



Laborator 10 – Java

1. Metodele unei clase de obiecte

2. Constructorii unei clase de obiecte

Probleme rezolvate:

Scrieti, compilati si rulati toate exemplele din acest laborator:

1. Definirea si apelul metodelor

Programele urmatoare (*ClasaTablou1.java* si *ClasaTablou2.java*) prezinta un exemplu de creare a unei clase care defineste o metoda numita *creareTablou*.

Acesta preia doua numere naturale (o limita inferioara si una superioara) si creaza un tablou unidimensional care contine toate numerele naturale aflate intre cele doua limite, inclusiv aceste limite.

Varianta de apel a unei metode prin crearea si utilizarea unei instante a clasei in care a fost definita metoda:

```
public class ClasaTablou1
{
    int [] creareTablou(int inf, int sup)
    {
        int [] tabl = new int[(sup - inf) + 1];
        for (int i = 0; i < tabl.length; i++)
            tabl[i] = inf++;
        return tabl;
    }
    public static void main(String args[])
    {
        ClasaTablou1 unTablou = new ClasaTablou1();
        int [] tablou = unTablou.creareTablou(1,10);
        System.out.print("Tablou: [ ");
        for (int i = 0; i < tablou.length; i++)
            System.out.print(tablou[i] + " ");
        System.out.println("");
    }
}
```

Rezultatul executiei programului este:

Tablou: [1 2 3 4 5 6 7 8 9 10]

```

1 public class ClasaTablou1
2 {
3     int [] creareTablou(int inf, int sup)
4     {
5         int [] tabl = new int[(sup - inf) +1];
6         for (int i = 0 ; i < tabl.length; i++)
7             tabl[i] = inf++;
8         return tabl;
9     }
10    public static void main(String args[]) {
11        ClasaTablou1 unTablou = new ClasaTablou1();
12        int [] tablou = unTablou.creareTablou(1,10);
13        System.out.print("Tablou: [ ");
14        for (int i = 0; i < tablou.length; i++)
15            System.out.print(tablou[i] + " ");
16        System.out.println("]");
17    }
18 }

```

Result

CPU Time: 0.16 sec(s), Memory: 33344 kilobyte(s)

```
Tablou: [ 1 2 3 4 5 6 7 8 9 10 ]
```

Varianta de apel a unei metode prin simpla folosire a numelui metodei, deoarece metoda este definita si apelata in aceeași clasă.

```

public class ClasaTablou2 {
    static int [] creareTablou(int inf, int sup)
    {
        int [] tabl = new int[(sup - inf) +1];
        for (int i = 0; i < tabl.length; i++)
            tabl[i] = inf++;
        return tabl;
    }
    public static void main(String args[])
    {
        int [] tablou = creareTablou(1,10);
        System.out.print("Tablou: [ ");
    }
}

```

```
        for (int i = 0; i < tablou.length; i++)  
            System.out.print(tablou[i] + " ");  
            System.out.println("");  
    }  
}
```

Rezultatul executiei programului este:

Tablou: [1 2 3 4 5 6 7 8 9 10]

```
1 public class ClasaTablou2 {  
2     static int [] creareTablou(int inf, int sup)  
3     {  
4         int [] tabl = new int[(sup - inf) +1];  
5         for (int i = 0 ; i < tabl.length; i++)  
6             tabl[i] = inf++;  
7         return tabl;  
8     }  
9     public static void main(String args[])  
10    {  
11        int [] tablou = creareTablou(1,10);  
12        System.out.print("Tablou: [ ");  
13        for (int i = 0; i < tablou.length; i++)  
14            System.out.print(tablou[i] + " ");  
15        System.out.println("]");  
16    }  
17 }
```

Result

CPU Time: 0.12 sec(s), Memory: 33236 kilobyte(s)

```
Tablou: [ 1 2 3 4 5 6 7 8 9 10 ]
```

