# Overloading.java:

import java.io.\*; **class function {**

**private int a, b, d; float c;**

**function() {**

**a = 50;**

**b = 50;**

**c = a + b;**

**}**

**function(int x, int y, int z) {**

**a = x;**

**b = y; d = z;**

**c = a + b + d;**

**}**

**function(float x, float y) { float p, q;**

**p = x;**

**q = y;**

**c = p + q;**

**}**

**public void display() { System.out.println("Total = " + c);**

**}**

**}**

**public class Overloading {**

**public static void main(String[] args) { function obj1 = new function(); obj1.display();**

**function obj2 = new function(25, 25, 25); obj2.display();**

**function obj3 = new function(4.7f, 4.9f); obj3.display();**

**}**

**}**

# Inheritance.java:

**class student { int rollno;**

**void getnumber(int n) { rollno = n;**

**}**

**void putnumber() { System.out.println("Roll no is" + rollno);**

**}**

**}**

**class Test extends student { float part1, part2;**

**void getmarks(float m1,float m2){ part1=m1;**

**part2=m2;**

**}**

**void putmarks(){**

**System.out.println("Marks obtained is part1="+part1+" part2="+part2);**

**}**

**}**

**class result extends Test{ float total;**

**void display() {**

**total = part1 + part2 ; putnumber(); putmarks();**

**System.out.println("Total marks is" + total);**

**}**

**}**

**class Hybrid {**

**public static void main(String args[]){ result student1=new result(); student1.getnumber(1234); student1.getmarks(27.5f, 33.0f); student1.display();**

**}**

**}**

# Exception.java:

**public class Exception {**

**public static void main(String[] args) { int data;**

**try {**

**data = 100 / 0;**

**} catch (ArithmeticException e) { System.out.println(e);**

**}**

**System.out.println("Rest of the code...");**

**}**

**}**

**Package.java:**

**import java.io.\*;**

**public class Expenditure**

**{**

**public static int sum;**

**int med,travel,phone,grosary; public int calc()**

**{**

**med=500; travel=200; phone=300; grosary=1500;**

**sum=med+travel+phone+grosary; return sum;**

**}**

**public void disp()**

**{**

**System.out.println("\n\t EXPENDITURE"); System.out.println("Medical:"+med); System.out.println("Travel:"+travel); System.out.println("Phone:"+phone); System.out.println("Grosary:"+grosary);**

**}**

**}**

**import java.io.\*; public class Income {**

**int sal,rent,interset;//data member public int get()**

**{**

**sal=5000; rent=1200; interset=400;**

**return sal+rent+interset;**

**}**

**public void disp()**

**{**

**System.out.println("\n\t INCOME"); System.out.println("salary"+sal); System.out.println("Rent"+rent); System.out.println("Interest"+interset);**

**}**

**}**

**import java.io.\*; import employee.\*; public class Pack**

**{**

**public static void main (String[] args)throws IOException**

**{**

**int x,y;**

**Income i=new Income(); x=i.get();**

**Expenditure e=new Expenditure(); y=e.calc();**

**i.disp();**

**e.disp();**

**System.out.println("\n\t INCOME OF THE FFAMILY"); System.out.println("INCOME:"+x); System.out.println("EXPENSE:"+y); System.out.println("SAVINGS"+(x-y));**

**}**

**}**

# Interface.java:

**class student { int rollno;**

**void getnumber(int n) { rollno = n;**

**}**

**void putnumber() { System.out.println("Roll no is" + rollno);**

**}**

**}**

**class Test extends student { float part1, part2;**

**void getmarks(float m1,float m2){ part1=m1;**

**part2=m2;**

**}**

**void putmarks(){**

**System.out.println("Marks obtained is part1="+part1+" part2="+part2);**

**}**

**}**

**interface sports {**

**float sportwt = 6.0f; void putwt();**

**}**

**class result extends Test implements sports { float total;**

**public void putwt() { System.out.println("sports wt is" + sportwt);**

**}**

**void display() {**

**total = part1 + part2 + sportwt; putnumber();**

**putmarks(); putwt();**

**System.out.println("Total marks is" + total);**

**}**

**}**

**class Hybrid {**

**public static void main(String args[]){ result student1=new result(); student1.getnumber(1234); student1.getmarks(27.5f, 33.0f); student1.display();**

