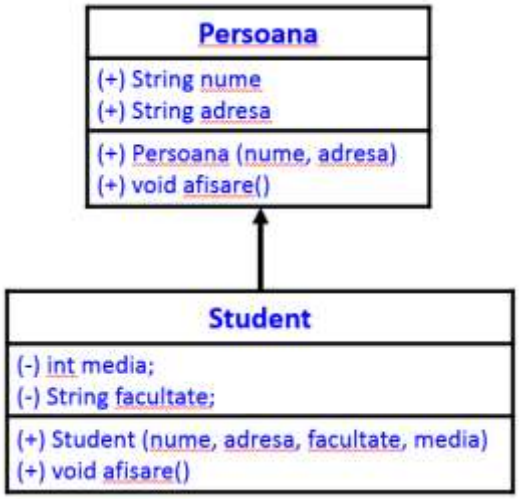
copyright@www.adrian.runceanu.ro

Laborator 11 - Java
Mostenire

Probleme rezolvate:

Scrieti, compilati si rulati urmatoarele 3 exemple din acest laborator:

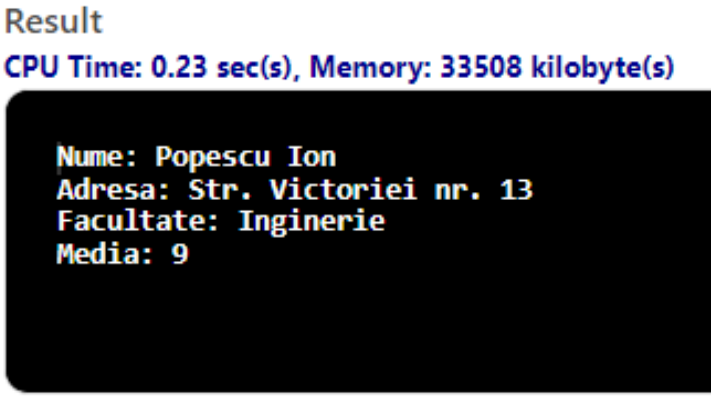



```
class Persoana
{
	private String nume;
	private String adresa;
	// constructor
	public Persoana (String nume, String adresa)
	{
		this.nume = nume;
		this.adresa = adresa;
	}
	public void afisare ()
	{
		System.out.println ("Nume: " + nume);
		System.out.println ("Adresa: " + adresa);
	}
}

class Student extends Persoana
{
	private int media;
	private String facultate;
	public Student (String nume, String adresa, String facultate, int media)
	{
		super (nume, adresa);
```

copyright@www.adrian.runceanu.ro

```
                this.media = media;
                this.facultate = facultate;
        }
        public void afisare ()
        {
                super.afisare ();
                System.out.println ("Facultate: " + facultate);
                System.out.println ("Media: " + media);
        }
}
public class MyClass{
        public static void main(String args[])  {
                Student ob = new Student("Popescu Ion", "Str. Victoriei nr. 13",
"Inginerie", 9);
                ob.afisare();
        }
}
```

Rezultatul executiei programului este:



Recomandam utilizarea compilatorului online **https://www.jdoodle.com/online-java-compiler-ide/** (varianta de compilator Java Advanced, in care se poate salva codul programului in clasa dorita).

