C3

Intrare/iEsire. Tablou. Sir

- Intrare si ieşrire în Java
- Tablouri
  - definitie, declaratie, creare si utilizare
- Siruri
  - definitie, declaratie, creare si utilizare

Obiective

Dupa parcurgerea acestui curs ar trebuie sa puteti:

- ☐ intelege conceptul de paradigma orientata pe obiect
- ☐ descrie avantajele si principiile POO
- ☐ intelege si opera cu clase si obiecte
- ☐ intelege modul de abordare a scrierii unei aplicatii orientate pe obiect prin prisma caractersticilor POO.

Operatii de întrare/iEşire (I/E) în Java

Operaţiile de intrare/ieşire în Java se pot efectua în modul text sau grafic. Modul grafic presupune construirea unei interfeţe grafice prin care se colectează datele de intrare. Programarea unei interfeţe grafice necesită necesită însă timp şi instrumente de dezvoltare pentru care uneori nu avem timp sau nu este cazul să le folosim.

Citirea datelor de la tastatura (intrare în mod text)
Operaţiunea de afişare se face prin scrierea "fluxului standard de ieşire" cu utilizarea lui `System.out.println()`. Procesul invers, de citire a fluxului standard de intrare se face cu ajutorul unui obiect numit `Scanner` ce este atasat acestui flux prin `System.in`. Codul tipic este de forma:

```
Scanner intrare = new Scanner(System.in);
```

Diverse metode din clasa `Scanner` se pot folosi din acest moment pentru citirea intrării. De exemplu pentru citirea interactivă unui `int` se foloseste codul:

```
System.out.println("Valoarea intregului: ");
int i = intrare.nextInt();
```

Pentru citirea unul sir de caratere folosim codul:

```
System.out.println("Care este sirul: ");
String si = intrare.nextLine();
```

Dacă dorim să citim un singur cuvânt (delimitarea se face prin spaţii) folosim metoda `next()`.

Clasa `Scanner` este definită în pachetul `java.util.*` la fiecare utilizare trebuie folosită directiva `import` pentru a aduce clasa în aplicaţie.

Formatarea ieşirii în modul text
Afişarea unei valori numerice stocate în variabila `x` se face cu linia de cod:

```
System.out.println(x);
```

Afişarea se realizează cu numărul maxim de zecimale corespunzător preciziei tipului lui `x`. Fie liniile de cod, se observă alinierea rezultatelor este o problemă:

```
System.out.println(15./7.);
System.out.print(15.f/7.f);
```

afişează rezultatele:

```
2.142857142857143
2.142857
```

Începând cu JDK 5.0 afişarea se poate face şi folosind cunoscita funcţie `printf()` din biblioteca C. Linia de cod:

```
System.out.printf("\n%9.7f",15.f/7.f);
```

afişează rezultatul:

```
2.1428571
```

Caracterele `\n` realizează afişarea valorii pe o linie nouă pe care `9` este laţimea maximă a caracterelor din valoarea afişate iar `7` precizia. Secvenţa între ghilimele conţine specificatori de format, fiecare începe cu caratcterul % şi trebuie să aibă un argumet corespunzator. Caracterul de conversie prin care se termină specificatorul de format defineste tipul de formatare a valorii de afişat: `f` pentru `float`, `s` pentru `String`, `d` pentru intregi.

Modul grafic (predefinit în swing)
Începând cu JDK 5.0 se pot utiliza ferestre de dialog pentru citirea datelor. Iată un exemplu:

```
String intrare = JOptionPane.showInputDialog(sirAfisat);
```

Valorea întoarsă este un de tipul şir (`String`) conţinând caracterele introduse de utilizator. Citirea de valori numerice necesită un cod adiţional prin care şirul să fie convertit la un număr (conversia presupune trecerea de la reprezentarea internă de şir la cea specifică unui anumit tip numeric).

```
String intrare = JoptionPane.showInputDialog("Pretul: ");
double pret = Double.parseDouble();
```

Mai sus linia `Double.parseDouble()` converteşte şirul "123.7" la numărul de tip `double` 123.7, presupunând ca el a fost introdus de la tastatură.

