LAB ASSIGNMENT

1) Question: Write a Java program to demonstrate the use of different data types and control structures.

Answer:

```
public class DataTypesAndControls {
  public static void main(String[] args) {
    int a = 10;
    double b = 7.5;
    char c = 'Z';
    boolean flag = true;
    System.out.println("int a = " + a);
    System.out.println("double b = " + b);
    System.out.println("char c = " + c);
    System.out.println("boolean flag = " + flag);
    if (a > b) {
      System.out.println("a is greater than b");
    } else {
      System.out.println("a is not greater than b");
    }
    int day = 3;
    switch (day) {
      case 1: System.out.println("Monday"); break;
      case 2: System.out.println("Tuesday"); break;
      case 3: System.out.println("Wednesday"); break;
      default: System.out.println("Other day");
    }
    System.out.print("for: ");
    for (int i=1; i<=5; i++) System.out.print(i+" ");
    System.out.println();
```

```
        System.out.print("while: ");
        int j=1;
        while (j<=5) { System.out.print(j+" "); j++; }
        System.out.println();
    }
}
```

Output:

```
int a = 10
double b = 7.5
char c = Z
boolean flag = true
a is greater than b
Wednesday
for: 1 2 3 4 5
while: 1 2 3 4 5
```

2) Question: Factorial using recursion and iteration.

Answer:

```
import java.math.BigInteger;
public class FactorialDemo {
    public static BigInteger factorialIter(int n) {
        BigInteger res = BigInteger.ONE;
        for (int i=2;i<=n;i++) res = res.multiply(BigInteger.valueOf(i));
        return res;
    }
    public static BigInteger factorialRec(int n) {
        if (n<=1) return BigInteger.ONE;
        return BigInteger.valueOf(n).multiply(factorialRec(n-1));
    }
    public static void main(String[] args) {
```

```
    int n = 10;
    System.out.println("Iteration: " + factorialIter(n));
    System.out.println("Recursion: " + factorialRec(n));
  }
}
```

Output:

```
Iteration: 3628800
Recursion: 3628800
```

3) Question: Second largest element in array.

Answer:

```
public class SecondLargest {
  public static int secondLargest(int[] arr) {
    if (arr.length < 2) throw new IllegalArgumentException("Need at least two elements");
    int largest = Integer.MIN_VALUE;
    int second = Integer.MIN_VALUE;
    for (int v : arr) {
      if (v > largest) { second = largest; largest = v; }
      else if (v > second && v != largest) second = v;
    }
    return second;
  }
  public static void main(String[] args) {
    int[] a = {3, 5, 2, 9, 9, 7};
    System.out.println("Second largest: " + secondLargest(a));
  }
}
```

Output:

```
Second largest: 7
```

4) Question: Palindrome check.

Answer:

```
public class PalindromeCheck {
  public static boolean isPalindrome(String s) {
     String clean = s.replaceAll("\\s+","" ).toLowerCase();
     int i=0, j=clean.length()-1;
     while (i<j) if (clean.charAt(i++) != clean.charAt(j--)) return false;
     return true;
  }
  public static void main(String[] args) {
     String s1 = "Madam";
     String s2 = "Hello";
     System.out.println(s1 + " -> " + isPalindrome(s1));
     System.out.println(s2 + " -> " + isPalindrome(s2));
  }
}
```

Output:

```
Madam -> true
Hello -> false
```

5) Question: Multiple interfaces.

Answer:

```
interface Walkable { void walk(); }
interface Talkable { void talk(); }
public class Person implements Walkable, Talkable {
  public void walk() { System.out.println("Walking..."); }
  public void talk() { System.out.println("Talking..."); }
  public static void main(String[] args) {
     Person p = new Person();
     p.walk();
     p.talk();
```

```
  }
}
```

Output:

```
Walking...
Talking...
```

6) Question: Even and odd threads.

Answer:

```
public class EvenOddThreads {
  public static void main(String[] args) {
    Object lock = new Object();
    Thread odd = new Thread(() -> {
      for (int i=1;i<=100;i+=2) {
        synchronized(lock) {
          System.out.print(i+" ");
          lock.notify();
        }
      }
    });
          try { if (i<99) lock.wait(); } catch (InterruptedException e) {}
    Thread even = new Thread(() -> {
      for (int i=2;i<=100;i+=2) {
        synchronized(lock) {
          System.out.print(i+" ");
          lock.notify();
          try { if (i<100) lock.wait(); } catch (InterruptedException e) {}
        }
      }
    });
    odd.start();
```

```
        even.start();
        try { odd.join(); even.join(); } catch (InterruptedException e) {}
    }
}
```

Output:

```
1 2 3 4 5 ... 99 100
```

7) Question: Insert + update MySQL.