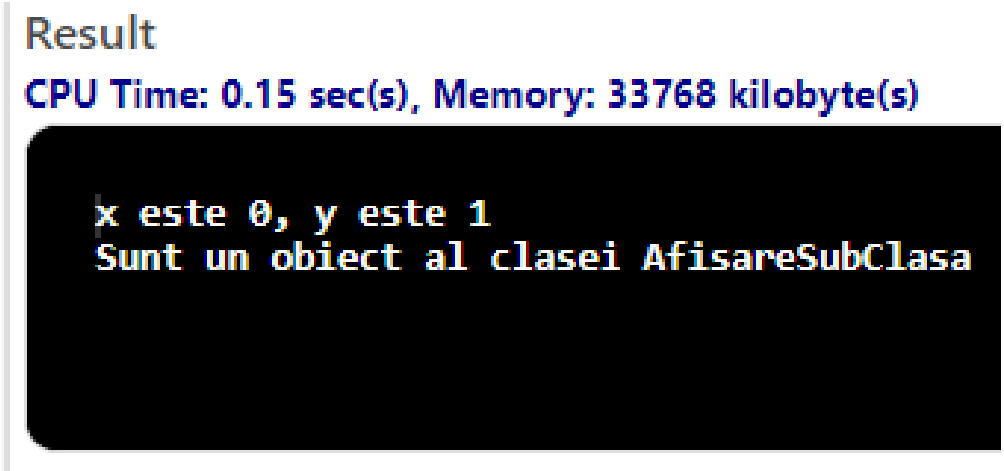
copyright@www.adrian.runceanu.ro

2. Programul urmator (AfisareSubClasa.java) prezinta o clasa ce contine o metoda *afisareDate()*, care afiseaza numele clasei si valorile variabilelor de instanta.
De asemenea, in acelasi fisier-sursa este inclusa si clasa denumita *AfisareSubClasa* derivata din clasa *AfisareClasa*.
A fost creat un obiect de tip *AfisareSubClasa* si a fost apelata metoda *afisareDate()*.
Deoarece clasa *AfisareSubClasa* nu defineste aceasta metoda, **Java** o cauta in superclasele clasei *AfisareSubClasa*, incepand cu superclasa *AfisareClasa*, unde gaseste metoda *afisareDate()* si o executa.

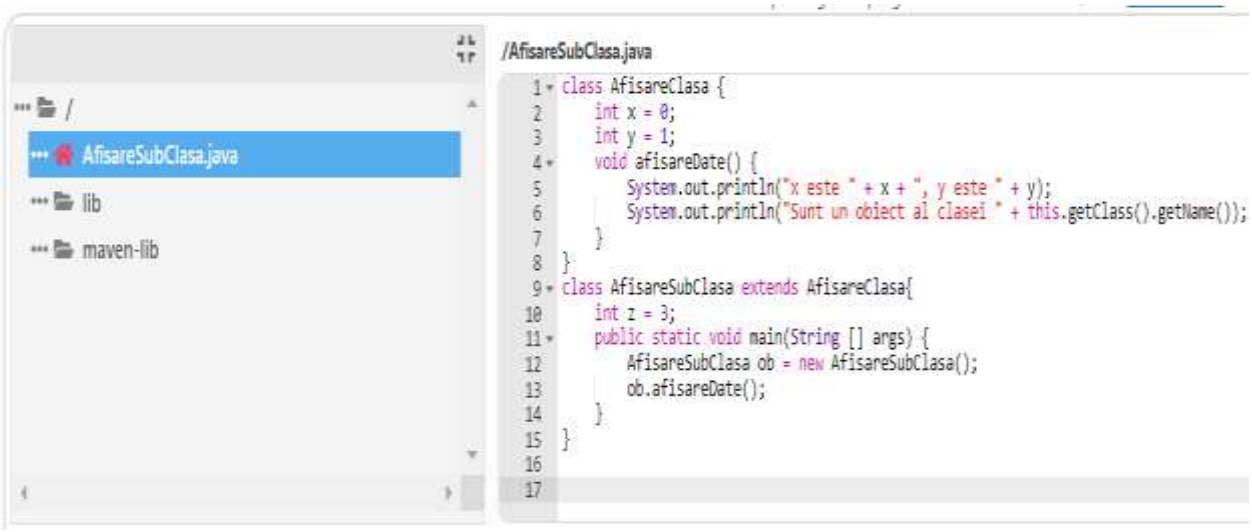
```
class AfisareClasa {
        int x = 0;
        int y = 1;
        void afisareDate() {
                System.out.println("x este " + x + ", y este " + y);
                System.out.println("Sunt un obiect al clasei " +
this.getClass().getName());
  }
}
class AfisareSubClasa extends AfisareClasa  {
        int z = 3;
   public static void main(String [] args) {
        AfisareSubClasa ob = new AfisareSubClasa();
        ob.afisareDate();
   }
}
```