**}**

**}**

**Employee gui:**

**import java.awt.\*; import java.awt.event.\*;**

**public class Employee extends Frame implements ActionListener { Label title, nameLbl, idLbl, deptLbl, salaryLbl;**

**TextField nameTxt, idTxt, deptTxt, salaryTxt; Button addBtn, updateBtn, deleteBtn, clearBtn; TextArea displayArea;**

**Panel inputPanel, buttonPanel, displayPanel;**

**public Employee() {**

**setTitle("Employee Management System"); setSize(500, 500); setLocationRelativeTo(null); setLayout(new BorderLayout(10, 10));**

**title = new Label("Employee Management System", Label.CENTER); title.setFont(new Font("Arial", Font.BOLD, 20));**

**add(title, BorderLayout.NORTH);**

**inputPanel = new Panel(new GridLayout(4, 2, 10, 10)); nameLbl = new Label("Name:");**

**nameTxt = new TextField(); inputPanel.add(nameLbl); inputPanel.add(nameTxt);**

**idLbl = new Label("ID:"); idTxt = new TextField(); inputPanel.add(idLbl); inputPanel.add(idTxt);**

**deptLbl = new Label("Department:"); deptTxt = new TextField(); inputPanel.add(deptLbl); inputPanel.add(deptTxt);**

**salaryLbl = new Label("Salary:"); salaryTxt = new TextField(); inputPanel.add(salaryLbl); inputPanel.add(salaryTxt); add(inputPanel, BorderLayout.CENTER);**

**buttonPanel = new Panel(new FlowLayout()); addBtn = new Button("Add");**

**updateBtn = new Button("Update"); deleteBtn = new Button("Delete"); clearBtn = new Button("Clear");**

**buttonPanel.add(addBtn); buttonPanel.add(updateBtn); buttonPanel.add(deleteBtn); buttonPanel.add(clearBtn); add(buttonPanel, BorderLayout.SOUTH);**

**addBtn.addActionListener(this); updateBtn.addActionListener(this); deleteBtn.addActionListener(this); clearBtn.addActionListener(this); displayPanel = new Panel(); displayPanel.setLayout(new BorderLayout()); displayArea = new TextArea(10, 40);**

**displayPanel.add(new Label("Employee Records:"), BorderLayout.NORTH); displayPanel.add(displayArea, BorderLayout.CENTER);**

**add(displayPanel, BorderLayout.EAST);**

**public void actionPerformed(ActionEvent e) { String command = e.getActionCommand(); String name = nameTxt.getText();**

**String id = idTxt.getText(); String dept = deptTxt.getText();**

**String salary = salaryTxt.getText(); if (command.equals("Add")) {**

**displayArea.append("Added: " + name + " | " + id + " | " + dept + " | " + salary + "\n");**

**} else if (command.equals("Update")) {**

**displayArea.append("Updated: " + name + " | " + id + " | " + dept + " | " + salary + "\n");**

**} else if (command.equals("Delete")) { displayArea.append("Deleted: " + id + "\n");**

**} else if (command.equals("Clear")) { nameTxt.setText(""); idTxt.setText(""); deptTxt.setText(""); salaryTxt.setText("");**