Answer:

```
import java.sql.*;
public class MySQLInsertUpdate {
    public static void main(String[] args) throws Exception {
        String url = "jdbc:mysql://localhost:3306/testdb";
        String user = "root";
        String pass = "rootpassword";
        try (Connection conn = DriverManager.getConnection(url,user,pass)) {
            try (Statement st = conn.createStatement()) {
                st.executeUpdate("CREATE TABLE IF NOT EXISTS students(id INT PRIMARY KEY
AUTO_INCREMENT, name VARCHAR(100), age INT)");
            }
            String insert = "INSERT INTO students(name, age) VALUES(?,?)";
            try (PreparedStatement ps = conn.prepareStatement(insert)) {
                ps.setString(1, "Alice");
                ps.setInt(2, 21);
                ps.executeUpdate();
            }
            String update = "UPDATE students SET age = ? WHERE name = ?";
            try (PreparedStatement ps = conn.prepareStatement(update)) {
                ps.setInt(1, 22);
                ps.setString(2, "Alice");
                System.out.println("Rows updated: " + ps.executeUpdate());
```

```
        }
      }
    }
}
```

Output:

Rows updated: 1

.8) Applet: input two numbers, show sum

```
import java.applet.Applet;
import java.awt.*;
import java.awt.event.*;
public class SumApplet extends Applet implements ActionListener {
  TextField t1, t2;
  Label res;
  public void init() {
    t1 = new TextField(5); t2 = new TextField(5);
    Button b = new Button("Add"); b.addActionListener(this);
    res = new Label("Result: ");
    add(new Label("A:")); add(t1);
    add(new Label("B:")); add(t2);
    add(b); add(res);
  }
  public void actionPerformed(ActionEvent e) {
    int a = Integer.parseInt(t1.getText());
    int b = Integer.parseInt(t2.getText());
    res.setText("Result: "+(a+b));
  }
}
```

Output:

Applet UI shows fields A, B and “Result: 7” after entering 3 and 4.

9) AWT GUI change background color

```
import java.awt.*;
import java.awt.event.*;
public class ColorButtonAWT extends Frame {
  public ColorButtonAWT() {
    super("AWT Color Button");
    setSize(300,200);
    Button b = new Button("Change Color");
    b.addActionListener(e -> setBackground(new Color((int)(Math.random()*0x1000000))));
    add(b, BorderLayout.SOUTH);
    setVisible(true);
    addWindowListener(new WindowAdapter(){ public void windowClosing(WindowEvent
e){ dispose(); }});
  }
  public static void main(String[] args){ new ColorButtonAWT(); }
}
```

Output:

Window opens. Background changes on each click.

10) BorderLayout + FlowLayout

```
import java.awt.*;
public class LayoutDemo extends Frame {
  public LayoutDemo() {
    setTitle("Border + Flow Layout");
    setSize(400,200);
    Panel top = new Panel(new FlowLayout());
    top.add(new Button("One"));
    top.add(new Button("Two"));
    add(top, BorderLayout.NORTH);
    Panel center = new Panel(new FlowLayout());
    center.add(new Label("Center area"));
```

```
        add(center, BorderLayout.CENTER);
        setVisible(true);
    }
    public static void main(String[] args){ new LayoutDemo(); }
}
```

Output:

Window shows buttons at top and a label in center.

11) Count objects

```
public class ObjectCounter {
    private static int count = 0;
    public ObjectCounter(){ count++; }
    public static int getCount(){ return count; }
    public static void main(String[] args){
        new ObjectCounter(); new ObjectCounter(); new ObjectCounter();
        System.out.println("Objects created: " + ObjectCounter.getCount());
    }
}
```

Output:

Objects created: 3

12) UDP client/server

```
// SERVER
import java.net.*;
public class UDPServer {
    public static void main(String[] args) throws Exception {
        DatagramSocket server = new DatagramSocket(9876);
        byte[] receive = new byte[1024];
        System.out.println("Server listening...");
        DatagramPacket packet = new DatagramPacket(receive, receive.length);
        server.receive(packet);
```

```
        String msg = new String(packet.getData(),0,packet.getLength());
        System.out.println("Received: " + msg);
        String reply = "Echo: " + msg;
        DatagramPacket sp = new DatagramPacket(reply.getBytes(), reply.length(),
packet.getAddress(), packet.getPort());
        server.send(sp);
        server.close();
    }
}
// CLIENT
import java.net.*;
public class UDPClient {
    public static void main(String[] args) throws Exception {
        DatagramSocket c = new DatagramSocket();
        byte[] buf = "Hello UDP".getBytes();
        DatagramPacket p = new DatagramPacket(buf, buf.length,
InetAddress.getByName("localhost"), 9876);
        c.send(p);
        byte[] recv = new byte[1024];
        DatagramPacket r = new DatagramPacket(recv, recv.length);
        c.receive(r);
        System.out.println("Server replied: " + new String(r.getData(),0,r.getLength()));
        c.close();
    }
}
```

