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

Rezultatul executiei programului este:
Tabloul: [1 2 3 4 5 6 7 8 9 10]

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

Result

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

```
Tabloul: [ 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 aceeasi clasa.

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

Rezultatul executiei programului este:
Tabloul: [1 2 3 4 5 6 7 8 9 10]

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

Result

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

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

2. Constructorii unei clase de obiecte

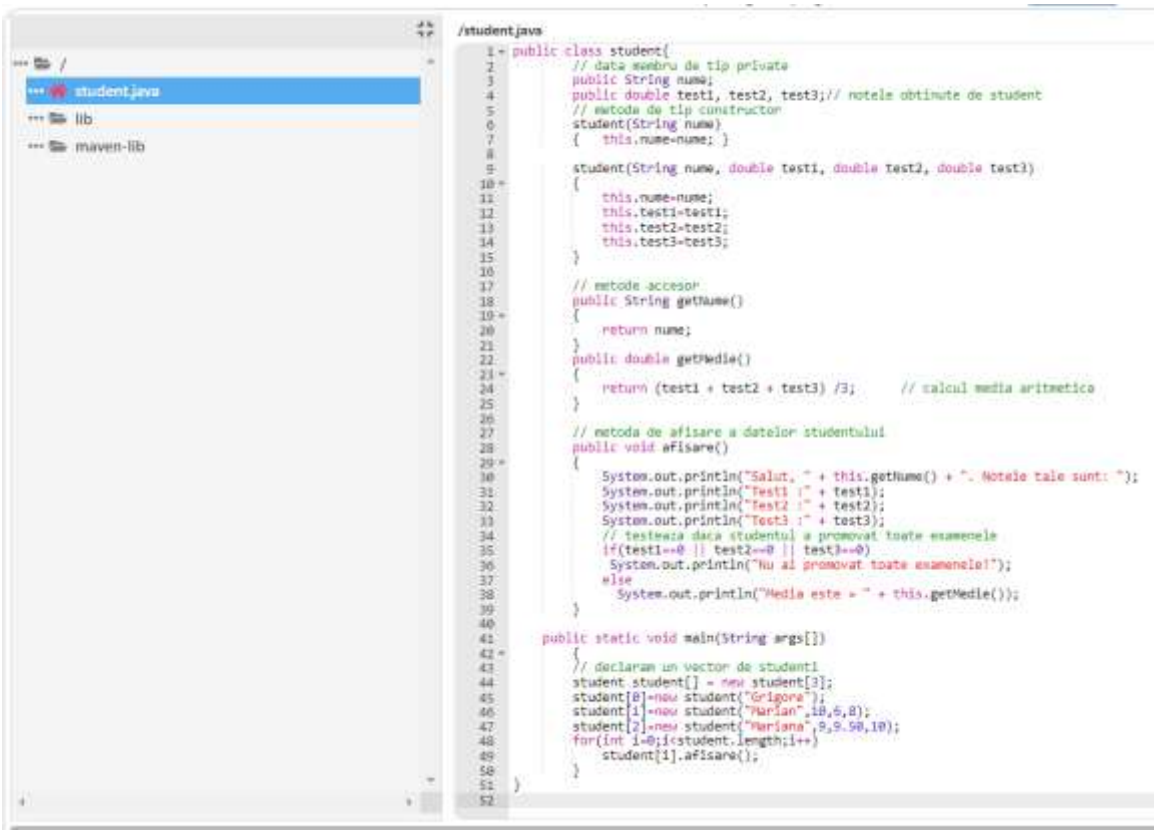
2. Sa se defineasca clasa **Student**. Clasa memoreaza numele studentului si rezultatele obtinute la trei examene. In cazul in care studentul nu a promovat, rezultatul la examenul respectiv este 0. De asemenea, clasa va avea doi constructori si va implementa metode pentru obtinerea numelui, pentru calculul 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 accesor