**}**

**}**

**public static void main(String[] args) { new Employee().setVisible(true);**

**}**

**}**

**Farenheit;**

**import java.awt.\*; import java.awt.event.\*;**

**import java.text.DecimalFormat;**

**public class CelsiusToFahrenheit extends Frame implements ActionListener { Label lblResult;**

**TextField txtCelsius, txtFahrenheit; Button btnCtoF, btnFtoC;**

**DecimalFormat df = new DecimalFormat("#.##");**

**public CelsiusToFahrenheit() { setLayout(new FlowLayout());**

**txtCelsius = new TextField(10); txtFahrenheit = new TextField(10);**

**btnCtoF = new Button("Celsius to Fahrenheit"); btnCtoF.addActionListener(this);**

**btnFtoC = new Button("Fahrenheit to Celsius"); btnFtoC.addActionListener(this);**

**lblResult = new Label("Result: ");**

**add(new Label("Celsius:")); add(txtCelsius); add(btnCtoF);**

**add(new Label("Fahrenheit:")); add(txtFahrenheit); add(btnFtoC);**

**add(lblResult);**

**addWindowListener(new WindowAdapter() { public void windowClosing(WindowEvent we) {**

**dispose(); System.*exit*(0);**

**}**

**});**

**setSize(320, 180); setTitle("Temperature Converter"); setVisible(true);**

**}**

**@Override**

**public void actionPerformed(ActionEvent e) { try {**

**if (e.getSource() == btnCtoF) {**

**double celsius = Double.*parseDouble*(txtCelsius.getText()); double fahrenheit = (celsius \* 9 / 5) + 32; lblResult.setText("Result: " + df.format(fahrenheit) + " °F");**

**} else if (e.getSource() == btnFtoC) {**

**double fahrenheit = Double.*parseDouble*(txtFahrenheit.getText()); double celsius = (fahrenheit - 32) \* 5 / 9; lblResult.setText("Result: " + df.format(celsius) + " °C");**

**}**

**} catch (NumberFormatException ex) { lblResult.setText("Invalid input");**

**}**

**}**

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

**new CelsiusToFahrenheit();**

**}**

**}**

**Network sockets:**

**Server.java :**

**import java.net.\*; import java.io.\*; public class Dserver{**

**public static DatagramSocket ds; static byte b[]=new byte[255]; static int c\_port=1001, s\_port=782;**

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

**BufferedReader br=new BufferedReader(new InputStreamReader(System.in)); System.out.println("Enter Data");**

**InetAddress ia = InetAddress.getByName("localhost"); ds = new DatagramSocket();**

**while(true)**

**{**

**String s=br.readLine(); if(s.equals("end"))**

**{**

**b=s.getBytes();**

**ds.send(new DatagramPacket(b,b.length,ia,s\_port));**

**}**

**b=s.getBytes();**

**ds.send(new DatagramPacket(b,b.length,ia,s\_port));**

**}**

**}**

**catch(Exception e){ System.out.println(e);**

**}**

**}**

**}**

**Client.java**

**import java.net.\*; import java.io.\*; public class Dclient{**

**public static DatagramSocket ds; static byte b[]=new byte[255]; static int c\_port=1001, s\_port=782;**

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

**ds = new DatagramSocket(s\_port); System.out.println("Its Waiting Bro.."); while(true){**

**DatagramPacket dp=new DatagramPacket(b,b.length); ds.receive(dp);**

**String msg=new String(dp.getData(),0,dp.getLength()); if(msg.equals("end"))**

**break; else**

**System.out.println(msg);**

**}**

**}**

**catch(Exception e){ System.out.println(e);**

**}**

**}**

**}**

**Library systemusing jdbc;**

**import java.awt.\*; import java.awt.event.\*; import java.sql.\*;**

**public class AWTsample extends Frame {**

**Label labelTitle, labelBookTitle, labelAuthor, labelYear, labelSearch; TextField textBookTitle, textAuthor, textYear, textSearch;**

**Button btnAdd, btnDisplay, btnSearch; TextArea textAreaDisplay;**

**Connection con; AWTsample() {**

**setTitle("Library Management"); setSize(500, 400); setLayout(null);**

**setVisible(true);**

**labelTitle.setBounds(150, 40, 200, 30); labelTitle.setFont(new Font("Dialog", Font.*BOLD*, 18)); add(labelTitle);**

**labelBookTitle = new Label("Book Title:"); labelBookTitle.setBounds(50, 90, 100, 25); add(labelBookTitle);**

**textBookTitle = new TextField(); textBookTitle.setBounds(160, 90, 180, 25); add(textBookTitle);**

**labelAuthor = new Label("Author:"); labelAuthor.setBounds(50, 130, 100, 25); add(labelAuthor);**

**textAuthor = new TextField(); textAuthor.setBounds(160, 130, 180, 25); add(textAuthor);**

**labelYear = new Label("Year:"); labelYear.setBounds(50, 170, 100, 25); add(labelYear);**

**textYear = new TextField(); textYear.setBounds(160, 170, 180, 25); add(textYear);**

**labelSearch = new Label("Search Book:"); labelSearch.setBounds(50, 200, 100, 25); add(labelSearch);**

**textSearch = new TextField(); textSearch.setBounds(160, 200, 180, 25); add(textSearch);**

**btnAdd = new Button("Add"); btnAdd.setBounds(50, 240, 80, 30); add(btnAdd);**

**textAreaDisplay = new TextArea(); textAreaDisplay.setBounds(50, 290, 400, 100); textAreaDisplay.setEditable(false); add(textAreaDisplay);**

