1. Explicați efectul instrucțiunilor //1, //2, //3 și //4. Precizați ce mesaje sunt afișate.

void main(){

B \*pb;

pb=new D(); //1

pb->m(); //2

pb->s(); //3

delete pb; //4

}

class D: public B{

public:

D(){cout<<"D()"<<endl;}

virtual ~D(){cout<<"~D()"<<endl;}

virtual void m(){cout<<"D::m()"<<endl;}

void s(){cout<<"D::s()"<<endl;}

};

#include <iostream.h>

class B{

public:

B(){cout<<"B()"<<endl;}

virtual ~B(){cout<<"~B()"<<endl;}

virtual void m(){cout<<"B::m()"<<endl;}

void s(){cout<<"B::s()"<<endl;}

};

1. Considerați următoarea definiție a clasei TwoDim.
   1. Să se supraîncarce operatorul << astfel încât instrucțiunea // 1 să afișeze  
      \*px=1; \*py =2;
   2. Explicați rezultatul afișat de instrucțiunea // 3
   3. Supraîncărcați operatorul = astfel încât instrucțiunea //3 să afișeze \*px=100; \*py =200;

void main(){

TwoDim x(1,2), y(10,20);

cout<<x<<endl; // 1

x=y; // 2

x.set(100,200);

y.set(1000,2000);

cout<<x<<endl; // 3

}

#include <iostream.h>

class TwoDim{

public:

TwoDim(int x, int y){ px=new int;

py=new int; \*px=x; \*py=y;

}

void set(int x, int y){\*px=x; \*py=y;}

private:

int \*px, \*py;

friend ostream& operator<<(ostream& o, const TwoDim t);

};

1. Adăugați programului de mai jos clasa Impl astfel încât prin executarea programului să fie afișat mesajul  
   Hello, Tudor!

public class Interf{

public static void main(String[] args){

ModAfisare ma= new Impl("Tudor");

ma.afisare("Hello");

}

}

interface ModAfisare{

public void afisare(String s);

}

1. În următorul program Java vi se cere să explicați:
   1. Rezultatele afișate;
   2. Rezultatele afișate, daca linia //synchr este stearsă;
   3. Rezultatele afișate, daca linia //static este stearsă (dar //synchr este prezentă);
   4. Ce se intamplă dacă se înlocuiește cuvântul cheie **while** prin **if**?

public class ExecFir{

public static void main (String[] args){

Fir f1, f2;

f1= new Fir(); f1.start();

f2= new Fir(); f2.start();

**while**(f1.isAlive() || f2.isAlive()){}

System.out.println("x="+Fir.x);

}

}

class Fir extends Thread{

static int x=0;

static //static

Object sem =new Object();

int t;

void incr() {

synchronized(sem) //synchr

{t=x;t++; x=t;}

}

public void run() { incr();}

}

1. Se consideră următoarea interfaţă Java, în care sunt definite câteva operaţii elementare asupra listelor de obiecte:

interface ListOperations{

public void add(Object o);// adauga un obiect in lista

// returneaza, fara a sterge,

// elementul curent din lista

public Object currentElement();

public void delete();// sterge elementul curent

}

Se cere să implementaţi această interfaţă printr-o clasă Queue ale cărei obiecte sunt liste eterogene First In First Out (cozi în care elementul current are cea mai mare vechime în listă şi este indicat de cursor).

Aceste liste sunt observabile în sensul următor: când cursorul este deplasat, noul element curent este transmis către toate obiectele observatoare (aici includem şi cazul în care se adaugă un element într-o listă vidă).

Obiectele observatoare sunt din clasa QueueObserver, pe care trebuie de asemenea s-o definiţi.

Completând corespunzător //…1, //…2, //…3 şi //…4, programul test de mai jos trebuie să afişeze rezultatele următoare:

100

100

Tudor

Andrei

class Queue //. . .1

public Queue(int mD){maxDim=mD; v=new Object[maxDim];}

// metode

//. . .2

//atribute

private int free=0; //pozitia libera

private int cursor=0;// pozitia cu elementul util

// 0<=numberOfElements<=maxDim,

//numarul real de elemente din coada

private int numberOfElements=0;

private Object v[];

private int maxDim;

}

class QueueObserver //. . .3

public QueueObserver(Queue q){obsQ=q;}

// metoda update

//. . .4

// atribute

private Queue obsQ; // coada observabila

}

q.add(new Integer(400));

System.out.println

(q.currentElement());

q.delete();

q.add(new Integer(500));

q.delete();

q.add(new Integer(100));

}

}

public class ObservableList{

public static void

main(String[] args){

Queue q=new Queue(4);

QueueObserver qO= new QueueObserver(q);

q.addObserver(qO);

q.add(new Integer(100));

q.add("Tudor");

q.add("Andrei");