Clasa `JOptionPane` este definită în pachetul `javax.swing` care trebuie importată în aplicaţie prin linia, scrisă la început de cod:

```
import javax.swing.*;
```

Afişarea se realizează cu `JoptionPane.showMessageDialog` iar în codurile ce urmează se prezintă exemple de utilizare ce clarifică modul de utilizare.

Tablouri în Java

Tabloul este o colecţie de date, numite şi elemente ale tabloului, identice ca tip - tipul se mai numeşte şi tip de bază - ce pot fi identificate unic prin indice. Numărul de dimensiuni ale tabloului depinde de limbaj, dar în general, nu este limitat. Se zice că un tablou este multidimensional dacă foloseşt mai mulţi indici pentru accesarea unui element de tablou.

Mai sus se prezintă modul de reprezentate a două tablouri, unul de întregi (tipul de bază este int) şi unul de obiecte (tipul de bază este Coordonate).

O variabilă de tip primitiv - un scalar - poate fi privită ca un tablou cu dimensiunea zero. Un tablou cu o singură dimensiune este cunoscut sub numele de "vector", iar unul cu două dimensiuni sub denumirea de "matrice". În majoritatea limbajelor imperative, o referinţă la un element de tablou se scrie sub forma a[i,j] unde, a este numele tabloului, iar i şi j sunt indexuri. Elementele tabloului sunt de obicei stocate în locaţii de memorie consecutive. Limbajele diferă de obicei prin metoda de stocare a datelor pe rânduri continue sau pe coloane continue.

Tablourile sunt folosite pentru stocarea unui grup de valori care primesc un singur nume şi care se vor accesa într-o ordine imprevizibilă (de exemplu, o structură de dată numită listă se pretează mult mai bine pentru stocarea unor valori care se accesează secvenţial).

Etapele lucrului cu tablouri

- **declararea tabloului:**
  `int []a;`
- **crearea unui obiect tablou:**
  `a = new int[4];`
- **iniţializarea tabloului:**
  `int[] a = {1, 7, 23, 1};`
- **indicele ∈ [0, nr_elem-1] > altfel apare excepţia "array index out of bounds"**
- **accesul la un element:**
  `a[indice]`

Pentru a putea utiliza un tablou cu tipul de bază dintre cele primitive trebuie să:

1. declarăm o variabilă de tipul tablou având `nume_tablou`, ce v-a stoca o referinţă către tablou prin: `tip [] nume_tablou;` sau `tip nume_tablou [];` în acest moment `nume_tablou` se iniţializează cu valoarea specială `null`; această valoare se modifică la adresa tabloului după utilizarea operatorului `new`;
2. creăm tabloul folosind operatorul `new` specificând lungimea acestuia (numărul maxim de elemente pe care îl poate stoca); lungimea trebuie să fie o valoare întreagă de tipul `int` (constantă sau expresie care se calculează în momentul rulării aplicaţiei)
3. iniţializăm tabloul cu valori diferite de cele cu care acesta se iniţializaeză implicit de către Java.

Valorile de iniţializare implicită pentru tablouri:

`char`	`boolean`	`byte`, `short`, `int`, `long`	`float`, `double`
'\u0000' - Unicode 0000 (Java foloseşte codificarea Unicode)	false	0	0

Deşi un tabloul este un obiect, nu există o clasă de tipul tablou. Java creează automat o clasă pentru fiecare tablou de tipuri primitive sau de clase. Pentru a afla numărul de elemente dintr-un tablou folosim notaţia `nume_tablou .length .`

