JAVA LAB

Expr 1:

Set up Java Programming development environment by using
i. Command Prompt
ii. Any IDE like Eclipse, Notepad++, JCreater etc

And Test Java Programming development environment by implementing a small program .

Solution:

setup javaprogramming environment

1.Install the JDK from the Oracle website

2.Identify the installation location of the JDK.

- It is usually a sub-folder within this path: C:\Program Files\Java

3.Edit the System Environment Variables.

4.Create a new variable named as JAVA_HOME, and set the installation location of the JDK as the value

5.Edit the PATH variable.

6.click "Okay" on all of the Environment Variables to save the new settings.

7.Open up Command Prompt

8.Verify the Java compiler is recognized

(Type `javac -version`

If you see the version of Java printed out, it worked)

9.Run a Java program

Program:

```
public class main
 {
  public static void main(String[] args)
{
    System.out.println("Hello World");
 }
}
```

Output: `Hello World`

Expr 2:

Implementing the Operations of stack using package .

Program:

```
import java.util.*;
public class ArrayStack < E > {

   public static final int CAPACITY = 1000; // default array capacity
   private int topIndex; // index of the top element in stack
   private E[] data; // generic array used for storage


   public ArrayStack() {
      this(CAPACITY);
   } // constructs stack with default capacity

   public ArrayStack(int capacity) { // constructs stack with given capacity
      topIndex = -1;
      data = (E[]) new Object[capacity]; // safe cast; compiler may give warning
   }

   public int size() {
      return (topIndex + 1);
   }

   public boolean empty() {
      return (topIndex == -1);
   }

   public void push(E e) throws IllegalStateException {
      if (size() == data.length) throw new IllegalStateException("Stack is full");
      data[++topIndex] = e; // increment topIndex before storing new item
   }

   public E peek() throws EmptyStackException {
      if (empty()) throw new EmptyStackException();
      return data[topIndex];
   }

   public E pop() throws EmptyStackException {
      if (empty()) throw new EmptyStackException();
      E answer = data[topIndex];
      data[topIndex] = null; // dereference to help garbage collection
      topIndex--;
      return answer;
   }

   public static void main(String args[]) {
      ArrayStack < Integer > mystack = new ArrayStack<>();
      mystack.push(9); //a
      mystack.push(3); //b
      mystack.push(8); //c
```

```
        System.out.println("Element at the top is :" + mystack.peek()); //d
        System.out.println("Element removed is : " + mystack.pop()); //e
        System.out.println("The size of the stack is : " + mystack.size()); //f
        System.out.println("Element removed is : " + mystack.pop()); //g
        System.out.println("Element at the top is : " + mystack.peek()); //h
        mystack.push(10); //i
        System.out.println("Stack is empty :  " + mystack.empty()); //j
    /*Note: In output charecters of the comments are written to correspond the
    output, they won't be printed.*/
    }
}
```

Output:

```
Element at the top is :8        //d

Element removed is : 8          //e

The size of the stack is : 2    //f

Element removed is : 3          //g

Element at the top is : 9       //h

Stack is empty :  false         //j
```

Expr 3:

Implementing the Operations queue using package.

Program:

```
import java.util.*;
public class ArrayQueue < E > {
    private E[] data; // generic array used for storage
    // constructors
    private int frontIndex;
    private int queueSize;

    public ArrayQueue(int capacity) { // constructs queue with given capacity
        data = (E[]) new Object[capacity]; // safe cast; compiler may give warning
        queueSize = 0; // current number of elements
        frontIndex = 0; // index of the front element


    }
    public ArrayQueue() {
        this(1000);
    } // constructs queue with default capacity

    // methods
    /* Returns the number of elements in the queue. */
    public int size() {
        return queueSize;
    }

    /* Tests whether the queue is empty. */
    public boolean isEmpty() {
        return (queueSize == 0);
    }

    /* Inserts an element at the rear of the queue. */
    public void enqueue(E e) throws IllegalStateException {
        if (queueSize == data.length) throw new IllegalStateException("Queue is full");
        int avail = (frontIndex + queueSize) % data.length; // use modular arithmetic
        data[avail] = e;
        queueSize++;
    }

    /* Returns, but does not remove, the first element of the queue (null if empty). */
    public E first() throws IllegalStateException {
        if (queueSize == data.length) throw new IllegalStateException("Queue is empty");
        return data[frontIndex];
    }

    /* Removes and returns the first element of the queue (null if empty). */
    public E dequeue() throws IllegalStateException {
        if (queueSize == data.length) throw new IllegalStateException("Queue is empty");
        E answer = data[frontIndex];
        data[frontIndex] = null; // dereference to help garbage collection
        frontIndex = (frontIndex + 1) % data.length;
        queueSize--;
        return answer;
    }

    public static void main(String[] args) {
        ArrayQueue queue = new ArrayQueue();
        queue.enqueue(18); //a
        System.out.println("Element at front :  " + queue.first()); //b
        System.out.println("Element removed from front : " + queue.dequeue()); //c
        System.out.println("Queue is Empty : " + queue.isEmpty()); //d
        queue.enqueue(79); //e
        queue.enqueue(90); //f
        System.out.println("Size of the queue : " + queue.size()); //g
        System.out.println("Element removed from front end : " + queue.dequeue());
        //h
    }
}
```

