JAVA Practical Programs

Slip 1

1.Write a Java program to display all the alphabets between 'A' to 'Z' after every 2 seconds.

```
public class Slip1_1 extends Thread{
    char c;
    public void run(){
        for(c = 'A'; c<='Z';c++){
            System.out.println(""+c);
            try{
                Thread.sleep(3000);
            }
            catch(Exception e){
                e.printStackTrace();
            }
        }
    }
    public static void main(String args[]){
        Slip1_1 t = new Slip1_1();
        t.start();
    }
}
```

2.Write a Java program to accept the details of Employee (Eno, EName, Designation, Salary) from a user and store it into the database. (Use Swing)

```
import java.sql.*;

public class EmployeeDetails {

    static final String JDBC_URL = "jdbc:mysql://localhost:3306/practical";
    static final String USERNAME = "root";
    static final String PASSWORD = "root";

    public static void main(String[] args) {
        try (Connection conn = DriverManager.getConnection(JDBC_URL, USERNAME,
PASSWORD)) {

            String eno = promptUser("Enter Employee Number: ");
            String ename = promptUser("Enter Employee Name: ");
            String designation = promptUser("Enter Employee Designation: ");
            double salary = Double.parseDouble(promptUser("Enter Employee
Salary: "));

            insertEmployee(conn, eno, ename, designation, salary);
            System.out.println("Employee details inserted successfully.");
```

```
        } catch (SQLException e) {
            e.printStackTrace();
        }
    }

    private static String promptUser(String message) {
        java.util.Scanner scanner = new java.util.Scanner(System.in);
        System.out.print(message);
        return scanner.nextLine();
    }

    private static void insertEmployee(Connection conn, String eno, String
ename, String designation, double salary) throws SQLException {
        String sql = "INSERT INTO Employee (Eno, EName, Designation, Salary)
VALUES (?, ?, ?, ?)";
        try (PreparedStatement pstmt = conn.prepareStatement(sql)) {
            pstmt.setString(1, eno);
            pstmt.setString(2, ename);
            pstmt.setString(3, designation);
            pstmt.setDouble(4, salary);
            pstmt.executeUpdate();
        }
    }
}
```

SQL->

```
CREATE TABLE Employee (
    Eno INT PRIMARY KEY,
    EName VARCHAR(255),
    Designation VARCHAR(255),
    Salary DECIMAL(10, 2)
);
```

Slip 4

1.Write a Java program using Runnable interface to blink Text on the frame

```
import javax.swing.*;
import java.awt.*;

public class BlinkingText implements Runnable {
    private JLabel label;

    public BlinkingText(JLabel label) {
        this.label = label;
    }

    @Override
    public void run() {
        try {
            while (true) {
                SwingUtilities.invokeLater(() -> label.setVisible(true));
                Thread.sleep(500);
                SwingUtilities.invokeLater(() -> label.setVisible(false));
                Thread.sleep(500);
            }
        } catch (InterruptedException e) {
            e.printStackTrace();
        }
    }

    public static void main(String[] args) {
        SwingUtilities.invokeLater(() -> {
            JFrame frame = new JFrame("Blinking Text");
            frame.setSize(300, 200);
            frame.setDefaultCloseOperation(JFrame.EXIT_ON_CLOSE);

            JLabel label = new JLabel("Blinking Text");
            label.setHorizontalAlignment(SwingConstants.CENTER);
            label.setFont(new Font("Arial", Font.BOLD, 20));
            frame.add(label, BorderLayout.CENTER);

            BlinkingText blinkingText = new BlinkingText(label);
            Thread thread = new Thread(blinkingText);
            thread.start();

            frame.setVisible(true);
        });
    }
}
```