		public String getNume()
		{
			return nume;
		}
		public double getMedie()
		{
			return (test1 + test2 + test3) /3;		// calcul media aritmetica
		}

		// 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



Result

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

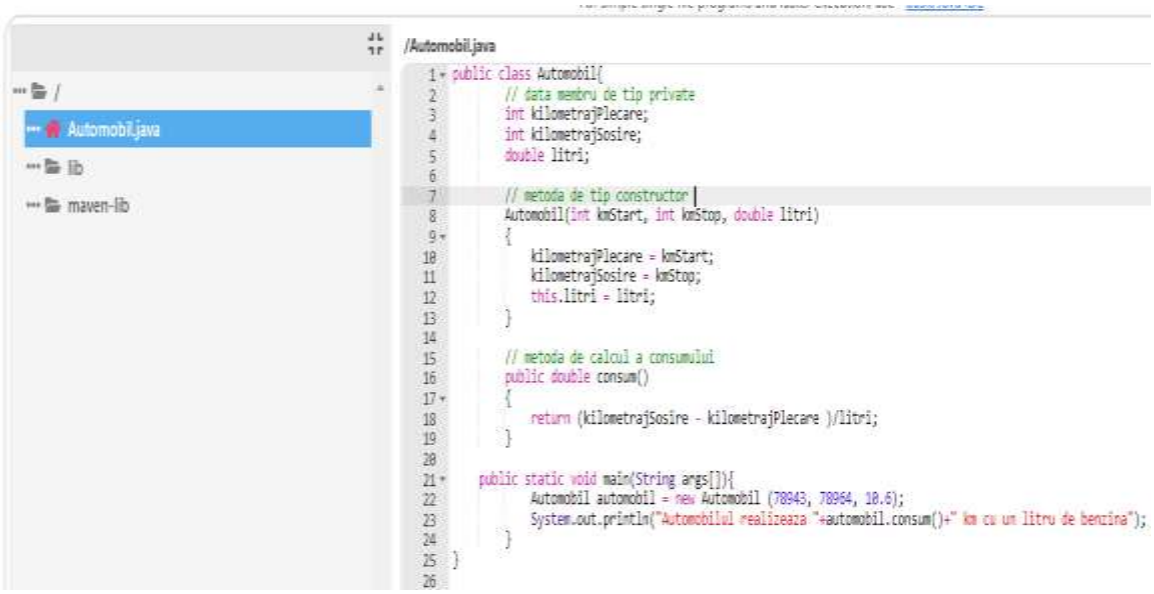
```
Salut, Grigore. Notele tale sunt:
Test1 :0.0
Test2 :0.0
Test3 :0.0
Nu ai promovat toate examenele!
Salut, Marian. Notele tale sunt:
Test1 :10.0
Test2 :6.0
Test3 :8.0
Media este = 8.0
Salut, Mariana. Notele tale sunt:
Test1 :9.0
Test2 :9.5
Test3 :10.0
Media este = 9.5
```

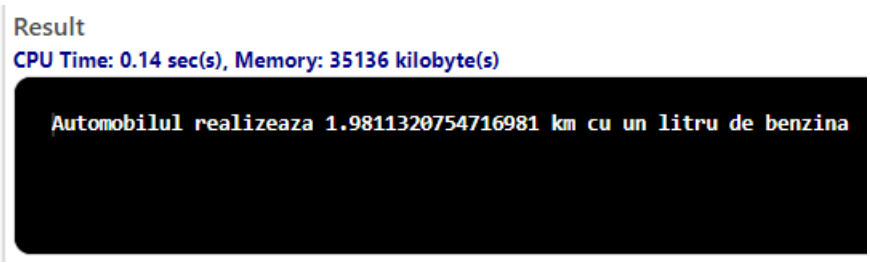
3. Clasa **Automobil**. Clasa are variabille kilometrajPlecare, kilometrajSosire de tip int si litri de tip double. Constructorul clasei va instatia un obiect de tip automobil care pleaca din punctul indicat de parametrul kmStart, ajunge in punctul indicat de parametrul kmStop si consuma o cantitate 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