2. Constructorii unei clase de obiecte

2. Sa se defineasca clasa **Student**. Clasa memoreaza numele studentului si rezultatele obtinute la trei examene. In cazul in care studentul nu a promovat, rezultatul la examenul respectiv este 0. De asemenea, clasa va avea doi constructori si va implementa metode pentru obtinerea numelui, pentru calculul mediei si pentru afisarea rezultatelor. Metoda de afisare va testa daca studentul a promovat toate examenele inainte de a calcula media. In cazul in care studentul nu a promovat un examen, va fi afisat un mesaj corespunzator.

```

public class student{
    // data membru de tip private
    public String nume;
    public double test1, test2, test3;// notele obtinute de student
    // metode de tip constructor
    student(String nume)
    {
        this.nume=nume;
    }

    student(String nume, double test1, double test2, double test3)
    {
        this.nume=nume;
        this.test1=test1;
        this.test2=test2;
        this.test3=test3;
    }

    // metode accesori
    public String getNume()
    {
        return nume;
    }
    public double getMedie()
    {
        return (test1 + test2 + test3) /3;           // calcul media
    }

    // metoda de afisare a datelor studentului
    public void afisare()
    {
        System.out.println("Salut, " + this.getNume() + ". Notele tale sunt:
");
        System.out.println("Test1 :" + test1);
        System.out.println("Test2 :" + test2);
        System.out.println("Test3 :" + test3);
        // testeaza daca studentul a promovat toate examenele
        if(test1==0 || test2==0 || test3==0)
            System.out.println("Nu ai promovat toate examenele!");
    }
}

```

```
        else
            System.out.println("Media este = " + this.getMedie());
    }
    public static void main(String args[]){
        // declaram un vector de studenti
        student student[] = new student[3];
        student[0]=new student("Grigore");
        student[1]=new student("Marian",10,6,8);
        student[2]=new student("Mariana",9,9.50,10);
        for(int i=0;i<student.length;i++)
            student[i].afisare();
    }
}
```

Rezultatul executiei programului este:

Salut, Mihai. Notele tale sunt:

Test1 :0.0

Test2 :0.0

Test3 :0.0

Nu ai promovat toate examenele!

Salut, Marian. Notele tale sunt:

Test1 :10.0

Test2 :6.0

Test3 :8.0

Media este = 8.0

Salut, Mihaela. Notele tale sunt:

Test1 :9.0

Test2 :9.5

Test3 :10.0

Media este = 9.5

```

1- public class student{
2 // data membru de tip private
3 public String nume;
4 public double test1, test2, test3; // notele obtinute de student
5 // metoda de tip constructor
6 student(String nume)
7 { this.nume=nume; }
8
9 student(String nume, double test1, double test2, double test3)
10 {
11     this.nume=nume;
12     this.test1=test1;
13     this.test2=test2;
14     this.test3=test3;
15 }
16
17 // metode accesori
18 public String getNume()
19 {
20     return nume;
21 }
22 public double getMedie()
23 {
24     return (test1 + test2 + test3) /3; // calcul media aritmetica
25 }
26
27 // metoda de afisare a datelor studentului
28 public void afisare()
29 {
30     System.out.println("Salut, " + this.getNume() + ". Notele tale sunt: ");
31     System.out.println("Test1 : " + test1);
32     System.out.println("Test2 : " + test2);
33     System.out.println("Test3 : " + test3);
34     // testam daca studentul a promovat toate examenele
35     if(test1==0 || test2==0 || test3==0)
36         System.out.println("Nu ai promovat toate examenele!");
37     else
38         System.out.println("Media este = " + this.getMedie());
39 }
40
41 public static void main(String args[])
42 {
43     // declaram un vector de studenti
44     student student[] = new student[3];
45     student[0]=new student("Grigore");
46     student[1]=new student("Marian",10,6,8);
47     student[2]=new student("Mariana",9,9,10);
48     for(int i=0;i<student.length;i++)
49         student[i].afisare();
50 }
51
52

```

Result

CPU Time: 0.18 sec(s), Memory: 34960 kilobyte(s)

```

Salut, Grigore. Notele tale sunt:
Test1 :0.0
Test2 :0.0
Test3 :0.0
Nu ai promovat toate examenele!
Salut, Marian. Notele tale sunt:
Test1 :10.0
Test2 :6.0
Test3 :8.0
Media este = 8.0
Salut, Mariana. Notele tale sunt:
Test1 :9.0
Test2 :9.5
Test3 :10.0
Media este = 9.5