2.Write a Java program to store city names and their STD codes using an appropriate collection and perform following operations: i. Add a new city and its code (No duplicates) ii. Remove a city from the collection iii. Search for a city name and display the code

```
import java.util.*;

public class CityStdCodes {
    private Map<String, Integer> cityStdCodes;

    public CityStdCodes() {
        this.cityStdCodes = new HashMap<>();
    }


    public void addCity(String city, int stdCode) {
        if (!cityStdCodes.containsKey(city)) {
            cityStdCodes.put(city, stdCode);
            System.out.println("City '" + city + "' added with STD code: " +
stdCode);
        } else {
            System.out.println("City '" + city + "' already exists with STD
code: " + cityStdCodes.get(city));
        }
    }

    public void removeCity(String city) {
        if (cityStdCodes.containsKey(city)) {
            cityStdCodes.remove(city);
            System.out.println("City '" + city + "' removed from the
collection.");
        } else {
            System.out.println("City '" + city + "' does not exist in the
collection.");
        }
    }

    public void searchCity(String city) {
        if (cityStdCodes.containsKey(city)) {
            System.out.println("STD code for city '" + city + "' is: " +
cityStdCodes.get(city));
        } else {
            System.out.println("City '" + city + "' not found in the
collection.");
        }
    }
```

```
    public static void main(String[] args) {
        CityStdCodes cityStdCodes = new CityStdCodes();

        cityStdCodes.addCity("New York", 212);
        cityStdCodes.addCity("London", 20);
        cityStdCodes.addCity("Paris", 33);
        cityStdCodes.addCity("Tokyo", 81);

        cityStdCodes.searchCity("Paris");
        cityStdCodes.searchCity("Berlin");

        cityStdCodes.removeCity("London");
        cityStdCodes.removeCity("Sydney");
    }
}
```

Slip 6

1. Write a Java program to accept 'n' integers from the user and store them in a collection. Display them in the sorted order. The collection should not accept duplicate elements. (Use a suitable collection). Search for a particular element using predefined search method in the Collection framework.

```
import java.util.*;

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

        Set<Integer> numbers = new TreeSet<>();

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

        System.out.println("Enter the integers:");

        for (int i = 0; i < n; i++) {
            int num = scanner.nextInt();
            numbers.add(num);
        }

        System.out.println("Integers in sorted order:");
        for (int num : numbers) {
            System.out.print(num + " ");
        }
        System.out.println();

        System.out.print("Enter the number to search: ");
        int searchNum = scanner.nextInt();
        if (numbers.contains(searchNum)) {
            System.out.println(searchNum + " is found in the collection.");
        } else {
            System.out.println(searchNum + " is not found in the
collection.");
        }

        scanner.close();
    }
}
```

2. Write a java program to simulate traffic signal using threads.

```
public class TrafficSignalSimulation {

    enum SignalState {
        RED, YELLOW, GREEN
    }

    static class TrafficSignal {
        private SignalState state;

        public TrafficSignal() {
            this.state = SignalState.RED;
        }

        public synchronized void changeState() {
            switch (state) {
                case RED:
                    state = SignalState.GREEN;
                    break;
                case YELLOW:
                    state = SignalState.RED;
                    break;
                case GREEN:
                    state = SignalState.YELLOW;
                    break;
            }
            System.out.println("Signal changed to " + state);
            notifyAll();
        }


        public synchronized SignalState getState() {
            return state;
        }
    }

    static class Car implements Runnable {
        private String name;
        private TrafficSignal signal;

        public Car(String name, TrafficSignal signal) {
            this.name = name;
            this.signal = signal;
        }

        @Override
```

```
        public void run() {
            while (true) {
                synchronized (signal) {
                    try {
                        while (signal.getState() != SignalState.GREEN) {
                            System.out.println(name + " is waiting at the
signal.");
                            signal.wait();
                        }
                        System.out.println(name + " is crossing the signal.");
                        Thread.sleep(2000);
                    } catch (InterruptedException e) {
                        e.printStackTrace();
                    }
                }
            }
        }
    }

    public static void main(String[] args) {
        TrafficSignal signal = new TrafficSignal();

        Thread car1 = new Thread(new Car("Car 1", signal));
        Thread car2 = new Thread(new Car("Car 2", signal));
        Thread car3 = new Thread(new Car("Car 3", signal));

        car1.start();
        car2.start();
        car3.start();

        while (true) {
            try {
                Thread.sleep(5000);
                signal.changeState();
            } catch (InterruptedException e) {
                e.printStackTrace();
            }
        }
    }
}
```