**connectDatabase();**

**addWindowListener(new WindowAdapter() { public void windowClosing(WindowEvent we) {**

**System.*exit*(0);**

**}**

**});**

**}**

**void connectDatabase() { try {**

**Class.*forName*("com.mysql.cj.jdbc.Driver"); con = DriverManager.*getConnection*(**

**"jdbc:mysql://localhost:3306/login\_db", "root",**

**"gowtham"**

**);**

**}catch (Exception e) {**

**textAreaDisplay.setText("Database Connection Failes:"+ e.getMessage());**

**}**

**}**

**void addBook() { try {**

**String sql = "INSERT INTO books (title, author, year) VALUES (?, ?, ?)"; PreparedStatement ps = con.prepareStatement(sql);**

**ps.setString(1, textBookTitle.getText()); ps.setString(2, textAuthor.getText());**

**ps.setInt(3, Integer.*parseInt*(textYear.getText())); int rows = ps.executeUpdate(); textAreaDisplay.setText(rows + " book(s) added.");**

**textBookTitle.setText(""); textAuthor.setText(""); textYear.setText("");**

**} catch (Exception e) {**

**textAreaDisplay.setText("Error adding book: " + e.getMessage());**

**}**

**}**

**void displayBooks() { try {**

**Statement st = con.createStatement();**

**ResultSet rs = st.executeQuery("SELECT \* FROM books"); StringBuilder sb = new StringBuilder();**

**while (rs.next()) {**

**sb.append("ID: ").append(rs.getInt("id"))**

**.append(", Title: ").append(rs.getString("title"))**

**.append(", Author: ").append(rs.getString("author"))**

**.append(", Year: ").append(rs.getInt("year"))**

**.append("\n");**

**}**

**textAreaDisplay.setText(sb.toString());**

**} catch (Exception e) {**

**textAreaDisplay.setText("Error displaying books: " + e.getMessage());**

**}**

**}**

**void searchBook() { try {**

**String sql = "SELECT \* FROM books WHERE title LIKE ?"; PreparedStatement ps = con.prepareStatement(sql); ps.setString(1, "%" + textSearch.getText() + "%");**

**ResultSet rs = ps.executeQuery(); StringBuilder sb = new StringBuilder(); while (rs.next()) {**

**sb.append("ID: ").append(rs.getInt("id"))**

**.append(", Title: ").append(rs.getString("title"))**

**.append(", Author: ").append(rs.getString("author"))**

**.append(", Year: ").append(rs.getInt("year"))**

**.append("\n");**

**}**

**textAreaDisplay.setText(sb.length() > 0 ? sb.toString() : "No books found.");**

**} catch (Exception e) {**

**textAreaDisplay.setText("Search failed: " + e.getMessage());**

**}**

**}**

**public static void main(String[] args) { new AWTsample();**

**}**

**}**

**Login authentication;**

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

**public class LoginPage extends JFrame implements ActionListener { private JLabel lblUsername, lblPassword, lblMessage;**

**private JTextField txtUsername;**

**private JPasswordField txtPassword; private JButton btnLogin;**

**private Connection conn;**

**public LoginPage() { super("Login Page");**

**JPanel panel = new JPanel(new GridBagLayout()); GridBagConstraints gbc = new GridBagConstraints(); gbc.insets = new Insets(5, 5, 5, 5);**