Rezultatul executiei programului este:

x este 0, y este 1
Sunt un obiect al clasei AfisareSubClasa



copyright@www.adrian.runceanu.ro



3. Programul urmator (TestPatrulater.java) implementeaza clasa Paralelogram care mosteneste clasa Patrulater (**fisierul Patrulater.java**).

```
class Patrulater{
        double xA,yA,xB,yB,xC,yC,xD,yD;
        int valid;
        public Patrulater(double x1,double y1,double x2,double y2,double x3,double
y3,double x4,double y4)
        {
                xA=x1;          yA=y1;
                xB=x2;          yB=y2;
                xC=x3;          yC=y3;
                xD=x4;          yD=y4;
                valid_1();
        }

public int valid_1()
{
        if(((xA!=xB)||(yA!=yB))&&((xA!=xC)||(yA!=yC))&&
((xA!=xD)||(yA!=yD))&&((xB!=xC)||(yB!=yC))&&((xB!=xD)||      (yB!=yD))      &&
((xC!=xD)||(yC!=yD)))
                valid=1;
        else valid=0;
        return valid;
 }
 void afis_1()
 {
        if (valid==1) System.out.println("Figura este patrulater.");
        else System.out.println("Figura NU este patrulater.");
 }
};
```

copyright@www.adrian.runceanu.ro

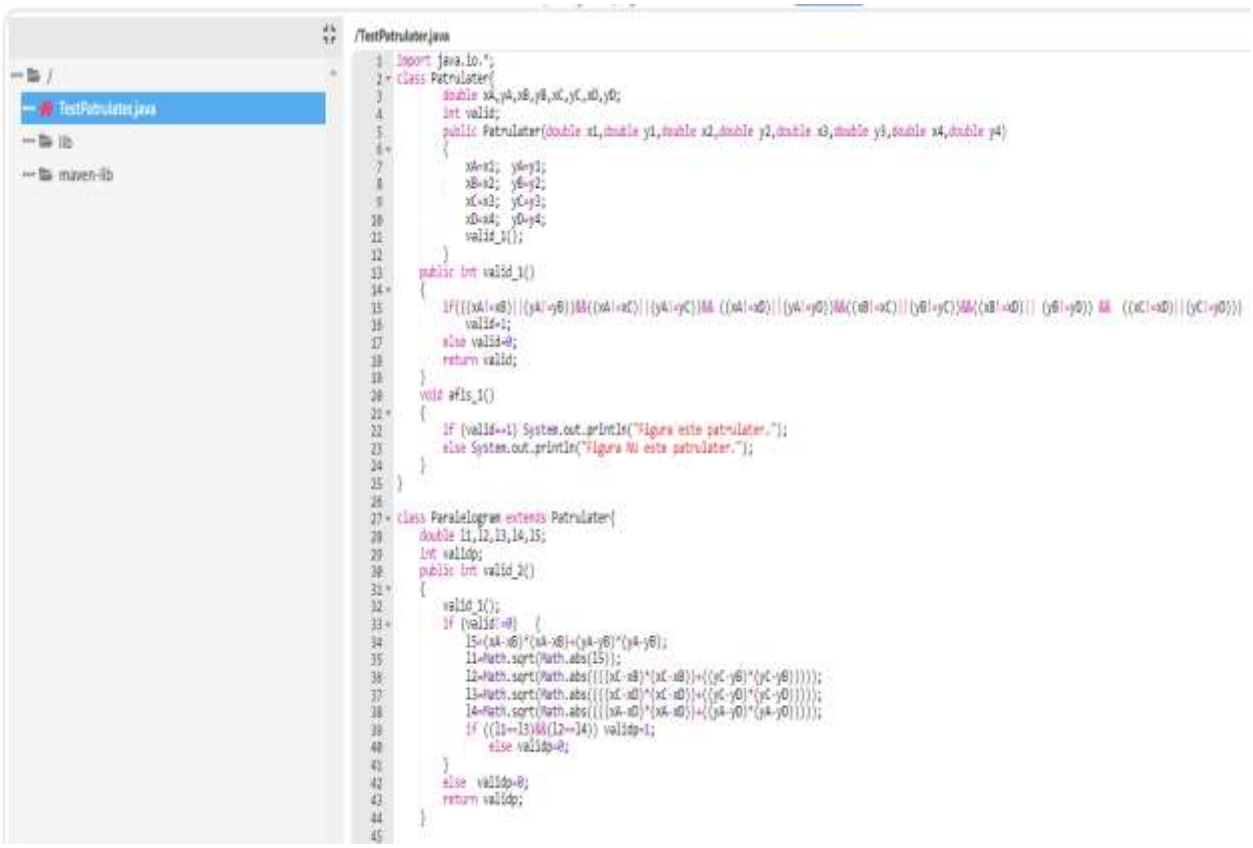
Fisierul TestPatrulater.java

```
import java.io.*;
class Paralelogram extends Patrulater{
        double l1,l2,l3,l4,l5;
        int validp;
        public int valid_2()
        {
                valid_1();
                if (valid!=0)    {
                   l5=(xA-xB)*(xA-xB)+(yA-yB)*(yA-yB);
                   l1=Math.sqrt(Math.abs(l5));
                   l2=Math.sqrt(Math.abs((((xC-xB)*(xC-xB))+((yC-yB)*(yC-yB)))));
                   l3=Math.sqrt(Math.abs((((xC-xD)*(xC-xD))+((yC-yD)*(yC-yD)))));
                   l4=Math.sqrt(Math.abs((((xA-xD)*(xA-xD))+((yA-yD)*(yA-yD)))));
                   if ((l1==l3)&&(l2==l4)) validp=1;
                        else validp=0;
                }
                else  validp=0;
                return validp;
        }

        public void afis_2()
        {
                if (validp!=0) System.out.println("Figura este paralelogram.");
                else System.out.println("Figura NU este paralelogram.");;