Slip 7

1. Write a java program that implements a multi-thread application that has three threads. First thread generates random integer number after every one second, if the number is even; second thread computes the square of that number and print it. If the number is odd, the third thread computes the of cube of that number and print it.

```
import java.util.Random;
class Square extends Thread
{
 int x;
 Square(int n)
 {
 x = n;
 }
 public void run()
 {
 int sqr = x * x;
 System.out.println("Square of " + x + " = " + sqr );
 }
}
class Cube extends Thread
{
 int x;
 Cube(int n)
 {x = n;
 }
 public void run()
 {
 int cub = x * x * x;
 System.out.println("Cube of " + x + " = " + cub );
 }
}
class Number extends Thread
{
 public void run()
 {
 Random random = new Random();
 for(int i =0; i<10; i++)
 {
 int randomInteger = random.nextInt(100);
 System.out.println("Random Integer generated : " + randomInteger);
 Square s = new Square(randomInteger);
 s.start();
 Cube c = new Cube(randomInteger);
 c.start();
 try {
 Thread.sleep(1000);} catch (InterruptedException ex) {
 System.out.println(ex);
```

```
}
 }
 }
}
public class Slip7_1 {
 public static void main(String args[])
 {
 Number n = new Number();
 n.start();
 }
}
```

2.Write a java program for the following: i. To create a Product(Pid, Pname, Price) table. ii. Insert at least five records into the table. iii. Display all the records from a table.

```
import java.sql.*;

public class ProductManagement {

    static final String JDBC_URL = "jdbc:mysql://localhost:3306/practical";
    static final String USERNAME = "root";
    static final String PASSWORD = "root";

    public static void main(String[] args) {
        try {
            Connection connection = DriverManager.getConnection(JDBC_URL,
USERNAME, PASSWORD);


            createProductTable(connection);

            insertRecords(connection);


            displayRecords(connection);


            connection.close();
        } catch (SQLException e) {
            e.printStackTrace();
        }
    }
    private static void createProductTable(Connection connection) throws
SQLException

{
```

```
        String createTableSQL = "CREATE TABLE IF NOT EXISTS Product (" +
                                "Pid INT ," +
                                "Pname VARCHAR(255)," +
                                "Price DECIMAL(10, 2)" +
                                ")";
        Statement statement = connection.createStatement();
        statement.execute(createTableSQL);
        statement.close();
    }


    private static void insertRecords(Connection connection) throws
SQLException {
        String insertSQL = "INSERT INTO Product (Pid, Pname, Price) VALUES (?,
?, ?)";
        PreparedStatement preparedStatement =
connection.prepareStatement(insertSQL);

        Object[][] data = {
            {1, "Product A", 10.99},
            {2, "Product B", 20.49},
            {3, "Product C", 15.79},
            {4, "Product D", 30.25},
            {5, "Product E", 25.99}
        };

        for (Object[] row : data) {
            preparedStatement.setInt(1, (int) row[0]);
            preparedStatement.setString(2, (String) row[1]);
            preparedStatement.setDouble(3, (double) row[2]);
            preparedStatement.executeUpdate();
        }

        preparedStatement.close();
    }

    private static void displayRecords(Connection connection) throws
SQLException {
        String selectSQL = "SELECT * FROM Product";
        Statement statement = connection.createStatement();
        ResultSet resultSet = statement.executeQuery(selectSQL);

        ResultSetMetaData metaData = resultSet.getMetaData();
        int columnCount = metaData.getColumnCount();
        for (int i = 1; i <= columnCount; i++) {
            System.out.print(metaData.getColumnName(i) + "\t");
        }
        System.out.println();
```

```
        while (resultSet.next()) {
            for (int i = 1; i <= columnCount; i++) {
                System.out.print(resultSet.getString(i) + "\t");
            }
            System.out.println();
        }

        resultSet.close();
        statement.close();
    }
}
```

SQL->

```
CREATE TABLE IF NOT EXISTS Product (
    Pid INT AUTO_INCREMENT PRIMARY KEY,
    Pname VARCHAR(255),
    Price DECIMAL(10, 2)
);
```

```
2)
INSERT INTO Product (Pid, Pname, Price) VALUES
(1, 'Laptop', 999.99),
(2, 'Mobile Phone', 599.99),
(3, 'Headphones', 99.99),
(4, 'Tablet', 399.99),
(5, 'Smartwatch', 199.99);
```