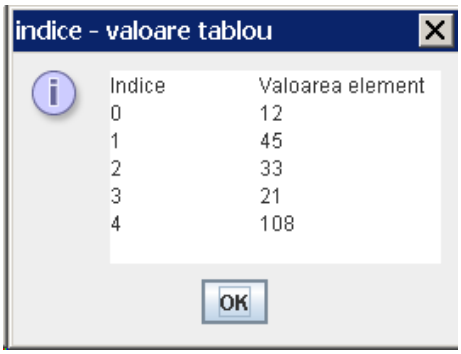
Aplicaţia 1 - tablou de tipul int, citire şi afişare

Aplicaţia următoare exemplifică modul de citire şi de afişare a elemtelor unui taboul. Tabloul are numele a şi numărul maxim de elemente fixat la 5. Valorile sunt citite de la tastatură iar afişarea se face prin două metode.

```
import java.util.Scanner;
import javax.swing.*;

public class TablouPrimitive {

    public static void main(String[] args) {
        final int NRELEM = 5; //lungimea tabloului
        int [] a; //declararea tabloului
        Scanner intrare;
        String iesire = "Indice\tValoarea element\n";

        intrare = new Scanner(System.in);
        a = new int[NRELEM];  //crearea tabloului;

        //citire elemente tablou
        for(int i = 0; i < a.length;++i) {
            System.out.print("a[" + i +"] = ");
            a[i]= intrare.nextInt();

            //pt. cea de a 2-a metoda de afisare
            iesire+= i +"\t"+ a[i] + "\n";
        }

        //metoda 1: afisare simpla de elemente tablou
        for(int i = 0; i < a.length;++i)
            System.out.println(a[i]);

        //metoda 2: afisare cu fereastra de dialog
        JTextArea fereastraiesire = new JTextArea();
        fereastraiesire.setText(iesire);
           JOptionPane.showMessageDialog(null,fereastraiesire,"indice - valoare
tablou", JOptionPane.INFORMATION_MESSAGE);
      }
}
```

Rezultate:

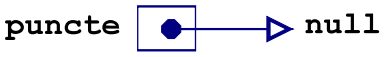
```
a[0] = 12
a[1] = 45
a[2] = 33
a[3] = 21
a[4] = 108
12
45
33
21
108
```



Etapele lucrului cu tablouri de obiecte

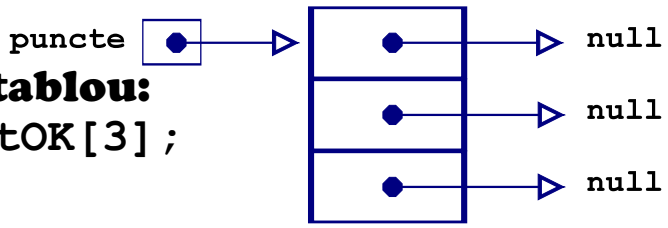
- **declararea tabloului:**

```
PunctOK puncte[];
//sau PunctOK[] puncte;
```



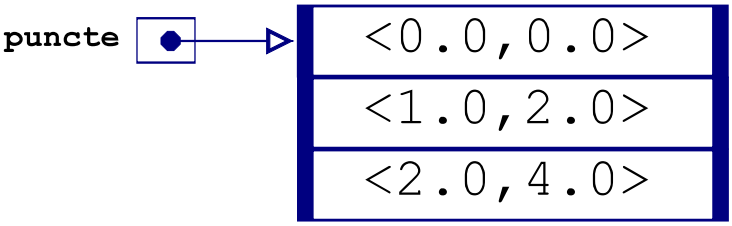
- **crearea unui obiect tablou:**

```
puncte = new PunctOK[3];
```



- **inițializarea obiectelor din tablou:**

```
puncte[i] = new PunctOK(i,2*i);
```



În situaţia lucrului cu tablouri de obiecte, etapele sunt indentice cu cele descrise la tablourile de primitive, cu excecţia că aici iniţializarea elementelor de tablou este obligatorie şi nu opţională.

