Slip 1

Q1. Write a Java program to display all the alphabets between 'A' to 'Z' after every 2 seconds.

Solution:

```
class PrintAlphabets extends Thread{
    public void run(){
        char ch = 'A';
        while(ch<='Z'){
            try{
                System.out.print(ch+" ");
                Thread.sleep(1000);
                ch++;
            }catch(InterruptedException ie){
                System.out.println(ie);
            }
        }
    }
}

class TestPrintAlphabets{
    public static void main(String args[]){
        new PrintAlphabets().start();
    }
}
```

Q2. 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)

Solution:

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

class EmployeeFrame extends JFrame implements ActionListener{
    Connection con;
    PreparedStatement pstmt;
    JLabel jl1,jl2,jl3,jl4;
    JTextField jtf1,jtf2,jtf3,jtf4;
    JPanel jp;
    JButton jb;

    EmployeeFrame(){
        try{
            Class.forName("org.postgresql.Driver");

            con =
DriverManager.getConnection("jdbc:postgresql://localhost/nsgacademy","sagrawal
","mjain");

            pstmt = con.prepareStatement("insert into employee
values(?,?,?,?)");

            setSize(700,300);
            setTitle("NSGAcademy");
            setLocationRelativeTo(null);

            jp = new JPanel();
            jp.setLayout(new GridLayout(4,2));

            jl1 = new JLabel("Employee No");
            jp.add(jl1);
            jtf1 = new JTextField(20);
            jp.add(jtf1);

            jl2 = new JLabel("Employee Name");
            jp.add(jl2);
            jtf2 = new JTextField(20);
            jp.add(jtf2);
```

```
            jl3 = new JLabel("Designation");
            jp.add(jl3);
            jtf3 = new JTextField(20);
            jp.add(jtf3);

            jl4 = new JLabel("Salary");
            jp.add(jl4);
            jtf4 = new JTextField(20);
            jp.add(jtf4);
            add(jp,BorderLayout.CENTER);
            jb = new JButton("Add Record");
            add(jb,BorderLayout.SOUTH);
            jb.addActionListener(this);
            setDefaultCloseOperation(JFrame.EXIT_ON_CLOSE);
            setVisible(true);

        }catch(ClassNotFoundException cnfe){
            JOptionPane.showMessageDialog(this,"Driver Not Found : "+cnfe,
"Error", JOptionPane.ERROR_MESSAGE);
            dispose();

        }catch(SQLException sqle){
            JOptionPane.showMessageDialog(this,"SQLException : "+sqle,
"Error", JOptionPane.ERROR_MESSAGE);
            dispose();

        }catch(Exception e){
            JOptionPane.showMessageDialog(this,"Exception : "+e, "Error",
JOptionPane.ERROR_MESSAGE);
            dispose();
        }
    }
```

```
    public void actionPerformed(ActionEvent ae){
        if(jtf1.getText().equals("") || jtf2.getText().equals("") ||
jtf3.getText().equals("") || jtf4.getText().equals(""))
            JOptionPane.showMessageDialog(this,"All inputs are mandatory",
"Error", JOptionPane.ERROR_MESSAGE);
        else{
            try{
                int eno = Integer.parseInt(jtf1.getText());
                String ename = jtf2.getText();
                String designation = jtf3.getText();
                float salary = Float.parseFloat(jtf4.getText());

                if(ae.getSource()==jb){
                    pstmt.setInt(1,eno);
                    pstmt.setString(2,ename);
                    pstmt.setString(3,designation);
                    pstmt.setFloat(4,salary);
                    pstmt.executeUpdate();
                    JOptionPane.showMessageDialog(this,"Record Stored
Successfully","Output",JOptionPane.INFORMATION_MESSAGE);
                }

            }catch(SQLException sqle){
                JOptionPane.showMessageDialog(this,"Error Executing DDL Query
: "+sqle,"Error",JOptionPane.ERROR_MESSAGE);

            }catch(Exception e){
                JOptionPane.showMessageDialog(this,"Exception :
"+e,"Error",JOptionPane.ERROR_MESSAGE);

            }finally{
                jtf1.setText("");
                jtf2.setText("");
                jtf3.setText("");
                jtf4.setText("");
                jtf1.requestFocus();
            }
        }
    }

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

Slip 2

Q1. Write a java program to read 'N' names of your friends, store it into HashSet and display them in ascending order.

Solution:

```
import java.util.*;

class NSGStudents{
    public static void main(String args[]){
        HashSet<String> friends = new HashSet<>();
        Scanner sc = new Scanner(System.in);
        int n;
        String name;

        System.out.print("Enter how many friends:");
        n = sc.nextInt();

        for(int i=0;i<n;i++){
            System.out.print("Enter name of "+(i+1)+" friends:");
            name = sc.next();
            friends.add(name);
        }

        ArrayList<String> sortedFriends = new ArrayList<>(friends);

        Collections.sort(sortedFriends);

        System.out.println("\nFriend names in ascending order:");

        for (String sf : sortedFriends) {
            System.out.println(sf);
        }
    }
}
```

Q2. Design a servlet that provides information about a HTTP request from a client, such as IP-Address and browser type. The servlet also provides information about the server on which the servlet is running, such as the operating system type, and the names of currently loaded servlets.

Solution:

```
<!-- index.html -->
<html>
    <body>
    <form method="GET" action="nsgacademy">
        <input type="submit" value="Get Info" />
    </form>
    </body>
</html>

<!-- web.xml -->
<web-app>
    <servlet>
        <servlet-name>ServletInfo</servlet-name>
        <servlet-class>ServletInfo</servlet-class>
    </servlet>
    <servlet-mapping>
        <servlet-name>ServletInfo</servlet-name>
        <url-pattern>/nsgacademy</url-pattern>
    </servlet-mapping>
</web-app>

// ServletInfo.java
import java.io.*;
import java.util.*;
import javax.servlet.*;
import javax.servlet.http.*;

public class ServletInfo extends HttpServlet{
    protected void doGet(HttpServletRequest request, HttpServletResponse
response) throws ServletException, IOException {
        response.setContentType("text/html");
        PrintWriter out = response.getWriter();

        // Get client information
        String ipAddress = request.getRemoteAddr();
        String browserType = request.getHeader("User-Agent");
```

```
        // Get server information
        String serverInfo = getServletContext().getServerInfo();
        String osType = System.getProperty("os.name");

        // Get loaded servlets
        Enumeration<String> servletNames =
getServletContext().getServletNames();

        // String servletNames = getServletNames();

        out.println("<html>");
        out.println("<head><title>Request Information</title></head>");
        out.println("<body>");
        out.println("<h1>Client Information:</h1>");
        out.println("<p>IP Address: " + ipAddress + "</p>");
        out.println("<p>Browser Type: " + browserType + "</p>");

        out.println("<h1>Server Information:</h1>");
        out.println("<p>Server Info: " + serverInfo + "</p>");
        out.println("<p>Operating System: " + osType + "</p>");

        // out.println("<p>Loaded Servlets: " + servletNames + "</p>");

        out.println("<h1>Loaded Servlets:</h1>");
        out.println("<ul>");
        while (servletNames.hasMoreElements()) {
            String servletName = servletNames.nextElement();
            out.println("<li>" + servletName + "</li>");
        }
        out.println("</ul>");
        out.println("</body></html>");
    }

    // private String getServletNames() {
    //     StringBuilder names = new StringBuilder();
    //     getServletContext().getServletRegistrations().forEach((name,
registration) -> {
    //         names.append(name).append(", ");
    //     });
    //     // Remove the trailing comma and space
    //     if (names.length() > 0) {
    //         names.setLength(names.length() - 2);
    //     }
    //     return names.toString();
    // }
}
```

Slip 3

Q1. Write a JSP program to display the details of Patient (PNo, PName, Address, age, disease) in tabular form on browser.

Solution:

```
<%@ page contentType="text/html" language="java" import="java.sql.*" %>
<html>
<head>
    <title>NSG Academy</title>
</head>
<body>
    <h1>Patients Details</h1>
    <table border="1">
        <thead>
            <tr>
                <th>Patient Number</th>
                <th>Name</th>
                <th>Address</th>
                <th>Age</th>
                <th>Disease</th>
            </tr>
        </thead>
        <tbody>
            <%
                String dbUrl = "jdbc:postgresql://localhost/nsgacademy";
                String dbUsr = "sagrawal";
                String dbPwd = "mjain";
                Connection con = null;
                Statement stmt = null;
                ResultSet rs = null;
                try {
                    Class.forName("org.postgresql.Driver");
                    con = DriverManager.getConnection(dbUrl, dbUsr, dbPwd);

                    String query = "SELECT * FROM patients";
                    stmt = con.createStatement();
                    rs = stmt.executeQuery(query);
                    while (rs.next()) {
                        String pNo = rs.getString("pno");
                        String pName = rs.getString("pname");
                        String address = rs.getString("address");
                        String age = rs.getString("age");
                        String disease = rs.getString("disease");
            %>
```

```
            <tr>
                <td><%= pNo %></td>
                <td><%= pName %></td>
                <td><%= address %></td>
                <td><%= age %></td>
                <td><%= disease %></td>
            </tr>

            <%
                    }
                } catch (ClassNotFoundException cnfe) {
                    out.println("Driver Not Found : "+cnfe);
                } catch (SQLException sqle) {
                    out.println("SQLException : "+sqle);
                } catch (Exception e) {
                    out.println("Exception : "+e);
                } finally {
                    try { if (rs != null) rs.close(); } catch (Exception e) {}
                    try { if (stmt != null) stmt.close(); } catch (Exception e) {}
                    try { if (con != null) con.close(); } catch (Exception e) {}
                }
            %>
        </tbody>
    </table>
</body>
</html>
```

Q2. Write a Java program to create LinkedList of String objects and perform the following:

i. **Add element at the end of the list**

ii. **Delete first element of the list**

iii. **Display the contents of list in reverse order**

Solution:

```
import java.util.*;
class NSGList{
    public static void main(String args[]){
        LinkedList<String> stringList = new LinkedList<>();
        int choice;
        String str;
        Scanner sc = new Scanner(System.in);
        do
        {
            System.out.println("1:Add Element at the end of list");
            System.out.println("2:Delete First Element of the list");
            System.out.println("3:Display the contents of list in reverse order");
            System.out.println("4:Exit");

            System.out.print("Enter your choice:");
            choice = sc.nextInt();
            switch(choice){
                case 1: System.out.print("Enter a String:");
                        str = sc.next();
                        stringList.add(str);
                        System.out.println("\nContent of List after adding
element:");
                        System.out.println(stringList);
                        break;

                case 2: if(!stringList.isEmpty()){
                            stringList.removeFirst();
                            System.out.println("\nContent of List after deleting
the first element:");
                            System.out.println(stringList);
                        }else
                            System.out.println("List is empty, cannot
delete.");
                        break;
```

```
                case 3: if(!stringList.isEmpty()){
                            System.out.println("Content of List in Reverse
Order");
                            ListIterator<String> litr =
stringList.listIterator(stringList.size());
                            while(litr.hasPrevious())
                                System.out.println(litr.previous());
                            /*
                            // *****Formatted printing*****
                            System.out.print("[");
                            while(true){
                                System.out.print(litr.previous());
                                if(!litr.hasPrevious())
                                    break;
                                System.out.print(",
");
                            }
                            System.out.println("]");
                            */
                        }else
                            System.out.println("List is empty.");
                        break;
            }
        }while(choice!=4);
    }
}
```

Slip 4

Q1. Write a Java program using Runnable interface to blink Text on the frame.