Slip 13

1. Write a Java program to display information about the database and list all the tables in the database. (Use DatabaseMetaData).

```
import java.sql.*;

public class DatabaseInfo {

    static final String JDBC_URL = "jdbc:mysql://localhost:3306/practical";
    static final String USERNAME = "root";
    static final String PASSWORD = "root";

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

            Connection connection = DriverManager.getConnection(JDBC_URL,
USERNAME, PASSWORD);

            DatabaseMetaData metaData = connection.getMetaData();

            System.out.println("Database Information:");
            System.out.println("Database Name: " +
metaData.getDatabaseProductName());
            System.out.println("Database Version: " +
metaData.getDatabaseProductVersion());
            System.out.println();

            ResultSet tablesResultSet = metaData.getTables(null, null, null,
new String[]{"TABLE"});
            System.out.println("Tables in the Database:");
            while (tablesResultSet.next()) {
                String tableName = tablesResultSet.getString("TABLE_NAME");
                System.out.println(tableName);
            }

            tablesResultSet.close();
            connection.close();
        } catch (SQLException e) {
            e.printStackTrace();
        }
    }
}
```

2. Write a Java program to show lifecycle (creation, sleep, and dead) of a thread. Program should print randomly the name of thread and value of sleep time. The name of the thread should be hard coded through constructor. The sleep time of a thread will be a random integer in the range 0 to 4999

```
import java.util.Random;

public class ThreadLifecycleDemo extends Thread {
    private String threadName;

    public ThreadLifecycleDemo(String threadName) {
        this.threadName = threadName;
    }

    @Override
    public void run() {
        System.out.println(threadName + " is created.");

        Random random = new Random();
        int sleepTime = random.nextInt(5000);

        System.out.println(threadName + " will sleep for " + sleepTime + "
milliseconds.");

        try {
            Thread.sleep(sleepTime);
        } catch (InterruptedException e) {
            System.out.println(threadName + " was interrupted while
sleeping.");
        }

        System.out.println(threadName + " is dead.");
    }

    public static void main(String[] args) {

        ThreadLifecycleDemo thread1 = new ThreadLifecycleDemo("Thread 1");
        ThreadLifecycleDemo thread2 = new ThreadLifecycleDemo("Thread 2");
        ThreadLifecycleDemo thread3 = new ThreadLifecycleDemo("Thread 3");

        thread1.start();
        thread2.start();
        thread3.start();
    }
}
```

Slip 16

1. **Write a java program to create a TreeSet, add some colors (String) and print out the content of TreeSet in ascending order**.

```
 import java.util.TreeSet;

public class Slip16_1 {
  public static void main(String[] args) {
  TreeSet<String> tree_set = new TreeSet<String>();
  tree_set.add("Red");
  tree_set.add("Green");
  tree_set.add("Orange");
  tree_set.add("White");
  tree_set.add("Black");
  System.out.println("Tree set: ");
  System.out.println(tree_set);
 }
}
```

2. **Write a Java program to accept the details of Teacher (TNo, TName, Subject). Insert at least 5 Records into Teacher Table and display the details of Teacher who is teaching "JAVA" Subject. (Use PreparedStatement Interface).**

```
import java.sql.*;

public class TeacherManagement {

    static final String JDBC_URL = "jdbc:mysql://localhost:3306/practical";
    static final String USERNAME = "root";
    static final String PASSWORD = "root";

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

            Connection connection = DriverManager.getConnection(JDBC_URL,
USERNAME, PASSWORD);

            insertRecords(connection);

            displayJavaTeachers(connection);

            connection.close();
        } catch (SQLException e) {
            e.printStackTrace();
        }
    }
```

```
    private static void insertRecords(Connection connection) throws
SQLException {
        String insertSQL = "INSERT INTO Teacher (TNo, TName, Subject) VALUES
(?, ?, ?)";
        PreparedStatement preparedStatement =
connection.prepareStatement(insertSQL);

        Object[][] data = {
            {1, "John Doe", "JAVA"},
            {2, "Jane Smith", "C++"},
            {3, "Alice Johnson", "JAVA"},
            {4, "Bob Brown", "Python"},
            {5, "Emily Davis", "JAVA"}
        };

        for (Object[] row : data) {
            preparedStatement.setInt(1, (int) row[0]);
            preparedStatement.setString(2, (String) row[1]);
            preparedStatement.setString(3, (String) row[2]);
            preparedStatement.executeUpdate();
        }

        preparedStatement.close();
    }

    private static void displayJavaTeachers(Connection connection) throws
SQLException {
        String selectSQL = "SELECT * FROM Teacher WHERE Subject = ?";
        PreparedStatement preparedStatement =
connection.prepareStatement(selectSQL);
        preparedStatement.setString(1, "JAVA");
        ResultSet resultSet = preparedStatement.executeQuery();

        System.out.println("Teachers teaching JAVA:");
        System.out.println("TNo\tTName\tSubject");
        while (resultSet.next()) {
            int tNo = resultSet.getInt("TNo");
            String tName = resultSet.getString("TName");
            String subject = resultSet.getString("Subject");
            System.out.println(tNo + "\t" + tName + "\t" + subject);
        }

        resultSet.close();
        preparedStatement.close();
    }
}
```