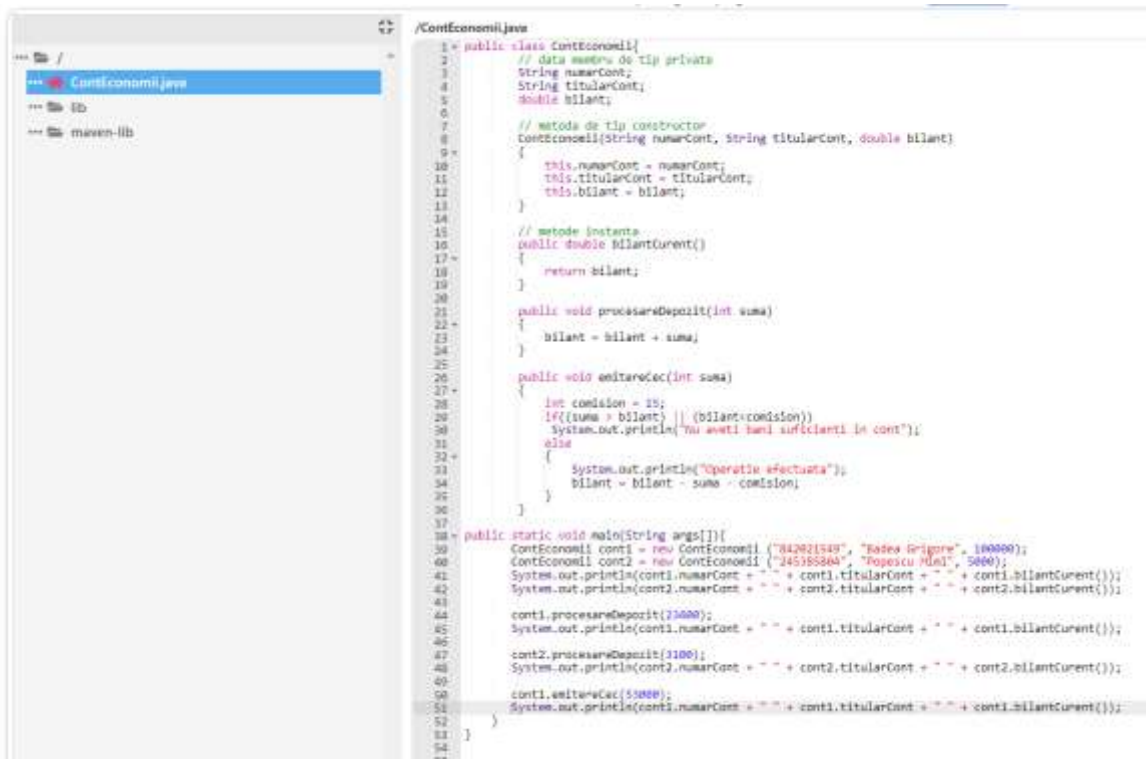
4. Clasa **ContEconomii** care are atributele numarCont, titularCont de tip String si bilant de tip int. Clasa are metode pentru afisarea bilantului current, efectuarea unei depuneri si a unei plati prin intermediul unui cec. Metoda pentru efectuarea unei depuneri va adauga o suma, specificata ca parametru 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



Probleme propuse spre rezolvare

Lab10_1: Clasa Complex. Să se definească o clasă "Complex" pentru operații cu numere complexe și să se testeze metodele implementate. Clasa va avea doi constructori astfel:

- Constructor cu doi parametri (parte reală și parte imaginară)
- Cosntrcutor fără parametri (constuctor implicit)
- Metodele necesare sunt: adunare, înmulțire, ridicare la putere naturală și afișare (șir de caractere de forma (real, imaginar)).

Să se scrie un program care crează mai multe obiecte folosind clasa **Complex** și testează operațiile de mai sus și afișări alternative.