Solution:

```
import java.awt.*;
import java.awt.event.*;
import javax.swing.*;
class BlinkingText extends JFrame implements Runnable{
    JLabel jl;
    Thread t;
    boolean flag;

    public BlinkingText() {
        setSize(400,400);
        setLocationRelativeTo(null);
        setTitle("NSGAcademy");

        jl = new JLabel("NSGAcademy",SwingConstants.CENTER);
        jl.setFont(new Font("TIMES NEW ROMAN", Font.BOLD, 40));
        add(jl, BorderLayout.CENTER);

        t = new Thread(this);
        t.start();

        setDefaultCloseOperation(JFrame.EXIT_ON_CLOSE);
        setVisible(true);
    }
    public void run(){
        while(true){
            try{
                flag = !flag;
                jl.setVisible(flag);
                Thread.sleep(100);
            }catch(InterruptedException ie){
                System.out.println(ie);
            }
        }
    }

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

Q2. 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**

Solution:

```
import java.util.*;

class NSGCity{
    public static void main(String args[]){
        int choice;
        String city,code;
        Scanner sc = new Scanner(System.in);

        HashMap<String,String> cityMap = new HashMap<>();

        do{
            System.out.println("1: Add a new city and its STD code");
            System.out.println("2: Remove a city");
            System.out.println("3: Search for a city name and display the STD
code");
            System.out.println("4: Exit");

            System.out.print("Enter your choice: ");
            choice = sc.nextInt();

            switch(choice){
                case 1: System.out.print("Enter city name: ");
                        city = sc.next();
                        System.out.print("Enter STD code: ");
                        code = sc.next();
                        if(cityMap.containsKey(city))
                            System.out.println("City already exists");
                        else{
                            cityMap.put(city,code);
                            System.out.println("City added successfully.");
                        }
                        break;
```

```
                case 2: System.out.print("Enter city name to be removed: ");
                        city = sc.next();
                        if(cityMap.containsKey(city)){
                            cityMap.remove(city);
                            System.out.println("City removed successfully.");
                        }else
                            System.out.println("City not
found.");
                        break;

                case 3: System.out.print("Enter city name to search: ");
                        city = sc.next();
                        if(cityMap.containsKey(city))
                            System.out.println("STD code for " + city + " is "
+ cityMap.get(city));
                        else
                            System.out.println("City not
found.");
                        break;
            }
        }while(choice != 4);
    }
}
```

Slip 5

Q1. Write a Java Program to create the hash table that will maintain the mobile number and student name. Display the details of student using Enumeration interface.

Solution:

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

        Hashtable<String, String> studentTable = new Hashtable<>();

        int choice;
        Scanner sc = new Scanner(System.in);
        String name,mobile;
        do{
            System.out.println("1 : Add student details");
            System.out.println("2 : Display all student details");
            System.out.println("3 : Exit");

            System.out.print("Enter your choice:");
            choice = sc.nextInt();

            switch(choice){
                case 1: System.out.print("Enter students name:");
                        name = sc.next();
                        System.out.print("Enter students mobile number:");
                        mobile = sc.next();
                        studentTable.put(mobile,name);
                        break;

                case 2: System.out.println("Student Details:");
                        System.out.println("Name\t\t\tMobile");
                        Enumeration<String> mobileNumbers =
studentTable.keys();
                        while (mobileNumbers.hasMoreElements()) {
                            mobile = mobileNumbers.nextElement();
                            name = studentTable.get(mobile);
                            System.out.println(name + "\t\t\t" + mobile);
                        }
                        break;
            }
        }while(choice!=3);
    }
}
```

Q2. Create a JSP page for an online multiple choice test. The questions are randomly selected from a database and displayed on the screen. The choices are displayed using radio buttons. When the user clicks on next, the next question is displayed. When the user clicks on submit, display the total score on the screen.

Solution:

```
<!-- Q2.jsp -->
<%@page language="java" session="false" import="java.util.*, java.sql.*" %>
<%
try{
    Connection con;
    Statement stmt;
    ResultSet rs;
    RequestDispatcher rd;
    Random rand=new Random();

    Class.forName("org.postgresql.Driver");
    con=DriverManager.getConnection("jdbc:postgresql://localhost/nsgacademy",
"sagrawal","mjain");
    stmt=con.createStatement();

    Cookie c[]=request.getCookies();
    int qs_id,qs_count;
    qs_id=qs_count=0;

    if(c==null)
    {
        do{
            int tempid=rand.nextInt(20);
            rs=stmt.executeQuery("select id from questions where isasked=false
and id="+tempid);
        }while(!rs.isBeforeFirst());

        rs.next();
        qs_id=rs.getInt(1);
        qs_count=1;

        Cookie c1=new Cookie("score",""+0);
        Cookie c2= new Cookie("qs_id",""+qs_id);
        Cookie c3= new Cookie("qs_count",""+qs_count);
        response.addCookie(c1);
        response.addCookie(c2);
        response.addCookie(c3);
```

```
        rs=stmt.executeQuery("select * from questions where id="+qs_id);
        rs.next();
    }
    else{
        int score=Integer.parseInt(c[0].getValue());
        qs_id=Integer.parseInt(c[1].getValue());
        qs_count=Integer.parseInt(c[2].getValue());

        rs=stmt.executeQuery("select correct_option from questions where
id="+qs_id);
        rs.next();
        String ans=rs.getString("correct_option");
        String choice=request.getParameter("ans");

        if(ans.equalsIgnoreCase(choice)==true)
            score++;
        else
            score=score;

        c[0].setValue(""+score);
        response.addCookie(c[0]);

        stmt.executeUpdate("update questions set isasked=true where id="+qs_id);

        if(qs_count==10)
        {
            stmt.executeUpdate("update questions set isasked=false");
            rd=request.getRequestDispatcher("score.jsp");
            rd.forward(request,response);

        }

        do{
            int tempid=rand.nextInt(20);
            rs=stmt.executeQuery("select id from questions where isasked=false
and id="+tempid);
        }while(!rs.isBeforeFirst());
        rs.next();
        qs_id=rs.getInt(1);
        qs_count++;

    c[1].setValue(""+qs_id);
    c[2].setValue(""+qs_count);
    response.addCookie(c[1]);
    response.addCookie(c[2]);
```

```
    rs=stmt.executeQuery("select * from questions where id="+qs_id);
    rs.next();
}
%>


    <form action="Q2.jsp" style="border:1px solid black; width:60%;
margin:5rem; ">
    <h1>Question number:<%= qs_count %></h1>
    <b><%= rs.getString(2) %></b><br>
    <hr>
    <input type=radio value="A" name="ans"><%= rs.getString(3)%></input><br>
    <input type=radio value="B" name="ans"><%= rs.getString(4)%></input><br>
    <input type=radio value="C" name="ans"><%= rs.getString(5)%></input><br>
    <input type=radio value="D" name="ans"><%= rs.getString(6)%></input><br>
    <hr>
<button style="margin:10px;">NEXT</button>
</form>

    <%

}catch(Exception e)
{
    out.print("Error:"+e);
}
%>



<!-- score.jsp -->
<%@page language="java" session="false" %>
<%
    Cookie c[]=request.getCookies();
    int score2=Integer.parseInt(c[0].getValue());

%>
<h1>
    Congratulations!! You scored <%=score2 %> out of 10!!!<br>
<h1>
```

Slip 6

Q1. 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.

Solution:

```
import java.util.*;
class NSGCollection {
    public static void main(String[] args) {
        Scanner sc = new Scanner(System.in);
        TreeSet<Integer> numbers = new TreeSet<>();
        int n,no;
        boolean ans;
        System.out.print("Enter how many integers:");
        n = sc.nextInt();
        System.out.print("Enter " + n + " integers:");
        while(n>0){
            no = sc.nextInt();
            ans = numbers.add(no);
            if(ans)
                n--;
            else
                System.out.println(no+" is duplicate. Enter unique number only");
        }

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

        List<Integer> sortedList = new ArrayList<>(numbers);

        System.out.print("\nEnter the element to be searched:");
        no = sc.nextInt();
        int index = Collections.binarySearch(sortedList,no);
        if (index >= 0)
            System.out.println(no + " found in the collection.");
        else
            System.out.println(no + " not found in the collection.");
    }
}
```

Q2. Write a java program to simulate traffic signal using threads.

Solution:

```
class TrafficSignal {
    private String currentSignal = "RED";

    public synchronized void changeSignal() {
        if (currentSignal.equals("RED")) {
            System.out.println("Changing signal from RED to GREEN");
            currentSignal = "GREEN";
        } else if (currentSignal.equals("GREEN")) {
            System.out.println("Changing signal from GREEN to YELLOW");
            currentSignal = "YELLOW";
        } else if (currentSignal.equals("YELLOW")) {
            System.out.println("Changing signal from YELLOW to RED");
            currentSignal = "RED";
        }
        notifyAll();
    }

    public synchronized String getCurrentSignal() {
        return currentSignal;
    }
}


class ChangeTrafficSignalThread extends Thread{
    private TrafficSignal trafficSignal;

    public ChangeTrafficSignalThread(TrafficSignal trafficSignal){
        this.trafficSignal = trafficSignal;
    }

    public void run(){
        while(true){
            try{
                Thread.sleep(5000);
                trafficSignal.changeSignal();
            }catch(InterruptedException ie){
                System.out.println(ie);
            }
        }
    }
}
```

```
class CarThread extends Thread{
    private TrafficSignal trafficSignal;
    public CarThread(TrafficSignal trafficSignal){
        this.trafficSignal = trafficSignal;
    }
    public void run(){
        while(true){
            try{
                Thread.sleep(2000);
                System.out.println(Thread.currentThread().getName() + "
approaching traffic signal, current signal: " +
trafficSignal.getCurrentSignal());
                if(trafficSignal.getCurrentSignal().equals("RED")){
                    System.out.println(Thread.currentThread().getName() + "
stopped at RED signal");
                    synchronized(trafficSignal){
                        trafficSignal.wait();
                    }
                }
            }catch(InterruptedException ie){
                System.out.println(ie);
            }
        }
    }
}
class NSGTrafficControl{
    public static void main(String[] args) {
        TrafficSignal trafficSignal = new TrafficSignal();
        ChangeTrafficSignalThread changerThread = new
ChangeTrafficSignalThread(trafficSignal);
        changerThread.setDaemon(true); // Set changer thread as daemon to exit
when main thread exits
        changerThread.start();
        CarThread c1 = new CarThread(trafficSignal);
        CarThread c2 = new CarThread(trafficSignal);
        CarThread c3 = new CarThread(trafficSignal);
        c1.setName("Car1");
        c2.setName("Car2");
        c3.setName("Car3");
        c1.start(); c2.start(); c3.start();
        try {
            Thread.sleep(Long.MAX_VALUE);//Main thread waits forever,other threads
are daemon
        } catch (InterruptedException ie) {System.out.println(ie);}
    }
}
```

Slip 7

Q1. 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.

Solution:

```
import java.util.*;

class RandomNumberThread extends Thread {
    public void run(){
        Random r = new Random();
        while(true){
            try {
                int no = r.nextInt(100);
                System.out.println("Random Integer generated : " + no);
                if(no%2 == 0) {
                    SquareThread st = new SquareThread(no);
                    st.start();
                }else{
                    CubeThread ct = new CubeThread(no);
                    ct.start();
                }
                Thread.sleep(1000);
            }catch (InterruptedException ie) {
                System.out.println(ie);
            }
        }
    }
}

class SquareThread extends Thread {
    int no;
    SquareThread(int no) {
        this.no = no;
    }
    public void run() {
        System.out.println("Square of " + no + " : " + (no * no));
    }
}
```

```
class CubeThread extends Thread {
    int no;
    CubeThread(int no) {
        this.no = no;
    }

    public void run() {
        System.out.println("Cube of " + no + " : " + (no * no * no));
    }
}

class TestMultiThread{
    public static void main(String args[]) {
        RandomNumberThread rnt = new RandomNumberThread();
        rnt.start();
    }
}
```

```
// Alternate Code
import java.util.*;
class NumberGeneratorThread extends Thread{
    int no;
    Random r = new Random();
    public void run(){
        try {
            while(true){
                no = r.nextInt(100);
                System.out.println("Generated Number : "+no);
                synchronized(this) {
                    notifyAll();
                }
                Thread.sleep(1000);
            }
        }catch(InterruptedException ie) {
            System.out.println(ie);
        }
    }
    synchronized int getRandomNumber() throws InterruptedException {
        wait();
        return no;
    }
}

class SquareThread extends Thread{
    NumberGeneratorThread numberGenerator;

    public SquareThread(NumberGeneratorThread numberGenerator) {
        this.numberGenerator = numberGenerator;
    }

    public void run() {
        try {
            while (true) {
                int number = numberGenerator.getRandomNumber();
                if (number % 2 == 0) {
                    System.out.println("Square of " + number + ": " + (number
* number));
                }
            }
        }catch(InterruptedException ie){
            System.out.println(ie);
        }
    }
}
```

```
class CubeThread extends Thread{
    NumberGeneratorThread numberGenerator;

    public CubeThread(NumberGeneratorThread numberGenerator) {
        this.numberGenerator = numberGenerator;
    }

    public void run() {
        try {
            while (true) {
                int number = numberGenerator.getRandomNumber();
                if (number % 2 != 0) {
                    System.out.println("Cube of "+number+": " +
(number*number*number));
                }
            }
        }catch(InterruptedException ie){
            System.out.println(ie);
        }
    }
}

class InterThreadApplication{
    public static void main(String args[]){
        NumberGeneratorThread ngt = new NumberGeneratorThread();
        SquareThread st = new SquareThread(ngt);
        CubeThread ct = new CubeThread(ngt);

        ngt.start();
        st.start();
        ct.start();
    }
}
```

Q2. 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.**

Solution:

```
import java.sql.*;
import java.util.*;
class Product{
    public static void main(String[] args){
        Scanner sc = new Scanner(System.in);
        Connection con;
        Statement stmt;
        PreparedStatement pstmt;
        ResultSet rs;
        String sql;
        int pid,cnt=1;
        String pname;
        float price;
        try {
            Class.forName("org.postgresql.Driver");
            con =
DriverManager.getConnection("jdbc:postgresql://localhost/nsgacademy","sagrawal",
"mjain");
            stmt = con.createStatement();

            sql = "create table if not exists product (" + "pid int primary
key," + "pname varchar(30)," + "price decimal(10,2))";
            stmt.executeUpdate(sql);
            System.out.println("Product table created successfully.");

            pstmt = con.prepareStatement("insert into product values(?,?,?)");
            do{
                System.out.println("Record "+cnt);
                System.out.print("Enter Product ID:");
                pid = sc.nextInt();
                pstmt.setInt(1,pid);
                System.out.print("Enter Product Name:");
                pname = sc.next();
                pstmt.setString(2,pname);
                System.out.print("Enter Product Price:");
                price = sc.nextFloat();
                pstmt.setFloat(3,price);
```

```
                try{
                    pstmt.executeUpdate();
                    System.out.println("Record Inserted Successfully");
                    cnt++;
                }catch(SQLException s){
                    System.out.println("SQLException :"+s);
                }
            }while(cnt<=5);

            rs = stmt.executeQuery("select * from product");
            System.out.println("Pid\tPname\tPrice");
            while (rs.next()) {
                pid = rs.getInt("pid");
                pname = rs.getString("pname");
                price = rs.getFloat("price");
                System.out.println(pid + "\t" + pname + "\t" + price);
            }

        }catch(ClassNotFoundException cnfe){
            System.out.println("Driver not found :"+cnfe);

        }catch(SQLException sqle){
            System.out.println("SQLException :"+sqle);

        }catch(Exception e){
            System.out.println("Exception :"+e);
        }
    }
}
```

Slip 8

Q1. Write a java program to define a thread for printing text on output screen for 'n' number of times. Create 3 threads and run them. Pass the text 'n' parameters to the thread constructor.