Output:

```
Element at front :  18                //b

Element removed from front : 18       //c

Queue is Empty : true                 //d

Size of the queue : 2                 //g

Element removed from front end : 79   //h
```

Expr 4:

Write a program to implement an object oriented system and multithreaded processes as per needs and specifications.

Program:

```
class RunnableDemo implements Runnable {
   private Thread t;
   private String threadName;

   RunnableDemo( String name) {
      threadName = name;
      System.out.println("Creating " +  threadName );
   }

   public void run() {
      System.out.println("Running " +  threadName );
      try {
         for(int i = 4; i > 0; i--) {
            System.out.println("Thread: " + threadName + ", " + i);
            // Let the thread sleep for a while.
            Thread.sleep(50);
         }
      } catch (InterruptedException e) {
         System.out.println("Thread " +  threadName + "
interrupted.");
      }
      System.out.println("Thread " +  threadName + " exiting.");
   }

   public void start () {
      System.out.println("Starting " +  threadName );
      if (t == null) {
         t = new Thread (this, threadName);
         t.start ();
      }
   }
}

public class TestThread {

   public static void main(String args[]) {
      RunnableDemo R1 = new RunnableDemo( "Thread-1");
      R1.start();

      RunnableDemo R2 = new RunnableDemo( "Thread-2");
      R2.start();
   }
}
```

Output:

Expr 5:

Write a program to implement thread synchronization concept.

Program:

```
class Pyramid
 {
synchronized void draw_pyramid (char ch)
{
for (int i=0; i<10;1+=2)
 {
for (int k-10-1;k>0;k-=2)
  {
System.out.print (" ");
  }
for (int j=0;j<=1;j++)
  {

System.out.print(ch);
  }
System.out.println();
  }
}
}
class A extends Thread
  {
 Pyramid p;
 A (Pyramid p)
  {
this.p=p"
  }
public void run()
   }
}
p.draw_pyramid ('*');

class B extends Thread
```

```
  {
Pyramid p;
 B(Pyramid p)
{

this.p=p;
}
 public void run()
 {

p.draw_pyramid('#');
}
}

class SynchTest
 {
public static void main(String args[])
  {
Pyramid pobj= new Pyramid();
A threadA = new A (pobj);
B threadB = new B (pobj);
threadA.start(); threadB.start();
}
}
```

Output:

Expr 6:

Write a program to implement employee information in a file and perform the operations on it .

Program:

```
public class GFG {

      static String Employee_name;
      static float Employee_salary;

      static void set(String n, float p) {
            Employee_name = n;
            Employee_salary = p;
      }

      static void get() {
            System.out.println("Employee name is: " +Employee_name );
            System.out.println("Employee CTC is: " +
Employee_salary);
      }

      public static void main(String args[]) {
            GFG.set("Rathod Avinash", 10000.0f);
            GFG.get();
      }
}
```

Output:

```
C:\Users\student\Desktop\Aadi>java GFG
Employee name is: Rathod Avinash
Employee CTC is: 10000.0
```

Expr 7:

Working with shape motion by applet programming.
(demonstrate Traffic Signal)

```
import java.io.*;
import java.lang.*;
import java.awt.*;
import java.awt.event.*;
import java.applet.*;

public class Thread39 extends Applet implements ActionListener,Runnable
{
            Thread t;
            Button b1,b2;
            boolean red,green,yellow;



            public void init()
            {
                        t=new Thread(this);
            }

            public void start()
            {
                  b1=new Button("START SIGNAL");
                  b2=new Button("STOP SIGNAL");

red=true;

                  add(b1);
                  add(b2);


                  setSize(400,400);
                  setVisible(true);

                  b1.addActionListener(this);
                  b2.addActionListener(this);
            }

            public void actionPerformed(ActionEvent ae)
            {
                      Object ob=ae.getSource();
```

```
                if(ae.getSource()==b1)
                {
                         t.start();
                }
                else
                {
                      if(ae.getSource()==b2)
                      {
                            t.stop();
                      }
                }
        }


        public void run()
        {
                try
                {
                      while(true)
                      {
                            if(red==true)
                            {
                                  red=false;
                                  yellow=true;
                                  green=false;
                            }
                            else
                            {
                                  if(yellow==true)
                                  {
                                        red=false;
                                        yellow=false;
                                        green=true;
                                  }
                                  else
                                  {
                                        if(green==true)
                                     {
                                             red=true;
                                             yellow=false;
                                             green=false;
                                        }
                                  }
                            }
                            repaint();
                          t.sleep(700);
                      }

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

        }

        public void paint(Graphics g)
        {
              if(red==true)
              {
```

```
                        g.setColor(Color.RED);
                        g.fillOval(100,100,70,70);

                  }
                  else
                  {
                        if(yellow==true)
                        {
                                  g.setColor(Color.YELLOW);
                                g.fillOval(100,200,70,70);
                        }
                        else
                        {
                                  if(green==true)
                                  {
                                       g.setColor(Color.GREEN);
                                       g.fillOval(100,300,70,70);
}
                        }
                  }
            }
}
```

Output:

<u>Expr 8 :</u>

Write a program to design Registration process form using Applet and AWT components.

Program:

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

public class RegistrationForm extends JFrame implements ActionListener {
private JLabel nameLabel, emailLabel, passwordLabel;
private JTextField nameField, emailField;
private JPasswordField passwordField;
private JButton registerButton;

public RegistrationForm() {
setTitle("Registration Form");
setDefaultCloseOperation(JFrame.EXIT_ON_CLOSE);
setSize(300, 200);
setLocationRelativeTo(null);
setLayout(new GridLayout(4, 2));

nameLabel = new JLabel("Name:");
add(nameLabel);
nameField = new JTextField();
add(nameField);

emailLabel = new JLabel("Email:");
add(emailLabel);
emailField = new JTextField();
add(emailField);
passwordLabel = new JLabel("Password:");
add(passwordLabel);
passwordField = new JPasswordField();
add(passwordField);

registerButton = new JButton("Register");
registerButton.addActionListener(this);
add(registerButton);

setVisible(true);
}

public void actionPerformed(ActionEvent e) {
if (e.getSource() == registerButton) {
String name = nameField.getText();
String email = emailField.getText();
String password = new String(passwordField.getPassword());

// Perform registration logic here

JOptionPane.showMessageDialog(this, "Registration Successful!");
}
}

public static void main(String[] args) {
SwingUtilities.invokeLater(new Runnable() {
public void run() {
new RegistrationForm();
}
});
}
}
```

Output:

<u>Expr 9:</u>

Write a Servlet code to demonstrate GET and POST methods with suitable example.

Program:

```
import java.io.IOException;
import java.io.PrintWriter;
import javax.servlet.ServletException;
import javax.servlet.http.HttpServlet;
import javax.servlet.http.HttpServletRequest;
import javax.servlet.http.HttpServletResponse;

public class DemoServlet extends HttpServlet {
protected void doGet(HttpServletRequest request, HttpServletResponse
response)
throws ServletException, IOException {
response.setContentType("text/html");
PrintWriter out = response.getWriter();
out.println("<html>");
out.println("<head>");
out.println("<title>GET Method Example</title>");
out.println("</head>");
out.println("<body>");
out.println("<h1>GET Method Example</h1>");
out.println("<p>Enter your name:</p>");
out.println("<form method=\"post\">");
out.println("<input type=\"text\" name=\"name\">");
out.println("<input type=\"submit\" value=\"Submit\">");
out.println("</form>");
out.println("</body>");
out.println("</html>");
}

protected void doPost(HttpServletRequest request, HttpServletResponse
response)
throws ServletException, IOException {
response.setContentType("text/html");
PrintWriter out = response.getWriter();

String name = request.getParameter("name");

out.println("<html>");
out.println("<head>");
out.println("<title>POST Method Example</title>");
out.println("</head>");
out.println("<body>");
out.println("<h1>POST Method Example</h1>");
out.println("<p>Hello, " + name + "!</p>");
out.println("</body>");
out.println("</html>");
}
}
```

Expr 10 :

write a program to chat between client and server. (swing use)

Program:

```
import java.io.*;
import java.lang.*;
import java.net.*;

class slip1client
{
public static void main(String args[])
{
String str1;
String str2;
try
{
Socket s=new Socket("localhost",8983);
InputStream is=s.getInputStream();
DataInputStream dis=new DataInputStream(is);

OutputStream os=s.getOutputStream();
DataOutputStream dos=new DataOutputStream(os);

BufferedReader br=new BufferedReader(new InputStreamReader(System.in));

System.out.println("\n Hello TYBCA iam Cient");
System.out.println("\n Enter Your Message:");

str1=br.readLine();

dos.writeUTF(str1);

System.out.println("\nData Sent to Server");

str2=dis.readUTF();

System.out.println("\nThe Server says:"+str2);
}
catch(Exception e)
{
System.out.println(e);
}
}
}
```

<u>Expr 11 :</u>

write a program to connect to any database and to execute the sql query operation using GUI interface?

```
( design one stand alone application where user will see the given options)
1.Display all record
2.Display  record with rollno
3.Display record with name
4.Insert record
5.Delete record with rollno
6.Delete record with name
7.Update record with rollno
8.Update record with name
Accept the choice from user & performs the  operations according to the
choice given.
```

Program:

```
import java.io.*;
import java.lang.*;
import java.sql.*;

class abc
{
      Connection con;
      Statement stmt;
      ResultSet rs;
      String sql;
```

```
	int ch = 0;
	int rn = 0;
	String nm;
	int mks = 0;
	int i = 0;

	public abc()
	{
		try
		{
			Class.forName("net.ucanaccess.jdbc.UcanaccessDriver");

	con=DriverManager.getConnection("jdbc:ucanaccess://tybcadb.accdb");
		}
		catch(Exception e)
		{
			System.out.println(e);
		}
	}

	public void display()
	{
		try
		{
			BufferedReader br=new BufferedReader(new
InputStreamReader(System.in));

			System.out.println("MENU: ");
			System.out.println("1. Display all records ");
			System.out.println("2. Display records with rollno");
			System.out.println("3. Display records with name");
			System.out.println("4. Insert records");
			System.out.println("5. Delete records with rollno");
			System.out.println("6. Delete records with name");
			System.out.println("7. Update records with rollno");
			System.out.println("8. Update records with name");

			System.out.println("Enter your choice :");
			ch=Integer.parseInt(br.readLine());

			switch(ch)
			{
				case 1:
					sql="select * from tybcatable";
					stmt=con.createStatement();
					rs=stmt.executeQuery(sql);

					if(rs.next())
					{
						System.out.println("\n ROLLNO \t NAME
\t MARKS");

						do
						{


	System.out.println("\t"+rs.getInt("rollno")+"\t"+rs.getString("name")
+"\t"+rs.getInt("marks"));
						}while(rs.next());
					}
```

```
                              else
                              {
                                    System.out.println("No data found");
                              }
                              break;

                        case 2:

                              System.out.println("Enter RollNo :");
                              rn=Integer.parseInt(br.readLine());

                              sql="select * from tybcatable where 
rollno="+rn;
                              stmt=con.createStatement();
                              rs=stmt.executeQuery(sql);

                              if(rs.next())
                              {
                                    System.out.println("\n ROLLNO \t NAME 
\t MARKS");

                                    do
                                    {


      System.out.println("\t"+rs.getInt("rollno")+"\t"+rs.getString("name")
+"\t"+rs.getInt("marks"));
                                    }while(rs.next());
                              }
                              else
                              {
                                    System.out.println("No data found");
                              }
                              break;

                        case 3:

                              System.out.println("Enter Name :");
                              nm=(br.readLine());

                              sql="select * from tybcatable where 
name='"+nm+"'";
                              stmt=con.createStatement();
                              rs=stmt.executeQuery(sql);

                              if(rs.next())
                              {
                                    System.out.println("\n ROLLNO \t NAME 
\t MARKS");

                                    do
                                    {


      System.out.println("\t"+rs.getInt("rollno")+"\t"+rs.getString("name")
+"\t"+rs.getInt("marks"));
                                    }while(rs.next());
                              }
                              else
                              {
                                    System.out.println("No data found");
                              }
                              break;

                        case 4:

                              System.out.println("Enter RollNo :");
                              rn=Integer.parseInt(br.readLine());

                              System.out.println("Enter Name :");
                              nm=(br.readLine());

                              System.out.println("Marks :");
                              mks=Integer.parseInt(br.readLine());

                              sql="insert into tybcatable(rollno, name,
marks) values("+rn+",'"+nm+"',"+mks+")";
                              stmt=con.createStatement();
                              i=stmt.executeUpdate(sql);

                              if(i>0)
                              {
                                    System.out.println("\n Data Inserted");
                              }
                              else
                              {
                                    System.out.println("\n Data not
Inserted");
                              }
                              break;

                        case 5:

                              System.out.println("Enter RollNo :");
                              rn=Integer.parseInt(br.readLine());

                              sql="delete * from tybcatable where
rollno="+rn;
                              stmt=con.createStatement();
                              i=stmt.executeUpdate(sql);

                              if(i>0)
                              {
                                    System.out.println("\n Data deleted
properly");
                              }
                              else
                              {
                                    System.out.println("\n Data not deleted
properly");
                              }
                              break;

                        case 6:

                              System.out.println("Enter Name :");
                              nm=(br.readLine());

                              sql="delete * from tybcatable where
name='"+nm+"'";
                              stmt=con.createStatement();
                              i=stmt.executeUpdate(sql);
                              if(i>0)
                              {
                                    System.out.println("\n Data deleted 
properly");
                              }
                              else
                              {
                                    System.out.println("\n Data not deleted 
properly");
                              }
                              break;

                        case 7:

                              System.out.println("Enter RollNo :");
                              rn=Integer.parseInt(br.readLine());

                              System.out.println("Enter Name :");
                              nm=(br.readLine());

                              System.out.println("Marks :");
                              mks=Integer.parseInt(br.readLine());

                              sql="update tybcatable set name='"+nm+"', 
marks="+mks+" where rollno="+rn;
                              stmt=con.createStatement();
                              i=stmt.executeUpdate(sql);

                              if(i>0)
                              {
                                    System.out.println("\n Data Updated 
properly");
                              }
                              else
                              {
                                    System.out.println("\n Data not Updated 
properly");
                              }
                              break;

                        case 8:

                              System.out.println("Enter Name :");
                              nm=(br.readLine());

                              System.out.println("Enter RollNo :");
                              rn=Integer.parseInt(br.readLine());

                              System.out.println("Marks :");
                              mks=Integer.parseInt(br.readLine());

                              sql="update tybcatable set rollno="+rn+", 
marks="+mks+" where name='"+nm+"'";
                              stmt=con.createStatement();
                              i=stmt.executeUpdate(sql);

                              if(i>0)
                              {
                                    System.out.println("\n Data Updated 
properly");
```

```
                              }
                              else
                              {
                                    System.out.println("\n Data not Updated
properly");
                              }
                              break;

                        default:
                                          System.out.println("Invalid
Choice");
                                          break;

                  }

            }
            catch(Exception e)
            {

            }
      }
}

class jdbc53
{
      public static void main(String args[])
      {
            abc a = new abc();
            a.display();
      }
}
```

Expr 12 :

write a program to demonstrate socket programming e.g send hello world to server from client.

```
*server.java
import java.io.*;
import java.net.*;

public class Server {
public static void main(String[] args) {
try {
ServerSocket serverSocket = new ServerSocket(8989);
System.out.println("Server started. Waiting for client...");

Socket socket = serverSocket.accept();
System.out.println("Client connected.");

BufferedReader reader = new BufferedReader(new
InputStreamReader(socket.getInputStream()));
String message = reader.readLine();
System.out.println("Received from client: " + message);

serverSocket.close();
} catch (IOException e) {
e.printStackTrace();
}
}
}

*client.java
import java.io.*;
import java.net.*;

public class Client {
public static void main(String[] args) {
try {
Socket socket = new Socket("localhost", 8989);

OutputStream outputStream = socket.getOutputStream();
PrintWriter writer = new PrintWriter(outputStream, true);
writer.println("Hello, World!");

socket.close();
} catch (IOException e) {
e.printStackTrace();
}
}
```

Expr 13:

write a program to demonstrate the use of jsp?

(Write a jsp example to display reverse of the no. on the web.)

Program:

```
<html>
<body>
<form method=post
action="http://localhost:8080/examples/jsp/Semester6_solved_Practical_slips
_jsp/slip2B.jsp">
Enter the Number To Reverse:<input type=text name="text1"></input>
<input type=submit value=Check></input>
</form>
</body>
</html>


<%@ page language="java" import="java.io.,java.lang."%>
<%!
int rem=0;
int rev=0;
int no=0;
%>

<%
no=Integer.parseInt(request.getParameter("text1"));

while(no!=0)
{
rem=no%10;
rev=(rev*10)+rem;
no=no/10;
}
```

```
out.println("The Reverse Numbers are Displayed Below:"+rev);

%>
```

Expr 14:

Write a program to connect to any database and to execute the SQL query operation on command prompt.