Output (server):

Received: Hello UDP

Output (client):

Server replied: Echo: Hello UDP

13) Event-handling label changer

```
import java.awt.*;
import java.awt.event.*;
public class LabelChange extends Frame {
  public LabelChange() {
    setSize(300,150);
    Label lbl = new Label("Original text");
    Button b = new Button("Change");
    b.addActionListener(e -> lbl.setText("Text changed!"));
    add(lbl, BorderLayout.CENTER);
    add(b, BorderLayout.SOUTH);
    setVisible(true);
    addWindowListener(new WindowAdapter(){ public void windowClosing(WindowEvent
e){ dispose(); }});
  }
  public static void main(String[] args){ new LabelChange(); }
}
```

Output:

Label updates to “Text changed!” when clicked.

14) MySQL retrieve data

```
import java.sql.*;
public class MySQLRetrieve {
  public static void main(String[] args) throws Exception {
    String url = "jdbc:mysql://localhost:3306/testdb";
    String user = "root";
    String pass = "rootpassword";
    try (Connection conn = DriverManager.getConnection(url,user,pass)) {
      Statement st = conn.createStatement();
      ResultSet rs = st.executeQuery("SELECT id, name, age FROM students");
      while (rs.next()) {
```

```
            System.out.println(rs.getInt("id") + ": " + rs.getString("name") + " (" +
rs.getInt("age") + ")");
          }
       }
   }
}
```

Output:

1: Alice (22)

15) Custom JSP Tag

Tag Handler (ShowTag.java):

```
package mytags;
import javax.servlet.jsp.tagext.*;
import javax.servlet.jsp.*;
import java.io.*;
public class ShowTag extends SimpleTagSupport {
   private String name;
   private int age;
   public void setName(String n){ name = n; }
   public void setAge(int a){ age = a; }
   public void doTag() throws JspException, IOException {
      JspWriter out = getJspContext().getOut();
      out.println("<div>");
      out.println("<h2>Welcome, " + name + "</h2>");
      out.println("<p>Your age is: " + age + "</p>");
      out.println("</div>");
   }
}
```

TLD entry + JSP usage:

```
<u:show name="Prince" age="21"/>
```

Output:

```
<div>
<h2>Welcome, Prince</h2>
<p>Your age is: 21</p>
</div>
```

16) Servlet request parameters

HTML Form:

```
<form action="/show" method="post">
<input name="name"/>
<input name="age"/>
<button type="submit">Send</button>
</form>
```

Servlet (ShowServlet.java):

```
import java.io.*;
import javax.servlet.*;
import javax.servlet.http.*;
public class ShowServlet extends HttpServlet {
   protected void doPost(HttpServletRequest req, HttpServletResponse res) throws ServletException, IOException {
      String name = req.getParameter("name");
      String age = req.getParameter("age");
      res.setContentType("text/html");
      PrintWriter out = res.getWriter();
      out.println("<html><body>");
      out.println("<h1>Hello, " + name + "</h1>");
      out.println("<p>Age: " + age + "</p>");
      out.println("</body></html>");
   }
}
```

Output:

```
<h1>Hello, Prince</h1>
```

Age: 21

17) HTML table with formatting

```
<!doctype html>
<html>
<body>
<table border="1">
<tr>
<th rowspan="2">Name</th>
<th colspan="2">Marks</th>
</tr>
<tr><th>Math</th><th>Science</th></tr>
<tr><td>Prince</td><td>85</td><td>90</td></tr>
</table>
</body></html>
```

Output:

Table renders with rowspan and colspan correctly.

18) Image map

```
<img src="world.png" usemap="#m">
<map name="m">
<area shape="rect" coords="0,0,100,100" href="asia.html">
<area shape="circle" coords="200,150,50" href="europe.html">
</map>
```

Output:

Clicking each area opens different pages.

19) Session tracking servlet

```
import java.io.*;
import javax.servlet.*;
import javax.servlet.http.*;
public class VisitServlet extends HttpServlet {
```

```
   protected void doGet(HttpServletRequest req, HttpServletResponse res) throws
ServletException, IOException {
      HttpSession session = req.getSession(true);
      Integer count = (Integer) session.getAttribute("visits");
      if (count == null) count = 0;
      count++;
      session.setAttribute("visits", count);
      res.setContentType("text/html");
      PrintWriter out = res.getWriter();
      out.println("<p>You've visited " + count + " times.</p>");
   }
}
```

Output:

You've visited 1 times. (increments every refresh)

20) JSP dynamic greeting

JSP:

```
<%
String name = request.getParameter("name");
String age = request.getParameter("age");
if (name == null) name = "Guest";
if (age == null) age = "unknown";
%>
<h1>Welcome, <%= name %>!</h1>
<p>Your age is <%= age %>.</p>
```

Output:

Welcome, Prince!

Your age is 21