Lab10_2: Clasa Stivă. Să se definească o clasă "Stivă" pentru stive de numere întregi, reprezentate prin vectori de numere întregi.
Datele clasei:

- Un vector de numere întregi
- Indicele elementului din vârful stivei (ultimul introdus)

Metodele clasei:

- Constructor fără parametri (dimensiunea implicită a stivei = 100)
- Constructor cu un parametru care reprezintă dimensiunea stivei
- void push(int): pune un număr întreg dat pe stivă
- int pop(): scoate elementul din vârful stivei
- boolean isEmpty(): verifică dacă stiva este vidă
- Considerați cazurile de stivă plină (la metoda push) și stivă vidă (la metoda pop).

Să se scrie un program care crează mai multe obiecte folosind clasa **Stivă** și testează operațiile de mai sus prin adăugări și eliminări succesive și afișări alternative.

Lab10_3: Clasa Mulțime. Să se definească o clasă Mulțime pentru o mulțime de numere întregi, care conține un vector numere întregi.
Metodele clasei:

- Constructor cu un parametru care reprezintă dimensiunea mulțimii
- void add(int): metoda de adăugare element, dacă nu există
- boolean contains(int): metodă care verifica dacă un număr dat se află sau nu în mulțime
- afișare: metodă care afișează mulțimea de elemente transformată în șir de caractere

Să se scrie un program care crează o mulțime folosind clasa **Mulțime** și testează operațiile de mai sus prin adăugări succesive și afișări.

Lab10_4: Clasa Carte. Definiți o clasă **Carte** pentru a gestiona informații despre cărți.
Clasa ar trebui să aibă următoarele atribute și metode:

- Atribute: titlu, autor, an publicare, număr pagini.
- Constructori:
  - Constructorul implicit (fără parametri) care inițializează toate atributele cu valori implicite.
  - Constructor care primește titlu, autor și anul publicării, iar numărul de pagini este setat implicit la 0.
  - Constructor care primește toate atributele cărții.

- Metode:
  - Metode de tip accesor pentru fiecare atribut.
  - Metodă de citire a datelor cărții de la tastatură.
  - Metodă de afișare a informațiilor despre carte.

Să se scrie un program care crează mai multe obiecte folosind clasa **Carte** și testează metodele de mai sus.

Lab10_5: Clasa Student. Definiți o clasă **Student** pentru a gestiona informații despre studenții dintr-o facultate. Clasa ar trebui să aibă următoarele atribute și metode:

1. Atribute:
   - nume: pentru numele studentului.
   - vârstă: pentru vârsta studentului.
   - facultate: pentru facultatea la care este înscris studentul.
2. Constructori:
   - Constructor implicit (fără parametri) care inițializează toate atributele cu valori implicite.
   - Constructor cu nume și vârstă.
   - Constructor cu toate atributele:
3. Metode de tip accesor:
   - Metode pentru obținerea atributelor
   - Metode adăugate:
     - Metodă pentru afișarea informațiilor
     - Metodă pentru înregistrarea studentului la cursuri

Să se scrie un program care crează mai multe obiecte folosind clasa **Student** și testează metodele de mai sus.

Lab10_6: Clasa Angajat. Definiți o clasă **Angajat** pentru a gestiona informații despre angajații dintr-o frimă. Clasa ar trebui să aibă următoarele atribute și metode:

1. Atribute:
   - nume: pentru numele angajatului.
   - salariu: pentru salariul angajatului.
   - departament: pentru departamentul în care lucrează angajatul.
2. Constructori:
   - Constructor implicit.
   - Constructor cu nume și salariu.
   - Constructor cu toate atributele:
3. Metode de tip accesor:
   - Metode pentru obținerea atributelor
   - Metode adăugate:
     - Metodă pentru afișarea informațiilor
     - Metodă pentru calculul unui bonus de performanță

Să se scrie un program care crează mai multe obiecte folosind clasa **Angajat** și testează metodele de mai sus.

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