```

3. Clasa **Automobil**. Clasa are variabile kilometrajPlecare, kilometrajSosire de tip int si litri de tip double. Constructorul clasei va instatia un obiect de tip automobil care pleaca din punctul indicat de parametrul kmStart, ajunge in punctul indicat de parametrul kmStop si consuma o cantitate de benzina egala cu valoarea parametrului litri. Clasa are o metoda consum() care va calcula numarul de km efectuati de automobile cu un litru de benzina.

```
public class Automobil{
    // data membru de tip private
    int kilometrajPlecare;
    int kilometrajSosire;
    double litri;

    // metoda de tip constructor
    Automobil(int kmStart, int kmStop, double litri)
    {
        kilometrajPlecare = kmStart;
        kilometrajSosire = kmStop;
        this.litri = litri;
    }

    // metoda de calcul a consumului
    public double consum()
    {
        return (kilometrajSosire - kilometrajPlecare )/litri;
    }

    public static void main(String args[]){
        Automobil automobil = new Automobil (78943, 78964, 10.6);
        System.out.println("Automobilul realizeaza
"+automobil.consum()+" km cu un litru de benzina");
    }
}
```

Rezultatul executiei programului este:

Automobilul realizeaza 1.9811320754716981 km cu un litru de benzina

```

1+ public class Automobil{
2     // data membru de tip private
3     int kilometrajPlecare;
4     int kilometrajSosire;
5     double litri;
6
7     // metoda de tip constructor |
8     Automobil(int kmStart, int kmStop, double litri)
9     {
10        kilometrajPlecare = kmStart;
11        kilometrajSosire = kmStop;
12        this.litri = litri;
13    }
14
15    // metoda de calcul a consumului
16    public double consum()
17    {
18        return (kilometrajSosire - kilometrajPlecare )/litri;
19    }
20
21    public static void main(String args[]){
22        Automobil automobil = new Automobil (78943, 78964, 18.6);
23        System.out.println("Automobilul realizeaza "+automobil.consum()+" km cu un litru de benzina");
24    }
25 }
26

```

Result
CPU Time: 0.14 sec(s), Memory: 35136 kilobyte(s)

```

Automobilul realizeaza 1.9811320754716981 km cu un litru de benzina

```

4. Clasa **ContEconomii** care are atributele numarCont, titularCont de tip String si bilant de tip int. Clasa are metode pentru afisarea bilantului current, efectuarea unei depuneri si a unei plati prin intermediul unui cec. Metoda pentru efectuarea unei depuneri va adauga o suma, specificata ca parametru al metodei la bilantul total al contului. Metoda pentru emiterea unui cec nu trebuie sa permita realizarea platii daca suma solicitata depaseste suma care se afla in cont. Pentru fiecare cec emis se percepe un comision de 15\$.

```

public class ContEconomii{
    // data membru de tip private
    String numarCont;
    String titularCont;
    double bilant;

    // metoda de tip constructor
    ContEconomii(String numarCont, String titularCont, double bilant)
    {
        this.numarCont = numarCont;
        this.titularCont = titularCont;
        this.bilant = bilant;
    }
}