        }
        public Paralelogram(double x1, double y1, double x2, double y2, double x3,
double y3, double x4, double y4)
        {
                super(x1,y1,x2,y2,x3,y3,x4,y4);
                valid_2();
        }
};


class TestPatrulater{
        public static void main(String [] argv) throws IOException  {
                Paralelogram pp = new Paralelogram(0.0,0.0,1.0,0.0,1.0,1.0,0.0,1.0);
                pp.valid_1();
                pp.afis_1();
                pp.valid_2();
                pp.afis_2();
                System.out.println("Dati coordonatele varfurilor unei alte figuri
geometrice:");
                InputStreamReader isr = new InputStreamReader(System.in);
                BufferedReader br = new BufferedReader(isr);
                String s1 = br.readLine();
```

copyright@www.adrian.runceanu.ro

```
                pp.xA = Double.parseDouble(s1);
                System.out.println("xA = "+pp.xA);
                String s2 = br.readLine();
                pp.yA = Double.parseDouble(s2);
                System.out.println("xA = "+pp.yA);
                String s3 = br.readLine();
                pp.xB = Double.parseDouble(s3);
                System.out.println("xB = "+pp.xB);
                String s4 = br.readLine();
                pp.yB = Double.parseDouble(s4);
                System.out.println("yB = "+pp.yB);

                String s5 = br.readLine();
                pp.xC = Double.parseDouble(s5);
                System.out.println("xC = "+pp.xC);
                String s6 = br.readLine();
                pp.yC = Double.parseDouble(s6);
                System.out.println("yC = "+pp.yC);
                String s7 = br.readLine();
                pp.xD = Double.parseDouble(s7);
                System.out.println("xD = "+pp.xD);
                String s8 = br.readLine();
                pp.yD = Double.parseDouble(s8);
                System.out.println("yD = "+pp.yD);
                pp.valid_1();
                pp.afis_1();
                pp.valid_2();
                pp.afis_2();
        }
}
```