Example:

i. **First thread prints "COVID19" 10 times.**
ii. **Second thread prints "LOCKDOWN2020" 20 times**
iii. **Third thread prints "VACCINATED2021" 30 times**

Solution:

```
class TextThread extends Thread{
    String txt;
    int n;
    TextThread(String txt,int n){
        this.txt = txt;
        this.n = n;
    }
    public void run(){
        for(int i=0;i<n;i++){
            System.out.println(txt);
            try{
                Thread.sleep(1000);
            }catch(InterruptedException ie){
                System.out.println(ie);
            }
        }
    }
}

class TestTextThread{
    public static void main(String args[]){
        TextThread t1 = new TextThread("COVID19",10);
        TextThread t2 = new TextThread("LOCKDOWN2020",20);
        TextThread t3 = new TextThread("VACCINATED2021",30);

        t1.start();
        t2.start();
        t3.start();
    }
}
```

Q2. Write a JSP program to check whether a given number is prime or not. Display the result in red color.

Solution:

```
<%@ page language="java" contentType="text/html"%>
<html>
<head>
    <title>NSG Academy</title>
</head>
<body>
    <h2>Prime no Checker</h2>
    <form action="" method="post">
        Enter a number <input type="text" name="no">
        <input type="submit" value="isPrime">
    </form>

    <%
        if (request.getMethod().equals("POST")) {
            int no = Integer.parseInt(request.getParameter("no"));

            boolean flag = true;

            if(no < 2) {
                flag = false;
            } else {
                for(int i = 2; i <= Math.sqrt(no); i++) {
                    if(no % i == 0) {
                        flag = false;
                        break;
                    }
                }
            }

            if(flag) { %>
                <p style="color:red;"><%= no %> is a prime no.</p>
            <% } else { %>
                <p style="color:red;"><%= no %> is not a prime no.</p>
            <% }
        }
    %>
</body>
</html>
```

Slip 9

Q1. Write a Java program to create a thread for moving a ball inside a panel vertically. The ball should be created when the user clicks on the start button.

Solution:

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

class CircleThread extends Thread{

    CircleFrame parent;

    int top;
    boolean flag;

    CircleThread(CircleFrame cf){
        parent = cf;
        top = -50;
    }

    public void run(){
        try{
            while(true){
                Thread.sleep(5);

                if(top<=0)
                    flag=false;
                if(top>=parent.p1.getHeight()-40)
                    flag=true;

                if(flag)
                    top=top-1;
                else
                    top=top+1;

                parent.p1.repaint();
            }

        }catch(Exception e){

        }
    }
}
```

```
class CircleFrame extends JFrame implements ActionListener{
    MyPanel p1;
    JPanel p2;
    JButton jb1;
    CircleThread ct;
    CircleFrame(){
        setSize(650,400);
        setLocationRelativeTo(null);
        setResizable(false);
        setTitle("NSGAcademy");

        ct = new CircleThread(this);

        p1=new MyPanel(this);
        p1.setBackground(Color.DARK_GRAY);
        add(p1,BorderLayout.CENTER);

        p2=new JPanel();
        p2.setBackground(Color.cyan);
        jb1=new JButton("start");
        p2.add(jb1);
        jb1.addActionListener(this);
        add(p2,BorderLayout.SOUTH);

        setDefaultCloseOperation(JFrame.EXIT_ON_CLOSE);
        setVisible(true);
    }
    public void actionPerformed(ActionEvent ae){
        ct.start();
        jb1.setEnabled(false);
    }
    public static void main(String args[]){
        CircleFrame cf = new CircleFrame();
    }
}

class MyPanel extends Panel{
    CircleFrame parent;
    MyPanel(CircleFrame p){
        parent = p;
    }
    public void paint(Graphics g){
        g.setColor(Color.blue);
        g.fillOval(0,parent.ct.top,50,50);
    }
}
```

```
// Alternate Code
import java.awt.*;
import java.awt.event.*;
import javax.swing.event.*;
import javax.swing.*;
class CircleThread extends Thread{
    CircleFrame parent;
    int top,left;
    boolean flag;
    CircleThread(CircleFrame cf,int t,int l){
        parent = cf;
        top = t;
        left = l;
    }
    public void run(){
        try{
            while(true){
                Thread.sleep(5);

                if(top<=0)
                    flag=false;
                if(top>=parent.p1.getHeight()-40)
                    flag=true;

                if(flag)
                    top=top-1;
                else
                    top=top+1;

                parent.p1.repaint();
            }
        }catch(Exception e){
        }
    }
}
class CircleFrame extends JFrame implements ActionListener{
    MyPanel p1;
    JPanel p2;
    JButton jb1,jb2;
    int n,top,left;
    CircleThread ct[]=new CircleThread[10];
    CircleFrame(){
        setSize(650,400);
        setLocationRelativeTo(null);
        setTitle("NSGAcademy");
        setLayout(new BorderLayout());
```

```
        setResizable(false);
        p1=new MyPanel(this);
        p1.setBackground(Color.DARK_GRAY);
        add(p1,BorderLayout.CENTER);

        p2=new JPanel();
        p2.setBackground(Color.cyan);
        jb1=new JButton("start");
        p2.add(jb1);
        jb1.addActionListener(this);

        jb2=new JButton("restart");
        jb2.setEnabled(false);
        p2.add(jb2);
        jb2.addActionListener(this);

        add(p2,BorderLayout.SOUTH);
        setDefaultCloseOperation(JFrame.EXIT_ON_CLOSE);
        setVisible(true);
    }
    public void actionPerformed(ActionEvent ae){
        if(ae.getSource()==jb2)
            {
                n=0;
                top=0;
                left=0;
                jb1.setEnabled(true);
                jb2.setEnabled(false);
            }

        ct[n]=new CircleThread(this,top,left);
        ct[n].start();

        top=top+25;
        left=left+65;
        n++;
        if(n>=10){
            jb1.setEnabled(false);
            jb2.setEnabled(true);
        }
    }
    public static void main(String args[]){
        CircleFrame cf=new CircleFrame();
    }
}
```

```
class MyPanel extends Panel{
    CircleFrame parent;

    Color c[] = {Color.red, Color.green, Color.blue, Color.orange,
Color.black, Color.white, Color.GRAY, Color.cyan, Color.magenta,
Color.yellow};

    MyPanel(CircleFrame p){
        parent=p;
    }

    public void paint(Graphics g){
        for(int i=0;i<parent.n;i++){
            g.setColor(c[i]);
            g.fillOval(parent.ct[i].left,parent.ct[i].top,50,50);
        }
    }
}
```

Q2. Write a Java program using Spring to display the message "If you can't explain it simply, you don't understand it well enough".

Solution:

```
// Q2.java
package com.nsg;
import org.springframework.context.ApplicationContext;
import org.springframework.context.support.ClassPathXmlApplicationContext;

public class Q2 {
    public static void main(String[] args) {
        ApplicationContext context = new
ClassPathXmlApplicationContext("Beans.xml");

        Message message = (Message) context.getBean("message");

        System.out.println(message.getMessage());
    }
}

// Message.java
package com.nsg;
public class Message {
    private String message;
    public String getMessage() {
        return message;
    }
    public void setMessage(String message) {
        this.message = message;
    }
}

<!-- Beans.xml -->
<?xml version="1.0" encoding="UTF-8"?>
<beans xmlns="http://www.springframework.org/schema/beans"
       xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
       xsi:schemaLocation="http://www.springframework.org/schema/beans
http://www.springframework.org/schema/beans/spring-beans.xsd">

    <!-- Define a bean for the message -->
    <bean id="message" class="com.nsg.Message">
        <property name="message" value="If you can't explain it simply, you
don't understand it well enough"/>
    </bean>
</beans>
```

Slip 10

Q1. Write a Java program to display the Current Date using spring.

Solution:

```
// Q1.java
package com.nsg;
import org.springframework.context.ApplicationContext;
import org.springframework.context.support.ClassPathXmlApplicationContext;
import java.time.LocalDate;

public class Q1 {
    public static void main(String[] args) {
        // Create the application context from XML configuration
        ApplicationContext context = new
ClassPathXmlApplicationContext("Beans.xml");

        // Get the current date bean
        LocalDate currentDate = (LocalDate) context.getBean("currentDate");

        // Print the current date
        System.out.println("Current Date: " + currentDate);
    }
}

<!-- Beans.xml -->
<?xml version="1.0" encoding="UTF-8"?>
<beans xmlns="http://www.springframework.org/schema/beans"
       xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
       xsi:schemaLocation="http://www.springframework.org/schema/beans
http://www.springframework.org/schema/beans/spring-beans.xsd">

    <!-- Define the current date bean -->
    <bean id="currentDate" class="java.time.LocalDate" factory-method="now"/>
</beans>
```

Q2. Write a Java program to display first record from student table (RNo, SName, Per) onto the TextFields by clicking on button. (Assume Student table is already created).

Solution:

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

class NSGStudent extends JFrame implements ActionListener{
    JPanel jp1,jp2,jp3;
    JLabel title,roll,sname,per;
    JTextField rollText,snameText,perText;
    JButton jb;
    NSGStudent(){
        setSize(500,200);
        setLocationRelativeTo(null);
        setTitle("NSG Academy");
        jp1 = new JPanel();
        title = new JLabel("NSG Academy");
        Font f = new Font("Times New Roman",Font.BOLD,30);
        title.setFont(f);
        jp1.add(title);
        add(jp1,BorderLayout.NORTH);

        jp2 = new JPanel();
        jp2.setLayout(new GridLayout(3,2));

        roll = new JLabel("RollNumber");
        jp2.add(roll);
        rollText = new JTextField(20);
        jp2.add(rollText);

        sname = new JLabel("Name");
        jp2.add(sname);
        snameText = new JTextField(20);
        jp2.add(snameText);

        per = new JLabel("Percentage");
        jp2.add(per);
        perText = new JTextField(20);
        jp2.add(perText);


        add(jp2,BorderLayout.CENTER);
```

```
        jp3 = new JPanel();
        jb = new JButton("Fetch First Record");
        jp3.add(jb);
        jb.addActionListener(this);
        add(jp3,BorderLayout.SOUTH);

        setDefaultCloseOperation(JFrame.EXIT_ON_CLOSE);
        setVisible(true);
    }
    public void actionPerformed(ActionEvent ae){
        Connection con;
        Statement stmt;
        ResultSet rs;
        try{
            Class.forName("org.postgresql.Driver");
            con =
DriverManager.getConnection("jdbc:postgresql://localhost/nsgacademy","sagrawal",
"mjain");
            stmt =
con.createStatement(ResultSet.TYPE_SCROLL_SENSITIVE,ResultSet.CONCUR_UPDATABLE);
            rs = stmt.executeQuery("select * from student");
            if(rs.isBeforeFirst()){
                rs.next();
                rollText.setText(rs.getString(1));
                snameText.setText(""+rs.getString(2));
                perText.setText(""+rs.getFloat(3));
            }else
                JOptionPane.showMessageDialog(this,"Table is Empty", "Error",
JOptionPane.ERROR_MESSAGE);

        }catch(ClassNotFoundException cnfe){
            System.out.println("Driver not found : "+cnfe);

        }catch(SQLException sqle){
            System.out.println("SQL Exception : "+sqle);

        }catch(Exception e){
            System.out.println("Exception : "+e);
        }
    }

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

Slip 11

Q1. Design an HTML page which passes customer number to a search servlet. The servlet searches for the customer number in a database (customer table) and returns customer details if found the number otherwise display error message.

Solution:

```
<!-- index.html -->
<html>
<head>
    <title>NSG Academy</title>
</head>
<body>
    <h2>Customer Search</h2>
    <form action="search" method="POST">
        Enter customer number <input type="text" name="cno"> <br>
        <input type="submit" value="Search">
    </form>
</body>
</html>

<!-- web.xml -->
<web-app>
    <servlet>
        <servlet-name>SearchServlet</servlet-name>
        <servlet-class>SearchServlet</servlet-class>
    </servlet>

    <servlet-mapping>
        <servlet-name>SearchServlet</servlet-name>
        <url-pattern>/search</url-pattern>
    </servlet-mapping>
</web-app>

// SearchServlet.java
import java.io.*;
import javax.servlet.*;
import javax.servlet.http.*;
import java.sql.*;
public class SearchServlet extends HttpServlet {
    protected void doPost(HttpServletRequest request, HttpServletResponse
response) throws ServletException, IOException {
        PrintWriter out = response.getWriter();
        String customerNumber = request.getParameter("cno");
```

```
        Connection con = null;
        PreparedStatement pstmt = null;
        ResultSet rs = null;
        try {
            Class.forName("org.postgresql.Driver");
            con =
DriverManager.getConnection("jdbc:postgresql://localhost/tyjdbc1","postgres","
password");
            pstmt = con.prepareStatement("SELECT * FROM customer WHERE
customer_number = ?");
            pstmt.setString(1, customerNumber);
            rs = pstmt.executeQuery();
            if (rs.next()) {
                String customerName = rs.getString("customer_name");
                String customerAddress = rs.getString("customer_address");
                out.println("<html><body>");
                out.println("<h2>Customer Details</h2>");
                out.println("<p>Customer Number: " + customerNumber + "</p>");
                out.println("<p>Customer Name: " + customerName + "</p>");
                out.println("<p>Customer Address: " + customerAddress + "</p>");
                out.println("</body></html>");
            } else {
                response.setContentType("text/html");
                out.println("<html><body>");
                out.println("<h2>Error</h2>");
                out.println("<p>Customer with number " + customerNumber + " not
found.</p>");
                out.println("</body></html>");
            }
        } catch (ClassNotFoundException cnfe) {
            out.println("Driver not found : "+cnfe);
        } catch (SQLException sqle) {
            out.println("SQLException : "+sqle);
        } catch (Exception e) {
            out.println("Exception : "+e);
        } finally {
            try {
                if (rs != null) rs.close();
                if (pstmt != null) pstmt.close();
                if (con != null) con.close();
            } catch (SQLException s) {
                out.println("SQLException : "+s);
            }
        }
    }
}
```

Q2. Write a Java program to display information about all columns in the DONAR table using ResultSetMetaData.

Solution:

```
import java.sql.*;
class DonorMetaData{
    public static void main(String args[]){
        Connection con;
        Statement stmt;
        ResultSet rs;
        ResultSetMetaData rmd;
        int n;
        try{
            Class.forName("org.postgresql.Driver");
            con =
DriverManager.getConnection("jdbc:postgresql://localhost/nsgacademy","sagrawal",
"mjain");
            stmt = con.createStatement();
            rs = stmt.executeQuery("select * from donor");
            rmd = rs.getMetaData();

            n = rmd.getColumnCount();
            System.out.println("Number of columns in DONOR table : "+n);

            System.out.println("No\tName\tLabel\tType\tDisplySize\tReadOnly\t
Writable\tNULL");
            for(int i=1;i<=n;i++){
                System.out.println(i+"\t"+rmd.getColumnName(i)+"\t"+rmd.getCol
umnLabel(i)+"\t"+rmd.getColumnTypeName(i)+"\t\t"+rmd.getColumnDisplaySize(i)+"
\t"+rmd.isReadOnly(i)+"\t\t"+rmd.isWritable(i)+"\t\t"+rmd.isNullable(i));
            }
        }catch(ClassNotFoundException cnfe){
            System.out.println("Driver not found : "+cnfe);
        }catch(SQLException sqle){
            System.out.println("SQL Exception : "+sqle);
        }catch(Exception e){
            System.out.println("Exception : "+e);
        }
    }
}
```

Slip 12

Q1. Write a JSP program to check whether given number is Perfect or not. (Use Include directive).

Solution:

```
<!-- Q1.jsp -->
<%@ page language="java" contentType="text/html"%>
<html>
<head>
    <title>NSG Academy</title>
</head>
<body>
    <h2>Perfect no Checker</h2>
    <form action="" method="post">
        Enter a number <input type="text" name="no">
        <input type="submit" value="isPerfect">
    </form>
    <%@ include file="nsg_perfect.jsp" %>
</body>
</html>


<!-- nsg_perfect.jsp -->
<%@ page language="java" contentType="text/html"%>
<%
    if (request.getMethod().equals("POST")) {
        int no = Integer.parseInt(request.getParameter("no"));
        int sum = 0;

        for(int i = 1; i < no; i++) {
            if(no % i == 0) {
                sum += i;
            }
        }

        if(sum == no) { %>
            <p style="color:red;"><%= no %> is a perfect number</p>
        <% } else { %>
            <p style="color:red;"><%= no %> is not a perfect number.</p>
        <% }
    }
%>
```

Q2. Write a Java Program to create a PROJECT table with field's project_id, Project_name, Project_description, Project_Status. Insert values in the table. Display all the details of the PROJECT table in a tabular format on the screen.(using swing).

Solution:

```
import java.sql.*;
import java.awt.*;
import java.awt.event.*;
import javax.swing.*;
import javax.swing.table.*;

class ProjectFrame extends JFrame implements ActionListener{
    Connection con;
    Statement stmt;
    PreparedStatement pstmt;
    ResultSet rs;

    JLabel jl1,jl2,jl3,jl4;
    JTextField jtf1,jtf2,jtf3,jtf4;
    JPanel jp1,jp2;
    JButton jb1,jb2;
    String sql;

    JFrame jf;
    DefaultTableModel dtm;
    JTable jt;
    JScrollPane jsp;

    ProjectFrame(){
        try{
            Class.forName("org.postgresql.Driver");
            con =
DriverManager.getConnection("jdbc:postgresql://localhost/nsgacademy","sagrawal",
"mjain");
            pstmt = con.prepareStatement("insert into project
values(?,?,?,?)");

            stmt = con.createStatement();
            sql = "create table if not exists project (" + "pid int primary key,"
+ "pname varchar(30)," + "description varchar(40)," + "status varchar(20))";
            stmt.executeUpdate(sql);
            JOptionPane.showMessageDialog(this,"Project Table Created
Successfully","Output",JOptionPane.INFORMATION_MESSAGE);
```

```
setSize(700,300);
setTitle("NSGAcademy");
setLocationRelativeTo(null);

jp1 = new JPanel();
jp1.setLayout(new GridLayout(4,2));

jl1 = new JLabel("Project ID");
jp1.add(jl1);
jtf1 = new JTextField(20);
jp1.add(jtf1);

jl2 = new JLabel("Project Name");
jp1.add(jl2);
jtf2 = new JTextField(20);
jp1.add(jtf2);

jl3 = new JLabel("Project Description");
jp1.add(jl3);
jtf3 = new JTextField(20);
jp1.add(jtf3);

jl4 = new JLabel("Project Status");
jp1.add(jl4);
jtf4 = new JTextField(20);
jp1.add(jtf4);

add(jp1,BorderLayout.CENTER);

jf = new JFrame();
jf.setSize(700,300);
jf.setTitle("NSGAcademy");
jf.setLocationRelativeTo(null);
dtm = new DefaultTableModel();
jt = new JTable(dtm);
jsp = new JScrollPane(jt);
dtm.addColumn("PID");
dtm.addColumn("PNAME");
dtm.addColumn("DESCRIPTION");
dtm.addColumn("STATUS");
jf.add(jsp);
jf.setVisible(false);

jp2 = new JPanel();
jb1 = new JButton("Add Project");
jb1.addActionListener(this);
```

```
            jp2.add(jb1);
            jb2 = new JButton("Display All Projects");
            jb2.addActionListener(this);
            jp2.add(jb2);
            add(jp2,BorderLayout.SOUTH);

            setDefaultCloseOperation(JFrame.EXIT_ON_CLOSE);
            setVisible(true);
        }catch(ClassNotFoundException cnfe){
            JOptionPane.showMessageDialog(this,"Driver Not Found : " + cnfe,
"Error",JOptionPane.ERROR_MESSAGE);
            dispose();
        }catch(SQLException sqle){
            JOptionPane.showMessageDialog(this,"SQLException : " + sqle,
"Error",JOptionPane.ERROR_MESSAGE);
            dispose();
        }catch(Exception e){
            JOptionPane.showMessageDialog(this,"Exception : " + e,
"Error",JOptionPane.ERROR_MESSAGE);
            dispose();
        }
    }

    public void actionPerformed(ActionEvent ae){
        if(ae.getSource()==jb2){
            try{
                jf.setVisible(true);
                dtm = (DefaultTableModel)jt.getModel();
                dtm.setRowCount(0);

                rs = stmt.executeQuery("select * from project");
                while(rs.next()){
                    dtm.addRow(new
Object[]{rs.getInt(1),rs.getString(2),rs.getString(3),rs.getString(4)});
                }
            }catch(SQLException sqle){
                JOptionPane.showMessageDialog(this,"SQLException : " + sqle,
"Error",JOptionPane.ERROR_MESSAGE);
            }
        }

        if(ae.getSource()==jb1){
            if(jtf1.getText().equals("") || jtf2.getText().equals("") ||
jtf3.getText().equals("") || jtf4.getText().equals(""))
                JOptionPane.showMessageDialog(this,"All inputs are mandatory",
"Error", JOptionPane.ERROR_MESSAGE);
```

```
            else{
                try{
                    int pid = Integer.parseInt(jtf1.getText());
                    String pname = jtf2.getText();
                    String description = jtf3.getText();
                    String status = jtf4.getText();
                    pstmt.setInt(1,pid);
                    pstmt.setString(2,pname);
                    pstmt.setString(3,description);
                    pstmt.setString(4,status);
                    pstmt.executeUpdate();
                    JOptionPane.showMessageDialog(this,"Project added
Successfully","Output",JOptionPane.INFORMATION_MESSAGE);

                }catch(SQLException sqle){
                    JOptionPane.showMessageDialog(this,"SQLException : " +
sqle,"Error",JOptionPane.ERROR_MESSAGE);
                }catch(Exception e){
                    JOptionPane.showMessageDialog(this,"Exception : " + e,
"Error",JOptionPane.ERROR_MESSAGE);
                }finally{
                    jtf1.setText("");
                    jtf2.setText("");
                    jtf3.setText("");
                    jtf4.setText("");
                    jtf1.requestFocus();
                }
            }
        }
    }

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

Slip 13

Q1. Write a Java program to display information about the database and list all the tables in the database. (Use DatabaseMetaData).

Solution:

```
import java.sql.*;
class PostgreSqlMetaData{
    public static void main(String args[]){
        Connection con;
        DatabaseMetaData dmd;
        ResultSet rs;
        try{
            Class.forName("org.postgresql.Driver");
            con =
DriverManager.getConnection("jdbc:postgresql://localhost/nsgacademy","sagrawal",
"mjain");
            dmd = con.getMetaData();
            System.out.println("User Name : "+dmd.getUserName());
            System.out.println("URL : "+dmd.getURL());
            System.out.println("Database Product Name : " +
dmd.getDatabaseProductName());
            System.out.println("Database Version Name : " +
dmd.getDatabaseProductVersion());
            System.out.println("Driver Name : "+dmd.getDriverName());
            System.out.println("Driver Version : "+dmd.getDriverVersion());
            System.out.println("Driver Major Version : " +
dmd.getDriverMajorVersion());
            System.out.println("Driver Minor Version : " +
dmd.getDriverMinorVersion());
            rs = dmd.getTables(null,null,null,new String[]{"TABLE"});
            System.out.println("table_cat\ttable_schem\ttable_name\ttable_type\t
remarks");
            while(rs.next()){
                System.out.println(rs.getString("table_cat")+"\t\t"+rs.getStri
ng("table_schem") + "\t\t" + rs.getString("table_name") + "\t\t" +
rs.getString("table_type") + "\t\t"+rs.getString("remarks"));
            }
        }catch(ClassNotFoundException cnfe){
            System.out.println("Driver not found : "+cnfe);
        }catch(SQLException sqle){
            System.out.println("SQL Exception : "+sqle);
        }catch(Exception e){
            System.out.println("Exception : "+e);
        }}}
```

Q2. 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.