SQL->

```sql
CREATE TABLE IF NOT EXISTS Teacher (
    TNo INT PRIMARY KEY,
    TName VARCHAR(255),
    Subject VARCHAR(50)
);
```

Slip 17

1. Write a java program to accept 'N' integers from a user. Store and display integers in sorted order having proper collection class. The collection should not accept duplicate elements.

```
import java.util.*;
import java.io.*;
class Slip17_1{
public static void main(String[] args) throws Exception{
int no,element,i;
BufferedReader br=new BufferedReader(new InputStreamReader(System.in));
TreeSet ts=new TreeSet();
System.out.println("Enter the of elements :");
no=Integer.parseInt(br.readLine());
for(i=0;i<no;i++){
System.out.println("Enter the element : ");
element=Integer.parseInt(br.readLine());
ts.add(element);
}
System.out.println("The elements in sorted order :"+ts);
System.out.println("Enter element to be serach : ");
element = Integer.parseInt(br.readLine());
if(ts.contains(element))
System.out.println("Element is found");
else
System.out.println("Element is NOT found");
}
}
```

2. Write a Multithreading program in java to display the number's between 1 to 100 continuously in a TextField by clicking on button. (Use Runnable Interface).

```
import javax.swing.*;
import java.awt.*;
import java.awt.event.ActionListener;
import java.awt.event.ActionEvent;
public class MultiThread extends JFrame implements ActionListener
{
    Container cc;
    JButton b1,b2;
    JTextField t1;
    MultiThread()
    {
        setVisible(true);
        setSize(1024,768);
        cc=getContentPane();
        setLayout(null);
```

```
        t1=new JTextField(500);
        cc.add(t1);
        t1.setBounds(10,10,1000,30);
        b1=new JButton("start");
        cc.add(b1);
        b1.setBounds(20,50,100,40);
        b1.addActionListener(this);
        setDefaultCloseOperation(EXIT_ON_CLOSE);
    }
    public void actionPerformed(ActionEvent e)
    {
        if(e.getSource()==b1)
        {
            new Mythread();
        }
    }
    class Mythread extends Thread
    {
        Mythread()
        {
        start();
        }
        public void run()
        {
            for(int i=1;i<=100;i++)
            {
                try                 {
                    Thread.sleep(1000);
                }
                catch (InterruptedException e) {
                }
                t1.setText(t1.getText()+""+i+"\n");
          }
        }
    }
    public static void main(String arg[])
    {
        new MultiThread().show();
    }
}
```

Slip 21

1. **Write a java program to accept 'N' Subject Names from a user store them into LinkedList Collection and Display them by using Iterator interface.**

```
import java.util.*;
import java.io.*;
public class Slip21_1
{
    public static void main(String args[])throws Exception
    {
        int n;
        BufferedReader br = new BufferedReader(new
InputStreamReader(System.in));
                LinkedList li = new LinkedList ();
                System.out.println("\nEnter number of  Employee : ");

        n = Integer.parseInt(br.readLine());
                System.out.println("\nEnter name : ");
        for(int i = 1; i <= n; i++)
        {
            li.add(br.readLine());
        }
                System.out.println("\nLink List Content : ");
        Iterator it = li.iterator();
        {
            System.out.println(it.next());
        }
                System.out.println("\nReverse order : ");
        ListIterator lt = li.listIterator();
        while(lt.hasNext())
        {
            lt.next();
        }
        while(lt.hasPrevious())
        {
            System.out.println(lt.previous());
        }
     }
}
```

