayout.NORTH);

        Panel buttonPanel = new Panel();
        buttonPanel.setLayout(new GridLayout(4, 4));

        String[] buttonLabels = {
            "7", "8", "9", "/",
            "4", "5", "6", "*",
            "1", "2", "3", "-",
            "0", ".", "=", "+"
        };

        for (String label : buttonLabels) {
            Button button = new Button(label);
            button.addActionListener(this);
            buttonPanel.add(button);
        }

        add(buttonPanel, BorderLayout.CENTER);
        addWindowListener(new java.awt.event.WindowAdapter() {
            public void windowClosing(java.awt.event.WindowEvent windowEvent) {
                System.exit(0);
            }
        });
        setVisible(true);
    }

    public void actionPerformed(ActionEvent e) {
        String command = e.getActionCommand();

        if (Character.isDigit(command.charAt(0)) || command.equals(".")) {
            displayField.setText(displayField.getText() + command);
        } else if (command.equals("C")) {
            displayField.setText("");
        }
    }
}
```

```

        operator = "";
        operand1 = operand2 = result = 0.0;
    } else if (command.equals("=")) {
        operand2 = Double.parseDouble(displayField.getText());
        calculateResult();
        displayField.setText(Double.toString(result));
        operand1 = result;
        operator = "";
    } else {
        operator = command;
        operand1 = Double.parseDouble(displayField.getText());
        displayField.setText("");
    }
}

private void calculateResult() {
    switch (operator) {
        case "+":
            result = operand1 + operand2;
            break;
        case "-":
            result = operand1 - operand2;
            break;
        case "*":
            result = operand1 * operand2;
            break;
        case "/":
            if (operand2 != 0) {
                result = operand1 / operand2;
            } else {
                displayField.setText("Error: Division by zero");
                operator = "";
                operand1 = operand2 = result = 0.0;
            }
            break;
    }
}

public static void main(String[] args) {new _25_Calculator();}
}

```

