**Experiment 5**

class MyException extends Exception{

String a;

MyException(String b){

a=b;

}

public String toString() {

return("Exception:"+ a);

}

}

import java.util.\*;

public class Exp5 {

public static void main(String[] args) {

int bal,amt;

Scanner sc = new Scanner(System.in);

System.out.println("Enter the balance:");

bal=sc.nextInt();

System.out.println("Enter the withdrawl amount:");

amt=sc.nextInt();

try {

bal=bal-amt;

if(bal<0)

throw new MyException("Insufficient Balance");

System.out.println("Remaining Balance"+ bal);

}

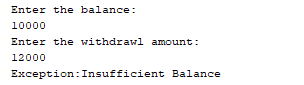
catch(MyException e) {

System.out.println(e);

}

}

}



**Experiment 4**

import java.util.\*;

public class Experiment4 {

public static void main(String[] args) {

figure b=new triangle();

double ar=b.area();

System.out.println("The area of the triangle is "+ar);

b=new rectangle();

ar=b.area();

System.out.println("The area of the rectangle is "+ar);

}

}

public abstract class figure {

abstract double area();

}

public class rectangle extends figure {

@Override double area(){

System.out.println("Enter the side1");

Scanner sc=new Scanner(System.in);

float a=sc.nextFloat();

System.out.println("Enter the side2");

float b=sc.nextFloat();

double ar=a\*b;

return ar;

}

}

public class triangle extends figure {

@Override double area(){

System.out.println("Enter the base");

Scanner sc=new Scanner(System.in);

float a=sc.nextFloat();

System.out.println("Enter the height");

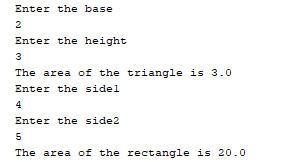
float b=sc.nextFloat();

double ar=0.5\*a\*b;

return ar;

}

}



**Part b**

public class Experiment4part2 {

public static void main(String[] args) {

java j=new java();

j.op();

}

}

public interface compiler {

void op();

}

public interface interpreter {

void op();

}

public class java implements compiler,interpreter {

@Override public void op(){

System.out.println("Multiple inheritence using interface");

}

}



**Experiment 6**

public class Multithreading {

public static void main(String[] args) {

threadmake t1=new threadmake();

try{

for(int j=5;j>0;j--){

System.out.println("Running main "+j);

Thread.sleep(750);

}

}

catch(InterruptedException e){

System.out.println("IE caught");

}

System.out.println("Exiting main");

}

}

public class threadmake extends Thread {

threadmake(){

super("Using thread class");

try{

System.out.println("Child thread "+this);

Thread.sleep(2000);

start();

}

catch(InterruptedException e){

System.out.println("IE caught");

}

}

public void run(){

try{

for(int i=5;i>0;i--){

System.out.println("child thread"+i);

Thread.sleep(250);

}

}

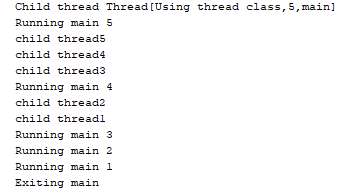
catch(InterruptedException e){

System.out.println("IE caught");

}

}

}



**Part b**

public class Synchronisation {

public static void main(String[] args) {

callit target=new callit();

caller obj1=new caller(target,"hello");

caller obj2=new caller(target,"Synchronized");

caller obj3=new caller(target,"world");

}

}

public class caller implements Runnable{

String msg;

callit target;

Thread t;

public caller(callit targ,String s){

target=targ;

msg=s;

t=new Thread(this);

t.start();

}

@Override

public void run(){

synchronized(target){

target.call(msg);

}

}

}

public class callit {

void call(String msg){

System.out.println("Out: "+msg);

try{

Thread.sleep(1000);

}

catch(InterruptedException e){

System.out.println("IE caught");

}

}

}



**Experiment 7**

package exp\_7;

import java.applet.\*;

import java.awt.\*;

public class app7 extends Applet {

public void paint(Graphics g){

g.drawRect(100,100,200,200);

Font f1=new Font("Arial",Font.PLAIN,18);