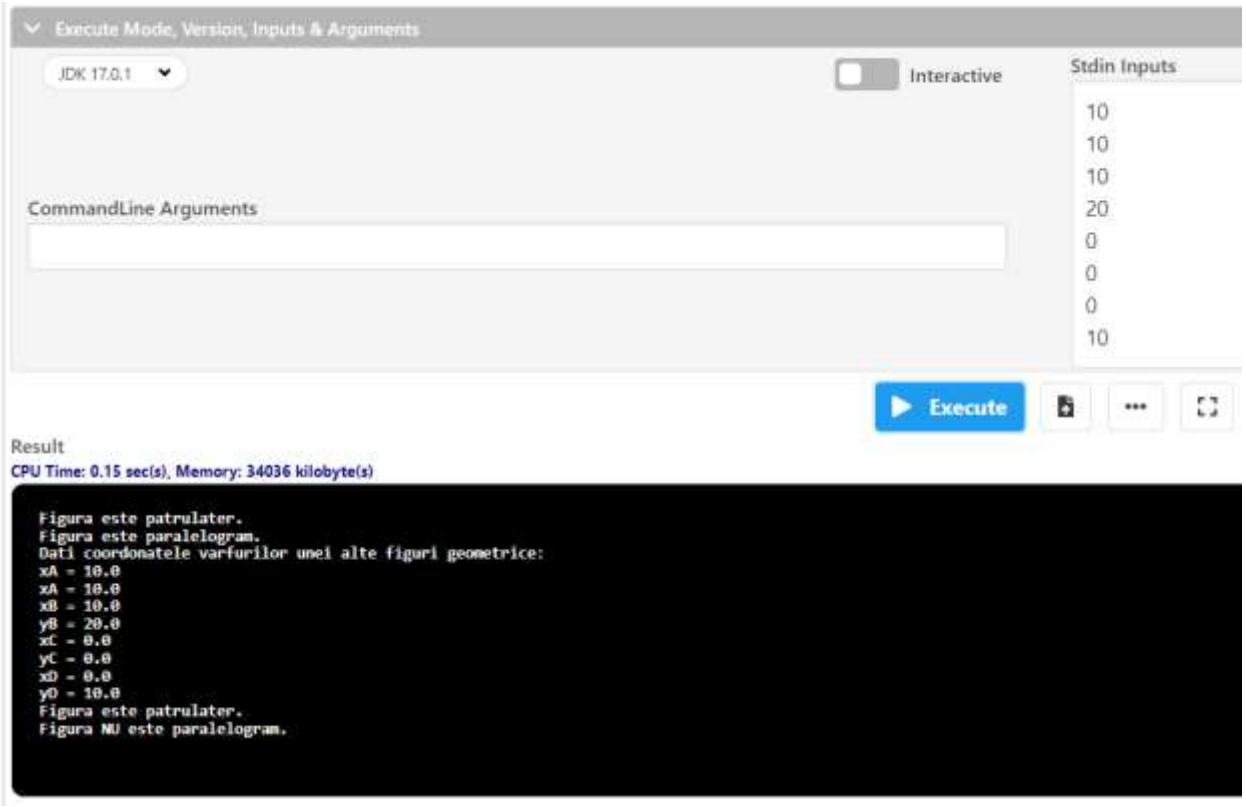
copyright@www.adrian.runceanu.ro




```
import java.io.*;
class Patrulater{
        double xA,yA,xB,yB,xC,yC,xD,yD;
        int valid;
        public Patrulater(double x1,double y1,double x2,double y2,double x3,double y3,double x4,double y4)
        {
            xA=x1;  yA=y1;
            xB=x2;  yB=y2;
            xC=x3;  yC=y3;
            xD=x4;  yD=y4;
            valid_1();
        }
    public int valid_1()
    {
        if(((xA!=xB)||(yA!=yB))&&((xA!=xC)||(yA!=yC))&& ((xA!=xD)||(yA!=yD))&&((xB!=xC)||(yB!=yC))&&((xB!=xD)|| (yB!=yD)) && ((xC!=xD)||(yC!=yD)))
            valid=1;
        else valid=0;
        return valid;
    }
    void afis_1()
    {
        if (valid==1) System.out.println("Figura este patrulater.");
        else System.out.println("Figura NU este patrulater.");
    }
}

class Paralelogram extends Patrulater{
    double l1,l2,l3,l4,l5;
    int validp;
    public int valid_2()
    {
        valid_1();
        if (valid!=0)  {
            l5=(xA-xB)*(xA-xB)+(yA-yB)*(yA-yB);
            l1=Math.sqrt(Math.abs(l5));
            l2=Math.sqrt(Math.abs(((xC-xB)*(xC-xB))+((yC-yB)*(yC-yB))));
            l3=Math.sqrt(Math.abs(((xC-xD)*(xC-xD))+((yC-yD)*(yC-yD))));
            l4=Math.sqrt(Math.abs(((xA-xD)*(xA-xD))+((yA-yD)*(yA-yD))));
            if ((l1==l3)&&(l2==l4)) validp=1;
                else validp=0;
        }
        else  validp=0;
        return validp;
    }