2. Write a java program to solve producer consumer problem in which a producer produces a value and consumer consume the value before producer generate the next value. (Hint: use thread synchronization)

```
import java.util.LinkedList;

public class Threadexample {
    public static void main(String[] args)
        throws InterruptedException
    {

        final PC pc = new PC();


        Thread t1 = new Thread(new Runnable() {
            @Override
            public void run()
            {
                try {
                    pc.produce();
                }
                catch (InterruptedException e) {
                    e.printStackTrace();
                }
            }
        });


        Thread t2 = new Thread(new Runnable() {
            @Override
            public void run()
            {
                try {
                    pc.consume();
                }
                catch (InterruptedException e) {
                    e.printStackTrace();
                }
            }
        });


        t1.start();
        t2.start();

        t1.join();
        t2.join();
    }
```

```
    public static class PC {

        LinkedList<Integer> list = new LinkedList<>();
        int capacity = 2;


        public void produce() throws InterruptedException
        {
            int value = 0;
            while (true) {
                synchronized (this)

                    while (list.size() == capacity)
                        wait();

                    System.out.println("Producer produced-"
                                       + value);

                    list.add(value++);

                    notify();

                    Thread.sleep(1000);
                }
            }
        }

        public void consume() throws InterruptedException
        {
            while (true) {
                synchronized (this)

                    while (list.size() == 0)
                        wait();

                    int val = list.removeFirst();

                    System.out.println("Consumer consumed-"
                                       + val);

                    notify();

                    Thread.sleep(1000);
                }
            }
        }
    }
}
```

Slip 23

1.Write a java program to accept a String from a user and display each vowel from a String after every 3 seconds.

```
import java.io.*;
public class Slip23_1 extends Thread{

    String s1;
    StringVowels(String s){
         s1=s;
        start();
    }
    public void run(){
        System.out.println("Vowels are :- ");
        for(int i=0;i<s1.length();i++ ){
            char ch=s1.charAt(i);
            if(ch=='a'|| ch=='e'|| ch=='i'|| ch=='o'|| ch=='u'|| ch=='A'||
ch=='E'|| ch=='I'|| ch=='O'|| ch=='U')
            System.out.print(" "+ch);
        }
    }
    public static void main(String args[]) throws Exception{

        BufferedReader br = new BufferedReader(new
InputStreamReader(System.in));
        System.out.println("Enter a string");
        String str=br.readLine();
        StringVowels v=new StringVowels(str);
    }
}
```

2. **Write a java program to accept 'N' student names through command line, store them into the appropriate Collection and display them by using Iterator and ListIterator interface.**

```
import java.util.ArrayList;
import java.util.Iterator;
import java.util.ListIterator;

public class StudentNames {

    public static void main(String[] args) {
        if (args.length == 0) {
```

```
            System.out.println("Please provide student names as command line
arguments.");
            return;
        }

        ArrayList<String> studentList = new ArrayList<>();


        for (String arg : args) {
            studentList.add(arg);
        }

        System.out.println("Student names entered:");
        displayWithIterator(studentList);
        System.out.println("\n\nStudent names entered in reverse order:");
        displayWithListIterator(studentList);
    }

    public static void displayWithIterator(ArrayList<String> studentList) {

        Iterator<String> iterator = studentList.iterator();
        while (iterator.hasNext()) {
            System.out.println(iterator.next());
        }
    }

    public static void displayWithListIterator(ArrayList<String> studentList)
{

        ListIterator<String> listIterator =
studentList.listIterator(studentList.size());
        while (listIterator.hasPrevious()) {
            System.out.println(listIterator.previous());
        }
    }
}
```

Slip 29

1. **Write a Java program to display information about all columns in the DONAR table using ResultSetMetaData.**

```
import java.util.LinkedList;
public class LinkedListOperations {
    public static void main(String[] args) {

        LinkedList<Integer> linkedList = new LinkedList<>();

        linkedList.add(10);
        linkedList.add(20);
        linkedList.add(30);

        linkedList.addFirst(5);

        linkedList.removeLast();

        int size = linkedList.size();
        System.out.println("Size of the LinkedList: " + size);
    }
}
```

2. **Write a Java program to create LinkedList of integer objects and perform the following: i. Add element at first position ii. Delete last element iii. Display the size of link list**

```
import java.sql.*;
public class DonarTableInfo {

    static final String JDBC_URL = "jdbc:mysql://localhost:3306/practical";
    static final String USERNAME = "root";
    static final String PASSWORD = "root";

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

            Connection connection = DriverManager.getConnection(JDBC_URL,
USERNAME, PASSWORD);

            DatabaseMetaData metaData = connection.getMetaData();
            ResultSet resultSet = metaData.getColumns(null, null, "DONAR",
null);
```

```
            System.out.println("Column Information for DONAR table:");
            while (resultSet.next()) {
                String columnName = resultSet.getString("COLUMN_NAME");
                String columnType = resultSet.getString("TYPE_NAME");
                int columnSize = resultSet.getInt("COLUMN_SIZE");

                System.out.println("Column Name: " + columnName);
                System.out.println("Column Type: " + columnType);
                System.out.println("Column Size: " + columnSize);
                System.out.println();
            }


            resultSet.close();
            connection.close();
        } catch (SQLException e) {
            e.printStackTrace();
        }
    }
}
```