```
// Connecting to the Database
import java.sql.*;

public class connect
{
	public static void main(String args[])
	{
		try
		{
			Class.forName("oracle.jdbc.driver.OracleDriver");

			// Establishing Connection
			Connection con = DriverManager.getConnection(
			"jdbc:oracle:thin:@localhost:1521:orcl", "login1", "pwd1");

			if (con != null)
				System.out.println("Connected");
			else
				System.out.println("Not Connected");

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

// inserting to the Database

```
import java.sql.*;

public class insert1
{
	public static void main(String args[])
	{
		String id = "id1";
		String pwd = "pwd1";
		String fullname = "geeks for geeks";
		String email = "geeks@geeks.org";

		try
		{
			Class.forName("oracle.jdbc.driver.OracleDriver");
			Connection con = DriverManager.getConnection("
			jdbc:oracle:thin:@localhost:1521:orcl", "login1", "pwd1");
			Statement stmt = con.createStatement();

			// Inserting data in database
			String q1 = "insert into userid values('" +id+ "', '" +pwd+
									"', '" +fullname+ "', '" +email+
"')";
			int x = stmt.executeUpdate(q1);
			if (x > 0)
				System.out.println("Successfully Inserted");
			else
				System.out.println("Insert Failed");

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

// updating the Database

```
import java.sql.*;

public class update1
{
	public static void main(String args[])
	{
		String id = "id1";
		String pwd = "pwd1";
		String newPwd = "newpwd";
		try
		{
			Class.forName("oracle.jdbc.driver.OracleDriver");
			Connection con = DriverManager.getConnection("
			jdbc:oracle:thin:@localhost:1521:orcl", "login1", "pwd1");
			Statement stmt = con.createStatement();

			// Updating database
			String q1 = "UPDATE userid set pwd = '" + newPwd +
					"' WHERE id = '" +id+ "' AND pwd = '" + pwd + "'";
			int x = stmt.executeUpdate(q1);
```

```
                if (x > 0)
                        System.out.println("Password Successfully Updated");

                else
                        System.out.println("ERROR OCCURRED :(");

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

// deleting from Database

```
import java.sql.*;

public class delete
{
        public static void main(String args[])
        {
            String id = "id2";
            String pwd = "pwd2";
            try
            {
                Class.forName("oracle.jdbc.driver.OracleDriver");
                Connection con = DriverManager.getConnection("
                jdbc:oracle:thin:@localhost:1521:orcl", "login1", "pwd1");
                Statement stmt = con.createStatement();

                // Deleting from database
                String q1 = "DELETE from userid WHERE id = '" + id +
                                "' AND pwd = '" + pwd + "'";

                int x = stmt.executeUpdate(q1);

                if (x > 0)
                        System.out.println("One User Successfully Deleted");

                else
                        System.out.println("ERROR OCCURRED :(");

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

```
			}
		}
}
```

// selecting from Database

```
import java.sql.*;

public class select
{
		public static void main(String args[])
		{
			String id = "id1";
			String pwd = "pwd1";
			try
			{
				Class.forName("oracle.jdbc.driver.OracleDriver");
				Connection con = DriverManager.getConnection("
						jdbc:oracle:thin:@localhost:1521:orcl", "login1",
"pwd1");
				Statement stmt = con.createStatement();

				// SELECT query
				String q1 = "select * from userid WHERE id = '" + id +
										"' AND pwd = '" + pwd +
"'";
				ResultSet rs = stmt.executeQuery(q1);
				if (rs.next())
				{
					System.out.println("User-Id : " + rs.getString(1));
					System.out.println("Full Name :" + rs.getString(3));
					System.out.println("E-mail :" + rs.getString(4));
				}
				else
				{
					System.out.println("No such user id is already registered");
				}
				con.close();
			}
			catch(Exception e)
			{
				System.out.println(e);
			}
		}
}
```