Solution:

```
import java.util.*;

class MyThread extends Thread{

    public MyThread(String name){
        super(name);
    }

    public void run(){
        System.out.println("Thread " + getName() + " is created.");

        Random random = new Random();

        int sleepTime = random.nextInt(5000);

        try{
            System.out.println("Thread " + getName() + " will sleep for " +
sleepTime + " milliseconds.");
            Thread.sleep(sleepTime);
            System.out.println("Thread " + getName() + " is dead.");

        }catch (InterruptedException ie){
            System.out.println(ie);
        }
    }
}

class ThreadLifecycle{
    public static void main(String[] args) {
        MyThread t1 = new MyThread("ChildThread1");
        MyThread t2 = new MyThread("ChildThread2");

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

Slip 14

Q1. Write a Java program for a simple search engine. Accept a string to be searched. Search the string in all text files in the current folder. Use a separate thread for each file. The result should display the filename and line number where the string is found.

Solution:

```
import java.util.*;
import java.io.*;

class SearchThread extends Thread{
    String fname;
    String str;

    SearchThread(String fname,String str){
        this.fname = fname;
        this.str = str;
    }

    public void run(){
        try{
            BufferedReader br = new BufferedReader(new FileReader(fname));
            int i=0;
            String line;

            while((line = br.readLine())!=null){
                Thread.sleep(1000);
                i++;
                if(line.contains(str))
                    System.out.println(fname+" : "+ i + " : " + line);
            }

        }catch(IOException ioe){
            System.out.println(ioe);

        }catch(InterruptedException ie){
            System.out.println(ie);
        }
    }
}
```

```
class FileSearch{
    public static void main(String args[]){
        String fnames[];
        String str;
        int n;
        int i;
        Scanner sc = new Scanner(System.in);

        System.out.print("Enter string to be searched in current directory:");
        str = sc.next();

        File f = new File(".");
        fnames = f.list();
        n = fnames.length;

        SearchThread st[] = new SearchThread[n];

        for(i=0;i<n;i++){
            st[i] = new SearchThread(fnames[i],str);
        }

        for(i=0;i<n;i++){
            st[i].start();
        }
    }
}
```

Q2. Write a JSP program to calculate sum of first and last digit of a given number.

Display sum in Red Color with font size 18.

Solution:

```
<%@ page language="java" contentType="text/html"%>
<html>
<head>
    <title>NSG Academy</title>
</head>
<body>
    <h2>Sum of First and Last Digit</h2>
    <form action="" method="post">
        Enter a no: <input type="text" name="no">
        <input type="submit" value="Calculate">
    </form>

    <%
        if (request.getMethod().equals("POST")) {
            int no = Integer.parseInt(request.getParameter("no"));

            int lastDigit = no % 10;
            int firstDigit = 0;

            while (no != 0) {
                firstDigit = no % 10;
                no = no / 10;
            }

            int sum = firstDigit + lastDigit;
    %>

    <p><font color="red" size="18">Sum of first and last digit of
<%= request.getParameter("no") %> is <%= sum %></font></p>

    <% } %>
</body>
</html>
```

Slip 15

Q1. Write a java program to display name and priority of a Thread.

Solution:

```
class NSGThread{
    public static void main(String[] args) {
        Thread thread = Thread.currentThread();

        System.out.println(thread);
        System.out.println("Thread Name: " + thread.getName());
        System.out.println("Thread Priority: " + thread.getPriority());

        thread.setName("Parent");
        thread.setPriority(10);

        System.out.println(thread);
        System.out.println("Thread Name: " + thread.getName());
        System.out.println("Thread Priority: " + thread.getPriority());
    }
}
```

Q2. Write a SERVLET program which counts how many times a user has visited a web page. If user is visiting the page for the first time, display a welcome message. If the user is revisiting the page, display the number of times visited. (Use Cookie)

Solution:

```
<!-- index.html -->
<html>
    <body>
        <form action="count">
            <input type="submit" value="click">
        </form>
    </body>
</html>

<!-- web.xml -->
<web-app>
    <servlet>
        <servlet-name>count</servlet-name>
        <servlet-class>CountServlet</servlet-class>
    </servlet>
    <servlet-mapping>
        <servlet-name>count</servlet-name>
        <url-pattern>/count</url-pattern>
    </servlet-mapping>
</web-app>
```

```
// CountServlet.java
import javax.servlet.*;
import javax.servlet.http.*;
import java.io.*;

public class CountServlet extends HttpServlet{
    public void service(HttpServletRequest request,HttpServletResponse response)
    throws ServletException,IOException{
        PrintWriter out = response.getWriter();
        int no = 1;
        Cookie c[] = request.getCookies();
        if(c==null){
            Cookie c1 = new Cookie("count",""+no);
            c1.setMaxAge(365 * 24 * 60 * 60); // Set cookie to expire in a
year
            response.addCookie(c1);
            out.println("<h2>Welcome to NSG Academy! You are visiting this
page for the first time.</h2>");
        }else{
            no = Integer.parseInt(c[0].getValue());
            no++;
            c[0].setValue(""+no);
            response.addCookie(c[0]);
            out.println("<h2>You have visited this page " + no + "
times.</h2>");
        }
    }
}
```

Slip 16

Q1. Write a java program to create a TreeSet, add some colors (String) and print out the content of TreeSet in ascending order.

Solution:

```
import java.util.*;

class ColorTree {
    public static void main(String[] args) {
        Set<String> colors = new TreeSet<>();

        colors.add("Red");
        colors.add("Blue");
        colors.add("Green");
        colors.add("Yellow");
        colors.add("Orange");

        System.out.println("Colors in ascending order:");

        for (String color : colors) {
            System.out.println(color);
        }
    }
}
```

Q2. 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)

Solution:

```
import java.sql.*;
import java.util.*;
class Teacher{
    public static void main(String[] args){
        Scanner sc = new Scanner(System.in);
        Connection con;
        PreparedStatement pstmt1,pstmt2;
        ResultSet rs;

        int tno,cnt=1;
        String tname,subject;
        try {
            Class.forName("org.postgresql.Driver");
            con =
DriverManager.getConnection("jdbc:postgresql://localhost/nsgacademy","sagrawal",
"mjain");

            pstmt1 = con.prepareStatement("insert into teacher values(?,?,?)");
            do{
                System.out.println("Record "+cnt);
                System.out.print("Enter Teacher No:");
                tno = sc.nextInt();
                pstmt1.setInt(1,tno);
                System.out.print("Enter Teacher Name:");
                tname = sc.next();
                pstmt1.setString(2,tname);
                System.out.print("Enter Subject:");
                subject = sc.next();
                pstmt1.setString(3,subject);
                try{
                    pstmt1.executeUpdate();
                    System.out.println("Record Inserted Successfully");
                    cnt++;
                }catch(SQLException s){
                    System.out.println("SQLException :"+s);
                }
            }while(cnt<=5);
```

```
            pstmt2 = con.prepareStatement("select * from teacher where
subject=?");
            pstmt2.setString(1,"java");
            rs = pstmt2.executeQuery();
            System.out.println("TNo\tTname\tSubject");
            while (rs.next()) {
                tno = rs.getInt("tno");
                tname = rs.getString("tname");
                subject = rs.getString("subject");
                System.out.println(tno + "\t" + tname + "\t" + subject);
            }
        }catch(ClassNotFoundException cnfe){
            System.out.println("Driver not found :"+cnfe);
        }catch(SQLException sqle){
            System.out.println("SQLException :"+sqle);
        }catch(Exception e){
            System.out.println("Exception :"+e);
        }
    }
}
```

Slip 17

Q1. 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.

Solution:

```
import java.util.*;
class NSGCollection {
    public static void main(String[] args) {
        Scanner sc = new Scanner(System.in);
        TreeSet<Integer> numbers = new TreeSet<>();
        int n,no;
        boolean ans;

        System.out.print("Enter how many integers:");
        n = sc.nextInt();

        System.out.print("Enter " + n + " integers:");
        while(n>0){
            no = sc.nextInt();
            ans = numbers.add(no);
            if(ans)
                n--;
            else
                System.out.println(no+" is duplicate. Enter unique number only");
        }

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

Q2. 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).

Solution:

```
import java.awt.*;
import java.awt.event.*;
import javax.swing.*;
class NumberFrame extends JFrame implements ActionListener,Runnable{
    JTextField jtf;
    JButton jb;
    Thread t;
    NumberFrame(){
        setSize(300,200);
        setLocationRelativeTo(null);
        setTitle("NSGAcademy");
        setLayout(new FlowLayout());

        t = new Thread(this);

        jtf = new JTextField(20);
        jb = new JButton("Start");
        jb.addActionListener(this);
        add(jtf);
        add(jb);
        setDefaultCloseOperation(JFrame.EXIT_ON_CLOSE);
        setVisible(true);
    }
    public void actionPerformed(ActionEvent ae){
        t.start();
        jb.setEnabled(false);
    }
    public void run(){
        try{
            for(int i=1;i<=100;i++){
                jtf.setText(""+i);
                Thread.sleep(100);
            }
        }catch(InterruptedException ie){
            System.out.println(ie);
        }
    }
    public static void main(String args[]){
        new NumberFrame();
    }
}
```

Slip 18

Q1. Write a java program to display name and priority of a Thread.

Solution:

```
class NSGThread{
    public static void main(String[] args) {
        Thread thread = Thread.currentThread();

        System.out.println(thread);
        System.out.println("Thread Name: " + thread.getName());
        System.out.println("Thread Priority: " + thread.getPriority());

        thread.setName("Parent");
        thread.setPriority(10);

        System.out.println(thread);
        System.out.println("Thread Name: " + thread.getName());
        System.out.println("Thread Priority: " + thread.getPriority());
    }
}
```

Q2. Write a SERVLET program in java to accept details of student (SeatNo, Stud_Name, Class, Total_Marks). Calculate percentage and grade obtained and display details on page.

Solution:

```
<!-- index.html -->
<html>
<head>
    <title>NSG Academy</title>
</head>
<body>
    <h2>Enter Student Details</h2>
    <form action="calculate" method="POST">
        Enter Seat Number <input type="text" name="seatNo"><br><br>
        Enter Student Name <input type="text" name="studentName"><br><br>
        Enter Class <input type="text" name="className"><br><br>
        Enter Total Marks <input type="text" name="totalMarks"><br><br>

        <input type="submit" value="Calculate">
    </form>
</body>
</html>

<!-- web.xml -->
<web-app>
    <servlet>
        <servlet-name>StudentServlet</servlet-name>
        <servlet-class>StudentServlet</servlet-class>
    </servlet>

    <servlet-mapping>
        <servlet-name>StudentServlet</servlet-name>
        <url-pattern>/calculate</url-pattern>
    </servlet-mapping>
</web-app>
```

```
// StudentServlet.java
import java.io.*;
import javax.servlet.*;
import javax.servlet.http.*;
public class StudentServlet extends HttpServlet {
    protected void doPost(HttpServletRequest request, HttpServletResponse
response) throws ServletException, IOException {
        response.setContentType("text/html");
        PrintWriter out = response.getWriter();
        int seatNo = Integer.parseInt(request.getParameter("seatNo"));
        String studentName = request.getParameter("studentName");
        String className = request.getParameter("className");
        float totalMarks = Float.parseFloat(request.getParameter("totalMarks"));
        double percentage = (totalMarks / 500.0) * 100; //Let 5 subjects
        String grade;
        if (percentage >= 90) {
            grade = "A+";
        } else if (percentage >= 80 && percentage < 90) {
            grade = "A";
        } else if (percentage >= 70 && percentage < 80) {
            grade = "B+";
        } else if (percentage >= 60 && percentage < 70) {
            grade = "B";
        } else if (percentage >= 50 && percentage < 60) {
            grade = "C+";
        } else if (percentage >= 40 && percentage < 50) {
            grade = "C";
        } else {
            grade = "Fail";
        }
        out.println("<html>");
        out.println("<head>");
        out.println("<title>Student Details</title>");
        out.println("</head>");
        out.println("<body>");
        out.println("<h2>Student Details</h2>");
        out.println("<p>Seat Number: " + seatNo + "</p>");
        out.println("<p>Student Name: " + studentName + "</p>");
        out.println("<p>Class: " + className + "</p>");
        out.println("<p>Total Marks: " + totalMarks + "</p>");
        out.println("<p>Percentage: " + percentage + "%</p>");
        out.println("<p>Grade: " + grade + "</p>");
        out.println("</body>");
        out.println("</html>");
    }
}
```

Slip 19

Q1. Write a java program to accept 'N' Integers from a user store them into LinkedList Collection and display only negative integers.