```



```

        // metode instanta
        public double bilantCurent()
        {
            return bilant;
        }

        public void procesareDepozit(int suma)
        {
            bilant = bilant + suma;
        }

        public void emitereCec(int suma)
        {
            int comision = 15;
            if((suma > bilant) || (bilant < comision))
                System.out.println("Nu aveti bani suficienti in cont");
            else
            {
                System.out.println("Operatie efectuata");
                bilant = bilant - suma - comision;
            }
        }

    public static void main(String args[])
    {
        ContEconomii cont1 = new ContEconomii ("842021549", "Badea Grigore",
        100000);
        ContEconomii cont2 = new ContEconomii ("245385804", "Popescu Mimi",
        5000);

        System.out.println(cont1.numarCont + " " + cont1.titularCont + " " +
        cont1.bilantCurent());
        System.out.println(cont2.numarCont + " " + cont2.titularCont + " " +
        cont2.bilantCurent());

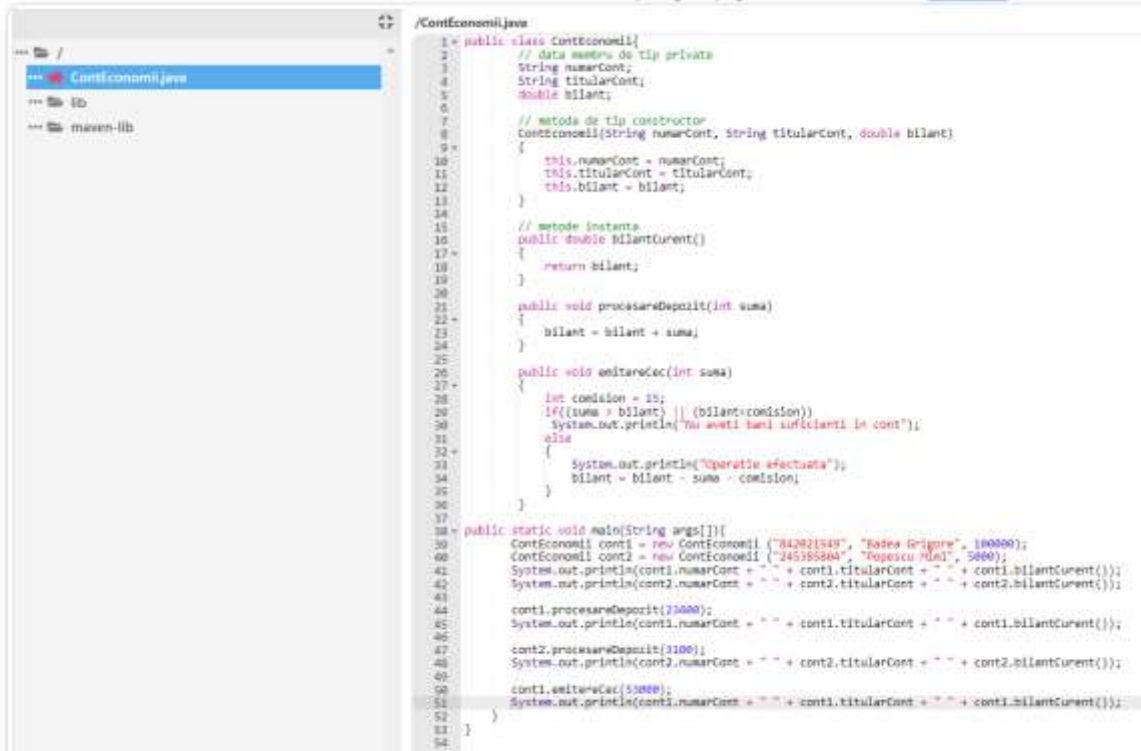
        cont1.procesareDepozit(23400);
        System.out.println(cont1.numarCont + " " + cont1.titularCont + " " +
        cont1.bilantCurent());

        cont2.procesareDepozit(3100);
        System.out.println(cont2.numarCont + " " + cont2.titularCont + " " +
        cont2.bilantCurent());

        cont1.emitereCec(53000);
        System.out.println(cont1.numarCont + " " + cont1.titularCont + " " +
        cont1.bilantCurent());
    }
}