```

```
    public void afis_2()
    {
        if (validp!=0) System.out.println("Figura este paralelogram.");
        else System.out.println("Figura NU este paralelogram.");;
    }
    public Paralelogram(double x1, double y1, double x2, double y2, double x3, double y3, double x4, double y4)
        {
            super(x1,y1,x2,y2,x3,y3,x4,y4);
            valid_2();
        }
}

class TestPatrulater{
    public static void main(String [] argv) throws IOException
    {
        Paralelogram pp = new Paralelogram(0.0,0.0,1.0,0.0,1.0,1.0,0.0,1.0);
        pp.valid_1();
        pp.afis_1();
        pp.valid_2();
        pp.afis_2();

        System.out.println("Dati coordonatele varfurilor unei alte figuri geometrice:");
        InputStreamReader isr = new InputStreamReader(System.in);
        BufferedReader br = new BufferedReader(isr);

        String s1 = br.readLine(); pp.xA = Double.parseDouble(s1);  System.out.println("xA = "+pp.xA);
        String s2 = br.readLine(); pp.yA = Double.parseDouble(s2);  System.out.println("xA = "+pp.yA);
        String s3 = br.readLine(); pp.xB = Double.parseDouble(s3);  System.out.println("xB = "+pp.xB);
        String s4 = br.readLine(); pp.yB = Double.parseDouble(s4);  System.out.println("yB = "+pp.yB);
        String s5 = br.readLine(); pp.xC = Double.parseDouble(s5);  System.out.println("xC = "+pp.xC);
        String s6 = br.readLine(); pp.yC = Double.parseDouble(s6);  System.out.println("yC = "+pp.yC);
        String s7 = br.readLine(); pp.xD = Double.parseDouble(s7);  System.out.println("xD = "+pp.xD);
        String s8 = br.readLine(); pp.yD = Double.parseDouble(s8);  System.out.println("yD = "+pp.yD);

        pp.valid_1();
        pp.afis_1();
        pp.valid_2();
        pp.afis_2();
    }
}
```

copyright@www.adrian.runceanu.ro



Probleme propuse spre rezolvare

Lab11_1: Clasa **Complex**. Aceasta clasa reprezinta numerele complexe. Sa se defineasca aceasta clasa impreuna cu o metoda pentru *adunarea a doua numere complexe*, precum si a 2(doua) metode accesor care intoarce partea imaginara si partea reala a unui numar complex. Clasa va avea doi constructori, unul care sa initializeze obiectele cu valori implicite si altul care initializeaza variabilele cu valorile trimise ca parametri constructorului.

Lab11_2: Clasa **Complex**. Aceasta clasa reprezinta numerele complexe. Sa se defineasca aceasta clasa impreuna cu o metoda pentru *scaderea a doua numere complexe*, precum si a 2(doua) metode accesor care intoarce partea imaginara si partea reala a unui numar complex. Clasa va avea doi constructori, unul care sa initializeze obiectele cu valori implicite si altul care initializeaza variabilele cu valorile trimise ca parametri constructorului.