Solution:

```
import java.util.*;
class NumberList{
    public static void main(String args[]){
        List<Integer> numbers = new LinkedList<>();
        Scanner sc = new Scanner(System.in);
        int n;

        System.out.print("Enter how many integers:");
        n = sc.nextInt();

        System.out.println("Enter "+n+" integers:");
        for(int i=0;i<n;i++){
            numbers.add(sc.nextInt());
        }

        System.out.println("All negative numbers are as follows");
        for(int x : numbers){
            if(x<0)
                System.out.println(x);
        }
    }
}
```

Q2. Write a SERVLET application to accept username and password, search them into database, if found then display appropriate message on the browser otherwise display error message.

Solution:

```
<!-- login.html -->
<html>
    <body>
        <form action="login" method="POST">
            Enter USERNAME <input type="text" name="usr"> <br>
            Enter Password <input type="password" name="pwd"> <br>

            <input type="submit" value="LOGIN">
        </form>
    </body>
</html>

<!-- web.xml -->
<web-app>
    <servlet>
        <servlet-name>a</servlet-name>
        <servlet-class>LoginServlet</servlet-class>
    </servlet>
    <servlet-mapping>
        <servlet-name>a</servlet-name>
        <url-pattern>/login</url-pattern>
    </servlet-mapping>

    <welcome-file-list>
        <welcome-file>login.html</welcome-file>
    </welcome-file-list>
</web-app>
```

```
// LoginServlet.java
import javax.servlet.*;
import javax.servlet.http.*;
import java.io.*;
import java.sql.*;

public class LoginServlet extends HttpServlet{
    public void service(HttpServletRequest request,HttpServletResponse response)
    throws ServletException,IOException{
        response.setContentType("text/html");
        PrintWriter out = response.getWriter();
        String user = request.getParameter("usr");
        String pass = request.getParameter("pwd");
        Connection con = null;
        Statement stmt = null;
        ResultSet rs = null;
        try{
            Class.forName("org.postgresql.Driver");
            con =
DriverManager.getConnection("jdbc:postgresql://localhost/nsgacademy","sagrawal",
"mjain");
            stmt = con.createStatement();
            rs = stmt.executeQuery("select * from login where uname='"+user+"'
and pname='"+pass+"'");

            if(rs.isBeforeFirst())
                out.println("<h1>Welcome, " + user + "!</h1>");
            else
                out.println("<h1>Error: Invalid username or password</h1>");
        }catch(ClassNotFoundException cnfe){
            out.println("Driver not found :"+cnfe);
        }catch(SQLException sqle){
            out.println("SQLException :"+sqle);
        }catch(Exception e){
            out.println("Exception :"+e);
        }finally {
            try {
                if (rs != null) rs.close();
                if (stmt != null) stmt.close();
                if (con != null) con.close();
            } catch (SQLException s) {
                s.printStackTrace(out);
            }
        }
    }
}
```

Slip 20

Q1. Create a JSP page to accept a number from a user and display it in words:

Example: 123 – One Two Three. The output should be in red color.

Solution:

```
<%@ page language="java" contentType="text/html"%>
<html>
<head>
    <title>NSG Academy</title>
</head>
<body>
    <h2>Number to Words Converter</h2>
    <form action="" method="post">
        Enter a number <input type="text" name="no">
        <input type="submit" value="Convert">
    </form>

    <%
        if (request.getMethod().equals("POST")) {
            String str = request.getParameter("no");
            int no = Integer.parseInt(str);

            String[] words = {"Zero", "One", "Two", "Three", "Four", "Five",
"Six", "Seven", "Eight", "Nine"};
            String result = "";

            while (no != 0) {
                int d = no % 10;
                result = words[d] + " " + result;
                no = no / 10;
            }
    %>

    <p style="color:red;"><%= str %> - <%= result.trim() %></p>

    <% } %>
</body>
</html>
```

Q2. Write a java program to blink image on the JFrame continuously.

Solution:

```
import java.awt.*;
import java.awt.event.*;
import javax.swing.*;
class BlinkingImage extends JFrame implements Runnable{
    JLabel imageLabel;
    Thread t;
    boolean flag;

    public BlinkingImage() {
        setSize(600,400);
        setLocationRelativeTo(null);
        setTitle("NSGAcademy");

        ImageIcon icon = new ImageIcon("nsgacademy.jpeg");
        imageLabel = new JLabel(icon);
        add(imageLabel, BorderLayout.CENTER);

        t = new Thread(this);
        t.start();

        setDefaultCloseOperation(JFrame.EXIT_ON_CLOSE);
        setVisible(true);
    }

    public void run(){
        while(true){
            try{
                flag = !flag;
                imageLabel.setVisible(flag);
                Thread.sleep(100);
            }catch(InterruptedException ie){
                System.out.println(ie);
            }
        }
    }

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

Slip 21

Q1. Write a java program to accept 'N' Subject Names from a user store them into LinkedList Collection and Display them by using Iterator interface.

Solution:

```
import java.util.*;
class SubjectList{
    public static void main(String args[]){
        List<String> subjects = new LinkedList<>();
        Scanner sc = new Scanner(System.in);
        int n;

        System.out.print("Enter how many subjects:");
        n = sc.nextInt();

        System.out.println("Enter "+n+" subject names:");
        for(int i=0;i<n;i++){
            subjects.add(sc.next());
        }

        System.out.println("All subjects are as follows");
        Iterator<String> itr = subjects.iterator();
        while(itr.hasNext()){
            System.out.println(itr.next());
        }

        /*
        // ****Formatted Output****
        System.out.print("[");
        while(true){
            System.out.print(itr.next());
            if(!itr.hasNext())
                break;
            System.out.print(", ");
        }
        System.out.println("]");
        */
    }
}
```

Q2. 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)

Solution:

```
import java.util.*;
class ProducerConsumer{
    LinkedList<Integer> mylist = new LinkedList<Integer>();
    synchronized void produce() throws InterruptedException{
        int x = 0;
        while(true){
            while(mylist.size()==5){
                wait();
            }
            System.out.println("Producer produced : "+x);
            mylist.add(x++);
            notify();
            Thread.sleep(1000);
        }
    }
    synchronized void consume() throws InterruptedException{
        while (true) {
            while (mylist.isEmpty()) {
                wait();
            }
            int x = mylist.removeFirst();
            System.out.println("Consumer consumed: " + x);
            notify();
            Thread.sleep(1000);
        }
    }
}
class ProducerThread extends Thread{
    ProducerConsumer ob;
    ProducerThread(ProducerConsumer o){
        ob = o;
    }
    public void run(){
        try{
            ob.produce();
        }catch(InterruptedException ie){
            System.out.println(ie);
        }
    }}
```

```
class ConsumerThread extends Thread{

    ProducerConsumer ob;

    ConsumerThread(ProducerConsumer o){
        ob = o;
    }

    public void run(){
        try{
            ob.consume();
        }catch(InterruptedException ie){
            System.out.println(ie);
        }
    }
}

class TestSynchronization{
    public static void main(String[] args) {
        ProducerConsumer ob = new ProducerConsumer();

        ProducerThread pt1 = new ProducerThread(ob);
        ConsumerThread ct1 = new ConsumerThread(ob);

        pt1.start();
        ct1.start();
    }
}
```

Slip 22

Q1. Write a Menu Driven program in Java for the following: Assume Employee table with attributes (ENo, EName, Salary) is already created. 1. Insert 2. Update 3. Display 4. Exit.

Solution:

```
import java.sql.*;
import java.util.*;
class EmployeeMenu{
    public static void main(String args[]){
        Connection con;
        PreparedStatement psinsert,psupdatename,psupdatesalary,psselect;
        ResultSet rs;
        int choice;
        int eno,ans;
        String ename;
        float esalary;
        Scanner sc = new Scanner(System.in);
        try{
            Class.forName("org.postgresql.Driver");
            con =
DriverManager.getConnection("jdbc:postgresql://localhost/nsgacademy","sagrawal",
"mjain");

            psinsert = con.prepareStatement("insert into employee values(?,?,?)");
            psupdatename = con.prepareStatement("update employee set name=?
where id=?");
            psupdatesalary = con.prepareStatement("update employee set salary=?
where id=?");
            psselect = con.prepareStatement("select * from employee");

            do{
                System.out.println("1:Insert");
                System.out.println("2:Update");
                System.out.println("3:Display");
                System.out.println("4:Exit");
                System.out.print("Enter your choice:");
                choice = sc.nextInt();

                switch(choice){
                    case 1: System.out.print("Enter employee no:");
                            eno = sc.nextInt();
```

```
                        sc.nextLine();

                        System.out.print("Enter employee name:");
                        ename = sc.nextLine();
                        System.out.print("Enter employee salary:");
                        esalary = sc.nextFloat();
                        psinsert.setInt(1,eno);
                        psinsert.setString(2,ename);
                        psinsert.setFloat(3,esalary);
                        try{
                            psinsert.executeUpdate();
                            System.out.println("Inserted successfully");
                        }catch(SQLException se){
                            System.out.println("record already exist");
                        }
                        break;

                    case 2: System.out.print("Enter employee no to be updated:");
                        eno = sc.nextInt();
                        System.out.println("1:Update name");
                        System.out.println("2:Update salary");
                        int choice1;
                        System.out.print("Enter your choice:");
                        choice1 = sc.nextInt();
                        sc.nextLine();
                        if(choice1==1){
                            System.out.print("Enter new name:");
                            ename = sc.nextLine();
                            psupdatename.setString(1,ename);
                            psupdatename.setInt(2,eno);

                            ans = psupdatename.executeUpdate(); //step-4
                            if(ans==0)
                                System.out.println("Employee No : "+ eno +
" not present");
                            else
                                System.out.println("record updated
successfully");

                        }else if(choice1==2){
                            System.out.print("Enter new salary:");
                            esalary = sc.nextFloat();
                            psupdatesalary.setFloat(1,esalary);
                            psupdatesalary.setInt(2,eno);

                            ans = psupdatesalary.executeUpdate();
```

```
                                if(ans==0)
                                    System.out.println("Employee No : "+eno+" not
present");
                                else
                                    System.out.println("record updated
successfully");
                            }else
                                System.out.println("Invalid choice");
                            break;

                    case 3: rs = psselect.executeQuery();
                            if(rs.isBeforeFirst()){
                                System.out.println("NO\t\tNAME\t\tSALARY");
                                while(rs.next()){
                                    System.out.println(rs.getInt(1)+"\t\t" +
rs.getString(2) + "\t\t"+rs.getFloat(3));
                                }
                            }else
                                System.out.println("Employee Table is Empty");
                            break;
                }
            }while(choice!=4);
        }catch(ClassNotFoundException cnfe){
            System.out.println("Driver Not Found : "+cnfe);
        }catch(SQLException sqle){
            System.out.println("SQL Exception : "+sqle);
        }catch(Exception e){
            System.out.println("Exception : "+e);
        }
    }
}
```

Q2. Write a JSP program which accepts UserName in a TextBox and greets the user according to the time on server machine.

Solution:

```
<%@ page language="java" contentType="text/html" import="java.util.*"%>
<html>
<head>
    <title>NSG Academy</title>
</head>
<body>
    <h2>Welcome!</h2>
    <form action="" method="post">
        Enter your name: <input type="text" name="usr">
        <input type="submit" value="Greet">
    </form>

    <%
        if (request.getMethod().equals("POST")) {
            String usr = request.getParameter("usr");

            Date today = new Date();
            Calendar calendar = Calendar.getInstance();
            calendar.setTime(today);
            int hour = calendar.get(Calendar.HOUR_OF_DAY);

            String greeting = "";
            if (hour < 12) {
                greeting = "Good morning";
            } else if (hour < 18) {
                greeting = "Good afternoon";
            } else {
                greeting = "Good evening";
            }
    %>

    <p><%= greeting %>, <%= usr %>!</p>

    <% } %>
</body>
</html>
```

Slip 23

Q1. Write a java program to accept a String from a user and display each vowel from a String after every 3 seconds.

Solution:

```
import java.util.*;
class VowelThread extends Thread{
    String str;
    VowelThread(String str){
        this.str = str;
    }
    public void run(){
        for(int i=0;i<str.length();i++) {
            char ch = str.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.println(ch);
                try{
                    Thread.sleep(3000);
                }catch(InterruptedException ie) {
                    System.out.println(ie);
                }
            }
        }
    }
}

class MyString{
    public static void main(String args[]){
        Scanner sc = new Scanner(System.in);
        String str;
        System.out.print("Enter a string:");
        str = sc.next();
        VowelThread vt = new VowelThread(str);
        vt.start();
    }
}
```

Q2. 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.

Solution:

```
// For execution : java NSGStudents krish anurag disha bhav

import java.util.*;
class NSGStudents{
    public static void main(String args[]){
        List<String> students = new ArrayList<>();
        for(int i=0;i<args.length;i++){
            students.add(args[i]);
        }

        System.out.println("All student names using Iterator");
        Iterator<String> itr = students.iterator();
        while(itr.hasNext())
            System.out.println(itr.next());


        System.out.println("All student names using ListIterator");
        ListIterator<String> litr = students.listIterator();
        while(litr.hasNext())
            System.out.println(litr.next());
    }
}
```

Slip 24

Q1. Write a java program to scroll the text from left to right continuously.

Solution:

```
import java.awt.*;
import java.awt.event.*;
import javax.swing.*;
class ScrollThread extends Thread{
    ScrollTextFrame stm;

    ScrollThread(ScrollTextFrame stm){
        this.stm = stm;
    }
    public void run(){
        while(true){
            try{
                if(stm.x < stm.getWidth())
                    stm.x = stm.x + 5;
                else
                    stm.x=0;

                stm.repaint();

                Thread.sleep(10);
            }catch(InterruptedException ie) {
                System.out.println(ie);
            }
        }
    }
}
class ScrollTextFrame extends JFrame{
    String str = "Welcome to NSGAcademy";
    ScrollThread t;
    int x;
    ScrollTextFrame(){
        setSize(700,300);
        setTitle("NSGAcademy");
        setLocationRelativeTo(null);

        t = new ScrollThread(this);
        t.start();

        setDefaultCloseOperation(JFrame.EXIT_ON_CLOSE);
        setVisible(true);
    }
```

```
    public void paint(Graphics g){
        super.paint(g);
        g.setColor(Color.RED);
        g.drawString(str, x, 50);
    }

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

```
// Alternate Code
import java.awt.*;
import java.awt.event.*;
import javax.swing.*;
class ScrollThread extends Thread{
    ScrollTextFrame stm;
    boolean flag;
    ScrollThread(ScrollTextFrame stm){
        this.stm = stm;
    }
    public void run(){
        while(true){
            try{
                if(stm.x == 0)
                    flag = true;

                if(stm.x == stm.getWidth()-150)
                    flag = false;

                if(flag)
                    stm.x = stm.x + 5;
                else
                    stm.x = stm.x - 5;

                stm.repaint();

                Thread.sleep(10);
            }catch(InterruptedException ie) {
                System.out.println(ie);
            }
        }
    }
}
class ScrollTextFrame extends JFrame{
    String str = "Welcome to NSGAcademy";
    ScrollThread t;
    int x;
    ScrollTextFrame(){
        setSize(700,300);
        setTitle("NSGAcademy");
        setLocationRelativeTo(null);
        t = new ScrollThread(this);
        t.start();
        setDefaultCloseOperation(JFrame.EXIT_ON_CLOSE);
        setVisible(true);
    }
```

```
    public void paint(Graphics g){
        super.paint(g);
        g.setColor(Color.RED);
        g.drawString(str, x, 50);
    }

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

Q2. Write a JSP script to accept username and password from user, if they are same then display "Login Successfully" message in Login.html file, otherwise display "Login Failed" Message in Error.html file.

Solution:

```
<!-- Q2.jsp -->
<%@ page language="java" contentType="text/html"%>
<html>
<head>
    <title>NSG Academy</title>
</head>
<body>
    <form action="" method="post">
        Enter username <input type="text" name="usr"> <br>
        Enter password <input type="password" name="pwd"> <br>
        <input type="submit" value="LOGIN">
    </form>

    <%
        if (request.getMethod().equals("POST")) {
            String usr = request.getParameter("usr");
            String pwd = request.getParameter("pwd");

            if (usr.equals(pwd)) {
                response.sendRedirect("login.html");
            } else {
                response.sendRedirect("error.html");
            }
    %>

    <% } %>
</body>
</html>
```

```
<!-- login.html -->
<html>
<head>
    <title>Login Successful</title>
</head>
<body>
    <h2>Login Successful</h2>
    <p>Welcome to NSG Academy</p>
</body>
</html>


<!-- error.html -->
<html>
<head>
    <title>Login Failed</title>
</head>
<body>
    <h2>Login Failed</h2>
    <p>Username and password do not match.</p>
</body>
</html>
```

Slip 25

Q1. Write a JSP program to accept Name and Age of Voter and check whether he is eligible for voting or not.

Solution:

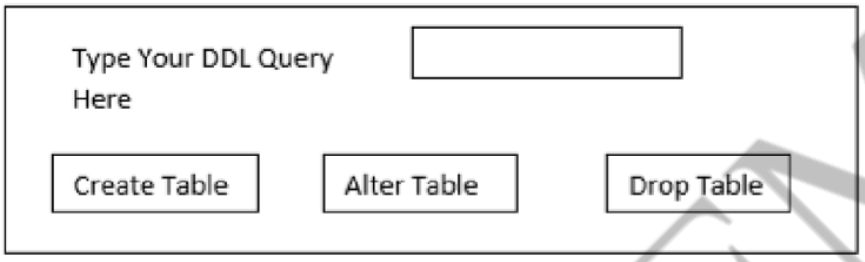
```
<%@ page language="java" contentType="text/html"%>
<html>
<head>
    <title>NSG Academy</title>
</head>
<body>
    <h2>Voter Eligibility Check</h2>
    <form action="" method="post">
        Enter your name <input type="text" name="name"><br><br>
        Enter your age <input type="text" name="age"><br><br>
        <input type="submit" value="Check Eligibility">
    </form>

    <%
        if (request.getMethod().equals("POST")) {
            String name = request.getParameter("name");
            int age = Integer.parseInt(request.getParameter("age"));
    %>

    <% if (age>=18) { %>
        <p style="color:green;"><%= name %> is eligible for voting.</p>
    <% } else { %>
        <p style="color:red;"><%= name %> is not eligible for voting.</p>
    <% } %>

    <% } %>
</body>
</html>
```

Q2. Write a Java Program for the following: Assume database is already created.



Solution:

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

class ExecuteDDL extends JFrame implements ActionListener{
    JLabel jl;
    JTextField jtf;
    JButton jb1,jb2,jb3;
    Connection con;
    Statement stmt;

    ExecuteDDL(){
        try{
            Class.forName("org.postgresql.Driver");

            con =
DriverManager.getConnection("jdbc:postgresql://localhost/nsgacademy","sagrawal",
"mjain");
            stmt = con.createStatement();

            setSize(700,300);
            setTitle("NSGAcademy");
            setLocationRelativeTo(null);
            setLayout(new FlowLayout());

            jl = new JLabel("Type Your DDL Query Here");
            jtf = new JTextField(20);
            jb1 = new JButton("Create Table");
            jb2 = new JButton("Alter Table");
            jb3 = new JButton("Drop Table");
```

```
            add(jl);
            add(jtf);
            add(jb1);
            add(jb2);
            add(jb3);

            jb1.addActionListener(this);
            jb2.addActionListener(this);
            jb3.addActionListener(this);

            setDefaultCloseOperation(JFrame.EXIT_ON_CLOSE);
            setVisible(true);

        }catch(ClassNotFoundException cnfe){
            JOptionPane.showMessageDialog(this,"Driver Not Found : "+cnfe,
"Error", JOptionPane.ERROR_MESSAGE);
            dispose();

        }catch(SQLException sqle){
            JOptionPane.showMessageDialog(this,"SQLException : "+sqle,
"Error", JOptionPane.ERROR_MESSAGE);
            dispose();

        }catch(Exception e){
            JOptionPane.showMessageDialog(this,"Exception : "+e, "Error",
JOptionPane.ERROR_MESSAGE);
            dispose();
        }
    }
```

```
    public void actionPerformed(ActionEvent ae){
        int ans;
        String sql = jtf.getText();

        try{
            if(sql.equals(""))
                JOptionPane.showMessageDialog(this,"TextField cannot be
empty", "Error", JOptionPane.ERROR_MESSAGE);
            else{
                if(ae.getSource()==jb1 && sql.startsWith("create")){
                    stmt.executeUpdate(sql);
                    JOptionPane.showMessageDialog(this,"Table is created
Successfully", "Output",JOptionPane.INFORMATION_MESSAGE);

                }else if(ae.getSource()==jb2 && sql.startsWith("alter")){
                    stmt.executeUpdate(sql);
                    JOptionPane.showMessageDialog(this,"Table is altered
Successfully", "Output",JOptionPane.INFORMATION_MESSAGE);
                }else if(ae.getSource()==jb3 && sql.startsWith("drop")){
                    stmt.executeUpdate(sql);
                    JOptionPane.showMessageDialog(this,"Table is dropped
Successfully", "Output",JOptionPane.INFORMATION_MESSAGE);
                }else
                    throw new SQLException();
            }
        }catch(SQLException sqle){
            JOptionPane.showMessageDialog(this,"Error Executing DDL Query :
"+sqle, "Error",JOptionPane.ERROR_MESSAGE);
        }finally{
            jtf.setText("");
            jtf.requestFocus();
        }
    }

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

Slip 26

Q1. Write a Java program to delete the details of given employee (ENo EName Salary). Accept employee ID through command line. (Use PreparedStatement Interface)

Solution:

```
import java.sql.*;
class Employee{
    public static void main(String args[]){
        Connection con;
        PreparedStatement pstmt;
        int ans;

        try{
            if(args.length==1){
                Class.forName("org.postgresql.Driver");
                con =
DriverManager.getConnection("jdbc:postgresql://localhost/nsgacademy","sagrawal",
"mjain");
                pstmt = con.prepareStatement("delete from employee where
eno=?");

                pstmt.setInt(1,Integer.parseInt(args[0]));
                ans = pstmt.executeUpdate();
                if(ans==0)
                    System.out.println("Employee having given ID not present");
                else
                    System.out.println("Employee having given ID deleted
successfully");
            }else
                System.out.println("Invalid Input");

        }catch(ClassNotFoundException cnfe){
            System.out.println("Driver not found :"+cnfe);

        }catch(SQLException sqle){
            System.out.println("SQL Exception :"+sqle);

        }catch(Exception e){
            System.out.println("Exception :"+e);
        }
    }
}
```

Q2. Write a JSP program to calculate sum of first and last digit of a given number.

Display sum in Red Color with font size 18.

Solution:

```
<%@ page language="java" contentType="text/html"%>
<html>
<head>
    <title>NSG Academy</title>
</head>
<body>
    <h2>Sum of First and Last Digit</h2>
    <form action="" method="post">
        Enter a no: <input type="text" name="no">
        <input type="submit" value="Calculate">
    </form>

    <%
        if (request.getMethod().equals("POST")) {
            int no = Integer.parseInt(request.getParameter("no"));

            int lastDigit = no % 10;
            int firstDigit = 0;

            while (no != 0) {
                firstDigit = no % 10;
                no = no / 10;
            }

            int sum = firstDigit + lastDigit;
    %>

    <p><font color="red" size="18">Sum of first and last digit of
<%= request.getParameter("no") %> is <%= sum %></font></p>

    <% } %>
</body>
</html>
```

Slip 27

Q1. Write a Java Program to display the details of College (CID, CName, address, Year) on JTable.

Solution:

```
import java.sql.*;
import javax.swing.*;
import javax.swing.table.*;
import java.awt.*;
class CollegeTable{
    public static void main(String args[]){
        try{
            Connection con; Statement stmt; ResultSet rs;
            Class.forName("org.postgresql.Driver");
            con =
DriverManager.getConnection("jdbc:postgresql://localhost/nsgacademy","sagrawal",
"mjain");
            stmt = con.createStatement();
            rs = stmt.executeQuery("select * from college");
            DefaultTableModel dtm = new DefaultTableModel();
            JTable jt = new JTable(dtm);
            JScrollPane jsp = new JScrollPane(jt);
            dtm.addColumn("CID");
            dtm.addColumn("CNAME");
            dtm.addColumn("ADDRESS");
            dtm.addColumn("YEAR");
            while(rs.next()){
                dtm.addRow(new Object[] {rs.getInt(1), rs.getString(2),
rs.getString(3), rs.getInt(4)});
            }
            JFrame jf = new JFrame();
            jf.setSize(300,300);
            jf.setLocationRelativeTo(null);
            jf.setTitle("NSGAcademy");
            jf.add(jsp);
            jf.setDefaultCloseOperation(JFrame.EXIT_ON_CLOSE);
            jf.setVisible(true);
        }catch(ClassNotFoundException cnfe){
            System.out.println(cnfe);
        }catch(SQLException sqle){
            System.out.println(sqle);
        }catch(Exception e){
            System.out.println(e);
        }}}
```

Q2. Write a SERVLET program to change inactive time interval of session.

Solution:

```
<!-- index.html -->
<head>
    <title>NSG Academy</title>
</head>
<body>
    <h1>Change Session Timeout</h1>
    <form action="timeout" method="post">
        New Timeout (seconds) <input type="text" name="timeout">
        <input type="submit" value="Change Timeout">
    </form>
</body>
</html>