SQL->

```
INSERT INTO DONAR (Name, BloodGroup, Age, Gender, ContactNumber, Email,
Address)
VALUES ('John Doe', 'O+', 30, 'Male', '1234567890', 'john.doe@example.com',
'123 Main St, City'),
       ('Jane Smith', 'A-', 25, 'Female', '9876543210',
'jane.smith@example.com', '456 Elm St, Town'),
       ('Alice Johnson', 'B+', 35, 'Female', '7890123456',
'alice.johnson@example.com', '789 Oak St, Village'),
       ('Bob Brown', 'AB-', 40, 'Male', '6543210987', 'bob.brown@example.com',
'321 Maple St, Suburb'),
       ('Emily Davis', 'A+', 28, 'Female', '4567890123',
'emily.davis@example.com', '987 Pine St, Countryside');
```

Slip 30

1. **Write a java program for the implementation of synchronization.**

```
class Counter {
    private int count = 0;
    public synchronized void increment() {
        count++;
    }

    public synchronized void decrement() {
        count--;
    }
    public synchronized int getCount() {
        return count;
    }
}

class CounterThread extends Thread {
    private Counter counter;
    private boolean increment;

    public CounterThread(Counter counter, boolean increment) {
        this.counter = counter;
        this.increment = increment;
    }

    public void run() {
        if (increment) {
            for (int i = 0; i < 1000; i++) {
                counter.increment();
            }
        } else {
            for (int i = 0; i < 1000; i++) {
                counter.decrement();
            }
        }
    }
}

public class SynchronizationExample {
    public static void main(String[] args) throws InterruptedException {
        Counter counter = new Counter();

        CounterThread incrementThread = new CounterThread(counter, true);
        CounterThread decrementThread = new CounterThread(counter, false);
```

```
        incrementThread.start();
        decrementThread.start();
        incrementThread.join();
        decrementThread.join();
        System.out.println("Final Counter Value: " + counter.getCount());
    }
}
```

2. **Write a Java Program for the implementation of scrollable ResultSet. Assume Teacher table with attributes (TID, TName, Salary) is already created.**

```
import java.sql.Connection;
import java.sql.DriverManager;
import java.sql.ResultSet;
import java.sql.SQLException;
import java.sql.Statement;

public class ScrollableResultSetExample {
    public static void main(String[] args) {
        Connection connection = null;
        Statement statement = null;
        ResultSet resultSet = null;

        try {
            connection =
DriverManager.getConnection("jdbc:mysql://localhost/tyjdbc", "root", "root");
            statement =
connection.createStatement(ResultSet.TYPE_SCROLL_SENSITIVE,
ResultSet.CONCUR_READ_ONLY);
            resultSet = statement.executeQuery("SELECT * FROM Teacher");

            resultSet.last();

            int rowCount = resultSet.getRow();

            resultSet.beforeFirst();

            System.out.println("TID\tTName\tSalary");
            while (resultSet.next()) {
                int tid = resultSet.getInt("TID");
                String tname = resultSet.getString("TName");
                double salary = resultSet.getDouble("Salary");
                System.out.println(tid + "\t" + tname + "\t" + salary);
            }

            System.out.println("Total rows: " + rowCount);
        } catch (SQLException e) {
```

```
            e.printStackTrace();
        } finally {
            try {
                if (resultSet != null) resultSet.close();
                if (statement != null) statement.close();
                if (connection != null) connection.close();
            } catch (SQLException e) {
                e.printStackTrace();
            }
        }
    }
}
```

SQL->

```
CREATE TABLE Teacher (
    TID INT PRIMARY KEY,
    TName VARCHAR(50),
    Salary DECIMAL(10, 2)
);

INSERT INTO Teacher (TID, TName, Salary)
VALUES (1, 'John Doe', 50000.00);
```