Font f2=new Font("Arial",Font.PLAIN,22);

Font f3=new Font("Arial",Font.PLAIN,24);

g.setFont(f1);

g.setColor(Color.red);

g.drawString("Large", 110, 120);

g.setFont(f2);

g.setColor(Color.blue);

g.drawString("Larger", 110, 150);

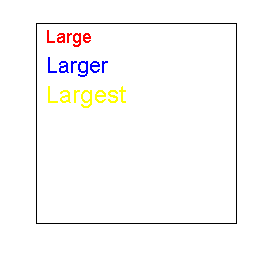
g.setFont(f3);

g.setColor(Color.yellow);

g.drawString("Largest", 110, 180);

}

}



**Part b**

package exp\_7;

import java.applet.\*;

import java.awt.\*;

public class face extends Applet {

public void paint(Graphics g){

g.drawString("Always smilling",50,30);

g.drawOval(60,60,200,200);

g.fillOval(90, 120, 50, 20);

g.fillOval(190, 120, 50, 20);

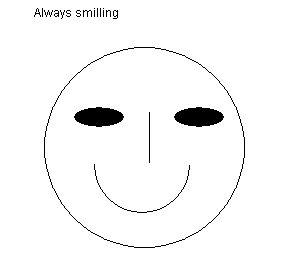
g.drawLine(165, 125, 165, 175);

g.drawArc(110, 130, 95, 95, 0, -180);

g.drawLine(165, 125, 165, 175);

}

}



**Experiment-8**

package exp\_7;

import java.applet.\*;

import java.awt.\*;

import java.applet.\*;

import java.awt.event.\*;

public class event\_h extends Applet implements ActionListener {

Button b1=new Button("OK");

public void init(){

add(b1);

b1.addActionListener(this);

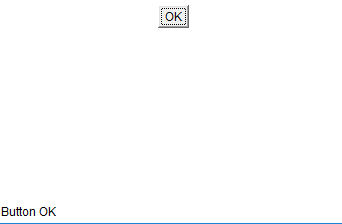
}

public void actionPerformed(ActionEvent e){

showStatus("Button OK");

}

}



Part b

package exp8;

import java.applet.\*;

import java.awt.\*;

import java.awt.event.\*;

public class scroll extends Applet implements AdjustmentListener{

Scrollbar s1,s2,s3;

public void init(){

s1=new Scrollbar(Scrollbar.HORIZONTAL,0,50,0,255);

s2=new Scrollbar(Scrollbar.HORIZONTAL,0,50,0,255);

s3=new Scrollbar(Scrollbar.HORIZONTAL,0,50,0,255);

s1.addAdjustmentListener(this);

s2.addAdjustmentListener(this);

s3.addAdjustmentListener(this);

add(s1);

add(s2);

add(s3);

}

public void adjustmentValueChanged(AdjustmentEvent e){

repaint();

}

public void paint(Graphics g){

int x,y,z;

x=s1.getValue();

y=s2.getValue();

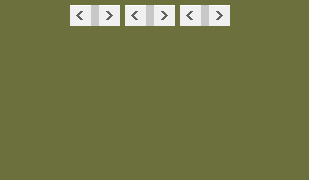
z=s3.getValue();

Color c=new Color(x,y,z);

setBackground(c);

}

}



Part c

package exp8;

import java.applet.\*;

import java.awt.\*;

import java.awt.event.\*;

public class moose extends Applet implements MouseListener{

public void init(){

addMouseListener(this);

}

public void mouseClicked(MouseEvent e){

showStatus("Mouse clicked at "+e.getX()+" "+e.getY());

}

public void mouseEntered(MouseEvent e){

showStatus("Mouse entered at "+e.getX()+" "+e.getY());

}

public void mouseExited(MouseEvent e){

showStatus("Mouse Exited at "+e.getX()+" "+e.getY());

}

public void mousePressed(MouseEvent e){

showStatus("Mouse Pressed at "+e.getX()+" "+e.getY());

}

public void mouseReleased(MouseEvent e){

showStatus("Mouse released at "+e.getX()+" "+e.getY());

}

}