<!-- web.xml -->
<web-app>
    <servlet>
        <servlet-name>timeout</servlet-name>
        <servlet-class>SessionTimeoutServlet</servlet-class>
    </servlet>

    <servlet-mapping>
        <servlet-name>timeout</servlet-name>
        <url-pattern>/timeout</url-pattern>
    </servlet-mapping>

    <servlet>
        <servlet-name>expired</servlet-name>
        <servlet-class>SessionExpiredServlet</servlet-class>
    </servlet>

    <servlet-mapping>
        <servlet-name>expired</servlet-name>
        <url-pattern>/expired</url-pattern>
    </servlet-mapping>
</web-app>
```

```
// SessionTimeoutServlet.java
import java.io.*;
import javax.servlet.*;
import javax.servlet.http.*;
public class SessionTimeoutServlet extends HttpServlet {
    protected void doPost(HttpServletRequest request,HttpServletResponse response)
            throws ServletException, IOException {
        HttpSession session = request.getSession();
        String timeoutString = request.getParameter("timeout");
        int timeout = Integer.parseInt(timeoutString);
        session.setMaxInactiveInterval(timeout);
        response.setContentType("text/html");
        PrintWriter out = response.getWriter();
        out.println("<html><body>");
        out.println("Session timeout changed successfully to " + timeout + "
seconds");
        out.println("Session ID :"+session.getId());
        out.println("<br><a href='expired'>Click here to simulate session
expiry</a>");
        out.println("</body></html>");
    }
}
// SessionExpiredServlet.java
import java.io.*;
import javax.servlet.*;
import javax.servlet.http.*;
public class SessionExpiredServlet extends HttpServlet {
    protected void doGet(HttpServletRequest request, HttpServletResponse
response) throws ServletException, IOException {
        response.setContentType("text/html");
        PrintWriter out = response.getWriter();
        HttpSession session = request.getSession();
        if (session.isNew()) {
            out.println("<html><body>");
            out.println("<h1>Session Expired</h1>");
            out.println("<p>Your session has expired. Please login again.</p>");
            out.println("Current Session ID :"+session.getId());
            out.println("</body></html>");
        } else {
            out.println("<html><body>");
            out.println("<h1>Session is active yet</h1>");
            out.println("Current Session ID :"+session.getId());
            out.println("</body></html>");
        }
    }
}
```

Slip 28

Q1. Write a JSP script to accept a String from a user and display it in reverse order.

Solution:

```
<%@ page language="java" contentType="text/html"%>
<html>
<head>
    <title>NSG Academy</title>
</head>
<body>
    <h2>Reverse String Display</h2>
    <form action="" method="post">
        Enter a string <input type="text" name="str">
        <input type="submit" value="Reverse">
    </form>

    <%
        if (request.getMethod().equals("POST")) {
            String str = request.getParameter("str");

            //Reverse Using self logic
            char[] strArray = str.toCharArray();
            int i = 0;
            int j = strArray.length - 1;
            while (i < j) {
                char t = strArray[i];
                strArray[i] = strArray[j];
                strArray[j] = t;
                i++;
                j--;
            }

            String reversedStr = new String(strArray);

            //Reverse Using Built-in function
            //StringBuilder reversedStr = new StringBuilder(str).reverse();
    %>

    <p>Original String: <%= str %></p>
    <p>Reversed String: <%= reversedStr %></p>

    <% } %>
</body>
</html>
```

Q2. Write a java program to display name of currently executing Thread in multithreading.

Solution:

```
class MyThread extends Thread{
    public void run(){
        System.out.println(Thread.currentThread());
        try{
            Thread.sleep(2000);
        }catch(InterruptedException ie){
            System.out.println(ie);
        }
    }
}

class TestMyThread{
    public static void main(String args[]){
        System.out.println(Thread.currentThread());
        MyThread ob1 = new MyThread();
        MyThread ob2 = new MyThread();
        MyThread ob3 = new MyThread();

        ob1.setName("Child1");
        ob2.setName("Child2");
        ob3.setName("Child3");

        ob1.start();
        ob2.start();
        ob3.start();
    }
}
```

Slip 29

Q1. Write a Java program to display information about all columns in the DONAR table using ResultSetMetaData.

Solution:

```
import java.sql.*;
class DonorMetaData{
    public static void main(String args[]){
        Connection con;
        Statement stmt;
        ResultSet rs;
        ResultSetMetaData rmd;
        int n;
        try{
            Class.forName("org.postgresql.Driver");
            con =
DriverManager.getConnection("jdbc:postgresql://localhost/nsgacademy","sagrawal",
"mjain");
            stmt = con.createStatement();
            rs = stmt.executeQuery("select * from donor");
            rmd = rs.getMetaData();

            n = rmd.getColumnCount();
            System.out.println("Number of columns in DONOR table : "+n);

            System.out.println("No\tName\tLabel\tType\tDisplySize\tReadOnly\t
Writable\tNULL");
            for(int i=1;i<=n;i++){
                System.out.println(i + "\t" + rmd.getColumnName(i) + "\t" +
rmd.getColumnLabel(i) + "\t" + rmd.getColumnTypeName(i)+ "\t\t" +
rmd.getColumnDisplaySize(i) + "\t" + rmd.isReadOnly(i) + "\t\t" +
rmd.isWritable(i) + "\t\t"+rmd.isNullable(i));
            }
        }catch(ClassNotFoundException cnfe){
            System.out.println("Driver not found : "+cnfe);
        }catch(SQLException sqle){
            System.out.println("SQL Exception : "+sqle);
        }catch(Exception e){
            System.out.println("Exception : "+e);
        }
    }
}
```

Q2. 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**

Solution:

```
import java.util.*;
class NSGList{
    public static void main(String args[]){
        LinkedList<Integer> integerList = new LinkedList<>();
        int choice, x;
        Scanner sc = new Scanner(System.in);
        do
        {
            System.out.println("1:Add Element at first position");
            System.out.println("2:Delete Last Element");
            System.out.println("3:Display the size of link list");
            System.out.println("4:Exit");
            System.out.print("Enter your choice:");
            choice = sc.nextInt();
            switch(choice){
                case 1: System.out.print("Enter an integer:");
                        x = sc.nextInt();
                        integerList.addFirst(x);
                        System.out.println("\nContent of List after adding
element:");
                        System.out.println(integerList);
                        break;
                case 2: if(!integerList.isEmpty()){
                            integerList.removeLast();
                            System.out.println("\nContent of List after
deleting the last element:");
                            System.out.println(integerList);
                        }else
                            System.out.println("List is empty, cannot
delete.");
                        break;
                case 3: System.out.println("Size of List : "+integerList.size());
                        break;
            }
        }while(choice!=4);
    }}
```

Slip 30

Q1. Write a java program for the implementation of synchronization.

Solution:

```
import java.util.*;
class ReadThread extends Thread{
    A ob;
    ReadThread(A o){
        ob = o;
    }
    public void run(){
        try{
            ob.accept();
        }catch(InterruptedException ie){}
    }
}

class WriteThread extends Thread{
    A ob;
    WriteThread(A o){
        ob = o;
    }
    public void run(){
        try{
            ob.display();
        }catch(InterruptedException ie){}
    }
}

class AddThread extends Thread{
    A ob;
    AddThread(A o){
        ob = o;
    }
    public void run(){
        try{
            ob.calculate();
        }catch(InterruptedException ie){}
    }
}
```

```
class A{
    int a;
    int b;
    public synchronized void accept() throws InterruptedException{
        Random r = new Random();
        a = r.nextInt(100);
        b = r.nextInt(100);
        notifyAll();
        System.out.println(Thread.currentThread()+" generated numbers, now you
can write/calculate");
    }
    public synchronized void display() throws InterruptedException{
        System.out.println(Thread.currentThread()+" is waiting for
readThread");
        wait();
        System.out.println("First number:"+a);
        System.out.println("Second number:"+b);
    }
    public synchronized void calculate() throws InterruptedException{
        int c;
        System.out.println(Thread.currentThread()+" is waiting for
readThread");
        wait();
        c = a+b;
        System.out.println("Total : "+c);
    }
}

class interthread{
    public static void main(String args[]){
        A ob = new A();

        ReadThread rt = new ReadThread(ob);
        WriteThread wt = new WriteThread(ob);
        AddThread at = new AddThread(ob);

        at.start();
        wt.start();
        try{
            Thread.sleep(10000);
        }catch(InterruptedException ie){}

        rt.start();
    }
}
```

Q2. Write a Java Program for the implementation of scrollable ResultSet. Assume Teacher table with attributes (TID, TName, Salary) is already created.

Solution:

```
import java.sql.*;
class ScrollableTeacher{
    public static void main(String args[]){
        Connection con;
        Statement stmt;
        ResultSet rs;
        try{
            Class.forName("org.postgresql.Driver");
            con =
DriverManager.getConnection("jdbc:postgresql://localhost/nsgacademy","sagrawal",
"mjain");
            stmt =
con.createStatement(ResultSet.TYPE_SCROLL_SENSITIVE,ResultSet.CONCUR_UPDATABLE);
            rs = stmt.executeQuery("select * from teacher");

            rs.last();
            System.out.println("Record "+rs.getRow()+" : "+rs.getInt(1) + "\t"
+ rs.getString(2)+"\t"+rs.getFloat(3));

            rs.first();
            System.out.println("Record "+rs.getRow()+" : " + rs.getInt(1) +
"\t" + rs.getString(2)+"\t"+rs.getFloat(3));

            rs.absolute(4);
            System.out.println("Record "+rs.getRow()+" : " + rs.getInt(1) +
"\t" + rs.getString(2)+"\t"+rs.getFloat(3));

            rs.relative(-1);
            System.out.println("Record "+rs.getRow()+" : " + rs.getInt(1) +
"\t" + rs.getString(2)+"\t"+rs.getFloat(3));

            rs.relative(2);
            System.out.println("Record "+rs.getRow()+" : " + rs.getInt(1) +
"\t" + rs.getString(2)+"\t"+rs.getFloat(3));


            rs.beforeFirst();
            while(rs.next()){
                System.out.println("Record "+rs.getRow()+" : " + rs.getInt(1)
+ "\t" + rs.getString(2)+"\t"+rs.getFloat(3));
            }
```

```
            if(rs.isAfterLast())
                System.out.println("cursor is at after last");

            rs.afterLast();
            while(rs.previous()){
                System.out.println("Record "+rs.getRow()+" : " + rs.getInt(1)
+ "\t" + rs.getString(2)+"\t"+rs.getFloat(3));
            }

            if(rs.isBeforeFirst())
                System.out.println("cursor is at befoer first");

            rs.last();
            if(rs.isLast())
                System.out.println("cursor is at last");

            rs.first();
            if(rs.isFirst())
                System.out.println("cursor is at first");


            System.out.println("Before deletion");
            while(rs.next()){
                System.out.println("Record "+rs.getRow()+" : " + rs.getInt(1)
+ "\t" + rs.getString(2)+"\t"+rs.getFloat(3));
            }

            rs.absolute(3);
            rs.deleteRow();

            System.out.println("After deletion");
            rs.beforeFirst();
            while(rs.next()){
                System.out.println("Record "+rs.getRow()+" : " + rs.getInt(1)
+ "\t" + rs.getString(2)+"\t"+rs.getFloat(3));
            }


            System.out.println("Before updation");
            while(rs.next()){
                System.out.println("Record "+rs.getRow()+" : " + rs.getInt(1)
+ "\t" + rs.getString(2)+"\t"+rs.getFloat(3));
            }

            rs.absolute(3);
```

```
            rs.updateInt(1,2121);
            rs.updateString(2,"MukeshJain");
            rs.updateFloat(3,30000);
            rs.updateRow();

            System.out.println("After updation");
            rs.beforeFirst();
            while(rs.next()){
                System.out.println("Record "+rs.getRow()+" : " + rs.getInt(1)
+ "\t" + rs.getString(2)+"\t"+rs.getFloat(3));
            }


            System.out.println("Before insertion");
            while(rs.next()){
                System.out.println("Record "+rs.getRow()+" : " + rs.getInt(1)
+ "\t" + rs.getString(2)+"\t"+rs.getFloat(3));
            }

            rs.moveToInsertRow();
            rs.updateInt(1,3131);
            rs.updateString(2,"SureshAgrawal");
            rs.updateFloat(3,30000);
            rs.insertRow();

            System.out.println("After insertion");
            rs.beforeFirst();
            while(rs.next()){
                System.out.println("Record "+rs.getRow()+" : " + rs.getInt(1)
+ "\t" + rs.getString(2)+"\t"+rs.getFloat(3));
            }

        }catch(Exception e){
            System.out.println(e);
        }
    }
}
```