**1.Write a program to implement concept of exception handling.**

public class ExcepTest{

public static void main(String args[]){

int a[] = new int[2];

try{

System.out.println("Access element three :" + a[3]);

}catch(ArrayIndexOutOfBoundsException e){

System.out.println("Exception thrown :" + e);

}

finally{

a[0] = 6;

System.out.println("First element value: " +a[0]);

System.out.println("The finally statement is executed");

}

}

}

**2. Write a program to implement concept of user defined exception handling.**

// File Name InsufficientFundsException.java

import java.io.\*;

public class InsufficientFundsException extends Exception

{

private double amount;

public InsufficientFundsException(double amount)

{

this.amount = amount;

}

public double getAmount()

{

return amount;

}

}

// File Name CheckingAccount.java

import java.io.\*;

public class CheckingAccount

{

private double balance;

private int number;

public CheckingAccount(int number)

{

this.number = number;

}

public void deposit(double amount)

{

balance += amount;

}

public void withdraw(double amount) throws

InsufficientFundsException

{

if(amount <= balance)

{

balance -= amount;

}

else

{

double needs = amount - balance;

throw new InsufficientFundsException(needs);

}

}

public double getBalance()

{

return balance;

}

public int getNumber()

{

return number;

}

}

The following BankDemo program demonstrates invoking the deposit() and withdraw() methods of CheckingAccount.

// File Name BankDemo.java

public class BankDemo

{

public static void main(String [] args)

{

CheckingAccount c = new CheckingAccount(101);

System.out.println("Depositing $500...");

c.deposit(500.00);

try

{

System.out.println("\nWithdrawing $100...");

c.withdraw(100.00);

System.out.println("\nWithdrawing $600...");

c.withdraw(600.00);

}catch(InsufficientFundsException e)

{

System.out.println("Sorry, but you are short $"

+ e.getAmount());

e.printStackTrace();

}

}

}

**3.Write a program to create single thread**

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();

}

}

**4.Write a program to create multiple thread.**

class ThreadDemo extends Thread {

private Thread t;

private String threadName;

ThreadDemo( 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[]) {

ThreadDemo T1 = new ThreadDemo( "Thread-1");

T1.start();

ThreadDemo T2 = new ThreadDemo( "Thread-2");

T2.start();

}

}

**5.Write an applet program to display message.**

import java.awt.\*;

import java.applet.\*;

/\*

<applet code="sim" width=300 height=300>

</applet>

\*/

public class sim extends Applet

{

String msg=" ";

public void init()

{

msg+="init()--->";

setBackground(Color.orange);

}

public void start()

{

msg+="start()--->";

setForeground(Color.blue);

}

public void paint(Graphics g)

{

msg+="paint()--->";

g.drawString(msg,200,50);

}

}

**6.Write an applet program for a moving banner.**

/\* <applet code="movingBanner" height=50 width=300> </applet> \*/

import java.awt.\*;

import java.applet.\*;

publicclass movingBanner extends Applet implements Runnable

{

String msg=" A moving Banner. ";

char ch;

boolean stopFlag= true;

Thread t= null;

publicvoid start(){

t = new Thread(this);

stopFlag=false;

t.start();

}

publicvoid run(){

for(;;){

try{

repaint();

Thread.sleep(250);

ch = msg.charAt(0);

msg = msg.substring(1,msg.length());

msg = msg + ch;

if(stopFlag)

break;

}catch(InterruptedException e) {}

}

}

publicvoid stop(){

stopFlag=true;

t = null;

}

publicvoid paint(Graphics g){

g.drawString(msg,60,30);

}

}

**7.Write an applet program to draw oval within triangle and change the background color using parameter.**

/\*

<applet code="DrawOvalsExample" width=500 height=500>

</applet>

\*/

import java.applet.Applet;

import java.awt.Color;

import java.awt.Graphics;

public class DrawOvalsExample extends Applet{

public void paint(Graphics g){

//set color to red

setForeground(Color.red);

/\*

\* to draw a oval in an applet window use,

\* void drawOval(int x1,int y1, int width, int height)

\* method.

\*

\* This method draws a oval of specified width and

\* height at (x1,y1)

\*/

//this will draw a oval of width 50 & height 100 at (10,10)

g.drawOval(10,10,50,100);

/\*

\* To draw a filled oval use

\* fillOval(int x1,int y1, int width, int height)

\* method of Graphics class.

\*/

//draw filled oval

g.fillOval(100,20,50,100);

}

}

**8.Write an applet program for a moving circle.**

import java.applet.Applet;

import java.awt.Color;

import java.awt.Graphics;

import java.awt.Graphics2D;

public class circles extends Applet implements Runnable {

int x = 0, y = 0;

public void start() {

setSize(500, 500);

setBackground(Color.BLACK);

new Thread(this).start();

}

public void run() {

while (true) {

try {

update();

Thread.sleep(50);

} catch (InterruptedException ex) {

}

}

}

public void update() {

x += 5;

y += 6;

repaint();

}

public void paint(Graphics e) {

super.paint(e);

Graphics2D g = (Graphics2D) e;

g.setColor(Color.BLUE);

g.fillOval(x, y, 20, 20);

}

}

**9.Write an applet program to implement awt class.**

import java.applet.\*;

import java.awt.\*;

public class HelloWorldApplet extends Applet

{

public void paint (Graphics g)