**lblUsername = new JLabel("Username:");**

**txtUsername = new JTextField(20);**

**lblPassword = new JLabel("Password:"); txtPassword = new JPasswordField(20);**

**btnLogin = new JButton("Login"); btnLogin.addActionListener(this);**

**lblMessage = new JLabel(); lblMessage.setHorizontalAlignment(SwingConstants.*CENTER*);**

**gbc.gridx = 0;**

**gbc.gridy = 0;**

**gbc.anchor = GridBagConstraints.*EAST*; panel.add(lblUsername, gbc);**

**gbc.gridx = 1;**

**gbc.anchor = GridBagConstraints.*WEST*; panel.add(txtUsername, gbc);**

**gbc.gridx = 0;**

**gbc.gridy = 1;**

**gbc.anchor = GridBagConstraints.*EAST*; panel.add(lblPassword, gbc);**

**gbc.gridx = 1;**

**gbc.anchor = GridBagConstraints.*WEST*; panel.add(txtPassword, gbc);**

**gbc.gridx = 0;**

**gbc.gridy = 2;**

**gbc.gridwidth = 2;**

**gbc.anchor = GridBagConstraints.*CENTER*; panel.add(btnLogin, gbc);**

**gbc.gridy = 3; panel.add(lblMessage, gbc);**

**add(panel);**

**setDefaultCloseOperation(JFrame.*EXIT\_ON\_CLOSE*); setSize(350, 200);**

**setLocationRelativeTo(null); // Center on screen setVisible(true);**

**connectToDatabase();**

**}**

**private void connectToDatabase() { try {**

**String url = "jdbc:mysql://localhost:3306/login\_db"; String user = "darkseid";**

**String password = "gowtham";**

**conn = DriverManager.*getConnection*(url, user, password); System.*out*.println("Database connected.");**

**} catch (SQLException e) {**

**System.*out*.println("Database connection error: " + e); lblMessage.setText("Database connection failed."); lblMessage.setForeground(Color.*RED*);**

**}**

**}**

**@Override**

**public void actionPerformed(ActionEvent e) { String username = txtUsername.getText();**

**// Using getPassword for security**

**String password = new String(txtPassword.getPassword());**

**if (authenticate(username, password)) { lblMessage.setText("Login Successful!"); lblMessage.setForeground(new Color(0, 128, 0)); // dark green**

**} else {**

**lblMessage.setText("Invalid credentials."); lblMessage.setForeground(Color.*RED*);**

**}**

**}**

**private boolean authenticate(String username, String password) { try {**

**String query = "SELECT \* FROM users WHERE username = ? AND password = ?"; PreparedStatement pst = conn.prepareStatement(query);**

**pst.setString(1, username); pst.setString(2, password);**

**ResultSet rs = pst.executeQuery(); return rs.next();**

**} catch (SQLException e) { System.*out*.println("Authentication error: " + e);**

**}**

**return false;**

**}**

**public static void main(String[] args) { SwingUtilities.*invokeLater*(() -> new LoginPage());**

**}**

**}**

**Rmi;**

**Adder.java**

**import java.rmi.\*;**

**public interface Adder extends Remote{**

**public int add(int x, int y)throws RemoteException;**

**}**

# AdderRemote.java

**import java.rmi.RemoteException;**

**import java.rmi.server.UnicastRemoteObject;**

**public class CalculatorImpl extends UnicastRemoteObject implements Calculator { protected CalculatorImpl() throws RemoteException {**

**super();**

**}**

**public double add(double a, double b) throws RemoteException { return a + b;**

**}**

**public double subtract(double a, double b) throws RemoteException { return a - b;**

**}**

**public double multiply(double a, double b) throws RemoteException { return a \* b;**

**}**

**public double divide(double a, double b) throws RemoteException { if (b == 0) throw new ArithmeticException("Division by zero"); return a / b;**

**}**

**}**

# Myclient.java

**import java.rmi.\*; public class MyClient{**

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

**Adder stub=(Adder)Naming.lookup("rmi://localhost:5000/darkseid"); System.out.println(stub.add(100,4));**

**}**

**catch(Exception e){ System.out.println(e);**

**}**

**}**

**}**

**Myserver.java**

**import java.rmi.Naming;**

**import java.rmi.registry.LocateRegistry;**

**public class CalculatorServer {**

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

**LocateRegistry.createRegistry(5000); Calculator calc = new CalculatorImpl();**

**Naming.rebind("rmi://localhost:5000/CalculatorService", calc);**

**System.out.println("Calculator Server is ready.");**

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

**}**

**}**