```

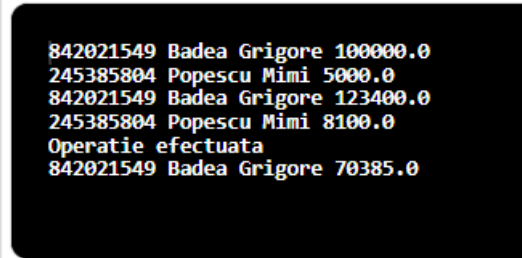
Rezultatul executiei programului este:
842021549 Badea Grigore 100000.0
245385804 Popescu Mimi 5000.0
842021549 Badea Grigore 123400.0
245385804 Popescu Mimi 8100.0
Operatie efectuata
842021549 Badea Grigore 70385.0



```
1 public class ContEconomici {
2     // data nume de tip privata
3     String numarCont;
4     String titularCont;
5     double bilant;
6
7     // metoda de tip constructor
8     ContEconomici(String numarCont, String titularCont, double bilant)
9     {
10        this.numarCont = numarCont;
11        this.titularCont = titularCont;
12        this.bilant = bilant;
13    }
14
15    // metode instance
16    public double bilantCurent()
17    {
18        return bilant;
19    }
20
21    public void procesareDepozit(int suma)
22    {
23        bilant = bilant + suma;
24    }
25
26    public void emitereCec(int suma)
27    {
28        int comision = 15;
29        if((suma > bilant) || (bilant < comision))
30            System.out.println("Nu aveti bani suficienti la cont");
31        else
32        {
33            System.out.println("Operatie efectuata");
34            bilant = bilant - suma - comision;
35        }
36    }
37
38    public static void main(String args[]) {
39        ContEconomici cont1 = new ContEconomici ("842021549", "Badea Grigore", 100000);
40        ContEconomici cont2 = new ContEconomici ("245385804", "Popescu Mimi", 5000);
41        System.out.println(cont1.numarCont + " " + cont1.titularCont + " " + cont1.bilantCurent());
42        System.out.println(cont2.numarCont + " " + cont2.titularCont + " " + cont2.bilantCurent());
43
44        cont1.procesareDepozit(23000);
45        System.out.println(cont1.numarCont + " " + cont1.titularCont + " " + cont1.bilantCurent());
46
47        cont2.procesareDepozit(3100);
48        System.out.println(cont2.numarCont + " " + cont2.titularCont + " " + cont2.bilantCurent());
49
50        cont1.emitereCec(5000);
51        System.out.println(cont1.numarCont + " " + cont1.titularCont + " " + cont1.bilantCurent());
52    }
53 }
54 }
```

Result

CPU Time: 0.23 sec(s), Memory: 35556 kilobyte(s)



```
842021549 Badea Grigore 100000.0
245385804 Popescu Mimi 5000.0
842021549 Badea Grigore 123400.0
245385804 Popescu Mimi 8100.0
Operatie efectuata
842021549 Badea Grigore 70385.0
```

Probleme propuse spre rezolvare

Lab10_1: Clasa Complex. Sa se defineasca o clasa “Complex” pentru operatii cu numere complexe si sa se testeze metodele implementate. Clasa va avea doi constructori astfel:

- cu doi parametri (parte reala si parte imaginara)
- fara parametri (constuctor implicit)
- Metodele necesare sunt: adunare, inmultire, ridicare la putere naturala si afisare (sir de caractere de forma (real,imaginar)).

Lab10_2: Clasa Stiva. Sa se defineasca o clasa “Stiva” pentru stive de numere intregi, reprezentate prin vectori de numere intregi.

Datele clasei:

- Un vector de intregi
- Indicele elementului din varful stivei (ultimul introdus)

Metodele clasei:

- Constructor fara parametri (dimensiunea implicita a stivei = 100)
- Constructor cu un parametru care reprezinta dimensiunea stivei
- void push(int): pune un intreg dat pe stiva
- int pop(): scoate elementul din varful stivei
- boolean isEmpty(): verifica daca stiva este vida
- Considerati cazurile de stiva plina (la push) si stiva vida (la pop).

Lab10_3: Clasa Multime. Sa se defineasca o clasa Multime pentru o multime de numere intregi, care contine un vector numere intregi.

Metodele clasei:

- Constructor cu un parametru care reprezinta dimensiunea multimii
- void add(int): adaugare element, daca nu exista
- boolean contains(int): verifica daca un numar dat se afla sau nu in multime
- afisare: multimea de elemente transformata in sir de caractere

Sa se scrie un program care creaza o multime folosind clasa Multime si testeaza operatiile de mai sus prin adaugari succesive si afisari.

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 și utilizarea calculatoarelor”, Editura Academica Brâncuși 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