{

g.drawString ("Hello World", 25, 50);

}

}

10.Write an applet program to implement awt events.

import java.awt.\*;

import java.awt.event.\*;

public class AwtControlDemo {

private Frame mainFrame;

private Label headerLabel;

private Label statusLabel;

private Panel controlPanel;

public AwtControlDemo(){

prepareGUI();

}

public static void main(String[] args){

AwtControlDemo awtControlDemo = new AwtControlDemo();

awtControlDemo.showEventDemo();

}

private void prepareGUI(){

mainFrame = new Frame("Java AWT Examples");

mainFrame.setSize(400,400);

mainFrame.setLayout(new GridLayout(3, 1));

mainFrame.addWindowListener(new WindowAdapter() {

public void windowClosing(WindowEvent windowEvent){

System.exit(0);

}

});

headerLabel = new Label();

headerLabel.setAlignment(Label.CENTER);

statusLabel = new Label();

statusLabel.setAlignment(Label.CENTER);

statusLabel.setSize(350,100);

controlPanel = new Panel();

controlPanel.setLayout(new FlowLayout());

mainFrame.add(headerLabel);

mainFrame.add(controlPanel);

mainFrame.add(statusLabel);

mainFrame.setVisible(true);

}

private void showEventDemo(){

headerLabel.setText("Control in action: Button");

Button okButton = new Button("OK");

Button submitButton = new Button("Submit");

Button cancelButton = new Button("Cancel");

okButton.setActionCommand("OK");

submitButton.setActionCommand("Submit");

cancelButton.setActionCommand("Cancel");

okButton.addActionListener(new ButtonClickListener());

submitButton.addActionListener(new ButtonClickListener());

cancelButton.addActionListener(new ButtonClickListener());

controlPanel.add(okButton);

controlPanel.add(submitButton);

controlPanel.add(cancelButton);

mainFrame.setVisible(true);

}

private class ButtonClickListener implements ActionListener{

public void actionPerformed(ActionEvent e) {

String command = e.getActionCommand();

if( command.equals( "OK" )) {

statusLabel.setText("Ok Button clicked.");

}

else if( command.equals( "Submit" ) ) {

statusLabel.setText("Submit Button clicked.");

}

else {

statusLabel.setText("Cancel Button clicked.");

}

}

}

}

**11.Write an applet program to implement layout manager.**

import java.awt.\*;

import java.applet.\*;

public class LayoutExample extends Applet

{

Button okButton1;

Button okButton2;

Button okButton3;

Button okButton4;

Button okButton5;

public void init()

{

// sets the LayoutManager to BorderLayout

setLayout(new BorderLayout());

okButton1 = new Button("Centered Button");

okButton2 = new Button("Cold North");

okButton3 = new Button("Go West");

okButton4 = new Button("At East");

okButton5 = new Button("Hot South");

// always says where the component should be placed when adding

// Options are center,East,West,Nort and South

add(okButton1,"Center");

add(okButton2,"North");

add(okButton3,"West");

add(okButton4,"East");

add(okButton5,"South");

}

}

**12. Write a program to implent concept of swing class.**

import javax.swing.JFrame;

import javax.swing.JLabel;

//import statements

//Check if window closes automatically. Otherwise add suitable code

public class HelloWorldFrame extends JFrame {

public static void main(String args[]) {

new HelloWorldFrame();

}

HelloWorldFrame() {

JLabel jlbHelloWorld = new JLabel("Hello World");

add(jlbHelloWorld);

this.setSize(100, 100);

// pack();

setVisible(true);

}

}

**13.Write a program to implement concept of jdbc**

import java.sql.\*;

public class FirstExample {

// JDBC driver name and database URL

static final String JDBC\_DRIVER = "com.mysql.jdbc.Driver";

static final String DB\_URL = "jdbc:mysql://localhost/EMP";

// Database credentials

static final String USER = "username";

static final String PASS = "password";

public static void main(String[] args) {

Connection conn = null;

Statement stmt = null;

try{

//STEP 2: Register JDBC driver

Class.forName("com.mysql.jdbc.Driver");

//STEP 3: Open a connection

System.out.println("Connecting to database...");

conn = DriverManager.getConnection(DB\_URL,USER,PASS);

//STEP 4: Execute a query

System.out.println("Creating statement...");

stmt = conn.createStatement();

String sql;

sql = "SELECT id, first, last, age FROM Employees";

ResultSet rs = stmt.executeQuery(sql);

//STEP 5: Extract data from result set

while(rs.next()){

//Retrieve by column name

int id = rs.getInt("id");

int age = rs.getInt("age");

String first = rs.getString("first");

String last = rs.getString("last");

//Display values

System.out.print("ID: " + id);

System.out.print(", Age: " + age);

System.out.print(", First: " + first);

System.out.println(", Last: " + last);

}

//STEP 6: Clean-up environment

rs.close();

stmt.close();

conn.close();

}catch(SQLException se){

//Handle errors for JDBC

se.printStackTrace();

}catch(Exception e){

//Handle errors for Class.forName

e.printStackTrace();

}finally{

//finally block used to close resources

try{

if(stmt!=null)

stmt.close();

}catch(SQLException se2){

}// nothing we can do

try{

if(conn!=null)

conn.close();

}catch(SQLException se){

se.printStackTrace();

}//end finally try

}//end try

System.out.println("Goodbye!");

}//end main

}//end FirstExample