Lab11_3: Clasa **Complex**. Aceasta clasa reprezinta numerele complexe. Sa se defineasca aceasta clasa impreuna cu o metoda pentru *inmultirea a doua numere complexe*, precum si a 2(doua) metode accesor care intoarce partea imaginara si partea reala a unui numar complex. Clasa va avea doi constructori, unul care sa initializeze obiectele cu valori implicite si altul care initializeaza variabilele cu valorile trimise ca parametri constructorului.

copyright@www.adrian.runceanu.ro

Lab11_4: Clasa **Complex**. Aceasta clasa reprezinta numerele complexe. Sa se defineasca aceasta clasa impreuna cu o metoda pentru *impartirea a doua numere complexe*, precum si a 2(doua) metode accesor care intoarce partea imaginara si partea reala a unui numar complex. Clasa va avea doi constructori, unul care sa initializeze obiectele cu valori implicite si altul care initializeaza variabilele cu valorile trimise ca parametri constructorului.

Lab11_5: Clasa **Complex**. Sa se calculeze si sa se afiseze z^n, unde n este o valoare n intreaga citita de la tastatura, iar z este un obiect din clasa **Complex**.

Lab11_6: Clasa **Complex**. Sa se calculeze si sa se afiseze *modulul unui numar complex* z, unde z este un obiect din clasa **Complex**.

Lab11_7: Clasa **Complex**. Sa se calculeze si sa verifice daca un numar complex z este real (adica y are valoarea 0), cu ajutorul unei metode de tip boolean, z este un obiect din clasa **Complex**.

Lab11_8: Clasa **Complex**. Sa se sorteze un vector de numere complexe cu obiecte din clasa Complex.

Lab11_9: Sa se defineasca clasa **Real**, care sa mosteneasca clasa **Complex**, dar care care sa nu aiba parte imaginara!(adica y este 0). Sa se defineasca doua obiecte, unul din clasa **Complex**, iar unul din clasa **Real** si sa se adune cele doua numere.

Lab11_10: Sa se defineasca clasa **Real**, care sa mosteneasca clasa **Complex**, dar care care sa nu aiba parte imaginara!(adica y este 0). Sa se defineasca doua obiecte, unul din clasa **Complex**, iar unul din clasa **Real** si sa se scada cele doua numere.

Lab11_11: Sa se defineasca clasa **Real**, care sa mosteneasca clasa **Complex**, dar care care sa nu aiba parte imaginara!(adica y este 0). Sa se defineasca doua obiecte, unul din clasa **Complex**, iar unul din clasa **Real** si sa se imparta cele doua numere.

Bibliografie:

[1] http://www.pbinfo.ro *Descrierea site-ului:"www.pbinfo.ro îți propune să rezolvi probleme de informatică, cu evaluator automat. Știi pe loc dacă soluția ta este corectă sau dacă trebuie să mai lucrezi la ea. Problemele sunt grupate după programa de informatică pentru liceu. Dar nu trebuie să fii la liceu ca să rezolvi aceste probleme. Poți fi elev de gimnaziu, student, profesor sau pur și simplu pasionat de informatică. De fapt, trebuie doar să vrei!!"*
[2] https://www.runceanu.ro/adrian
[3] Adrian Runceanu *„Programarea şi utilizarea calculatoarelor"*, Editura Academica Brâncuşi din Târgu-Jiu, 2003, ISBN 973-8436-44-3
[4] Adrian Runceanu, Mihaela Runceanu, *„Noțiuni de programare – limbajul C++"*, Editura Academica Brâncuși din Târgu-Jiu, 2012, ISBN 978-973-144-550-2
[5] Adrian Runceanu, Mihaela Runceanu, *„Algoritmi implementati in limbajul C++. Volumul I – Algoritmi elementari"*, Editura Academica Brâncuși din Târgu Jiu, 2021, ISBN 978-606-9614-06-8