Aplicaţia 2 - tablou de obiecte, iniţializare şi afişare

Aplicaţia are la bază clasa `PuncteOK`. Se creează un tablou de obiecte `PuncteOK` cu numele `puncte`. Iniţializarea obiectelor din tablou se face în ciclul `for` cu linia `puncte[i] = new PunctOK(i,2*i);`. Rezultatele afişate se obţin prin iniţializarea fiecărui obiect `PuncteOK` cu x = i şi y =2*i.

```
public class TablouObiecte {
    public static void main(String[] args) {

        //declarare tablou
        PunctOK puncte[];

        //creare tablou de obiecte
        puncte = new PunctOK[3];

        for(int i = 0; i < puncte.length;++i) {
            //initializarea obiectelor din tablou
            puncte[i] = new PunctOK(i,2*i);
        }

        //afisarea elementelor de tablou
        for(int i = 0; i < puncte.length;++i)
            System.out.println(puncte[i]);
    }
}
```

Rezultate:

```
<0.0,0.0>
<1.0,2.0>
<2.0,4.0>
```

Excepții întâlnite la tablouri

- **utilizarea unei indice în afara domeniului permis generează excepția ArrayIndexOutOfBoundsException:**

```
int [] a = new int[7];
System.out.println(a[13]);
```

(în exemplul anterior indicele poate fi în domeniu 0 - 6 = 7-1)

- **accesarea unor membri unui element obiect care încă nu a fost inițializat generează excepția NullPointerException:**

```
puncte = new PunctOK[3];
System.out.println(puncte[0].x());
```

(în exemplul anterior 0 $\in$ [0, 2], dar nu s-a folosit `new punct[0]` **pentru crearea unui obiect** `PunctOK`**)**

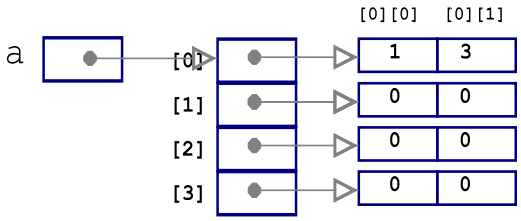
Tablouri multidimensionale

- **un tablou multidimensional este un tablou de tablouri;**
- **pentru accesarea elementelor de folosesc mai mulți indici;**
- **declararea şi crearea unui tablou cu 2 dimensiuni (matrice):**

```
tip [][] numetablou = new tip[nl][nc];

int [][] a = new int[4][2];
a[0][0] = 1;
a[0][1]=3;
```

- **inițializare:**

```
int[][] a =
    {
        {1,3},
        {0,0},
        {0,0},
        {0,0}
    };
```



- **declarare şi inițializare fără specuficarea numărului de coloane:**

```
tip [][] numetablou = new tip[nl][];

numetablou[il] = new tip[nc];
```

Aplicaţia 1 - crearea unui tablou cu număr "variabil" de coloane

În aplicaţia care urmează lungimea liniilor din matrice este diferită, adică variază de la linie la linie.

```
public class TabMulti {
    public static void main(String[] args) {
        int tab2d [][] = new int [4][];
        tab2d[0] = new int[5];
        tab2d[1] = new int[2];
        tab2d[2] = new int[4];
        tab2d[3] = new int[7];

        int k = 0;

        for(int i=0; i < tab2d.length; ++i)
            for(int j=0; j < tab2d[i].length; ++j)
                tab2d[i][j] = k++;


        for(int i=0; i < tab2d.length; ++i) {
            for(int j=0; j < tab2d[i].length; ++j)
                System.out.print(tab2d[i][j]+" ");
            System.out.println();
        }
    }
}
```

Rezultate:

```
0 1 2 3 4
5 6
7 8 9 10
11 12 13 14 15 16 17
```

Şiruri de caractere în Java

Şir de caractere = o secvenţa de caractere de lungime arbitrară

Java NU are implementat un tip primitiv şir de caractere

`String` **clasă Java predefinită în pachetul java.lang.String pentru manipularea şirurilor de caractere**

```
String s; //declaratia unei variabile şir
String sal = ”Salut”; //decl. cu iniţial.
```

Operatori: atribuirea (=)
concatenarea (+)
atribuirea compusă (+=).

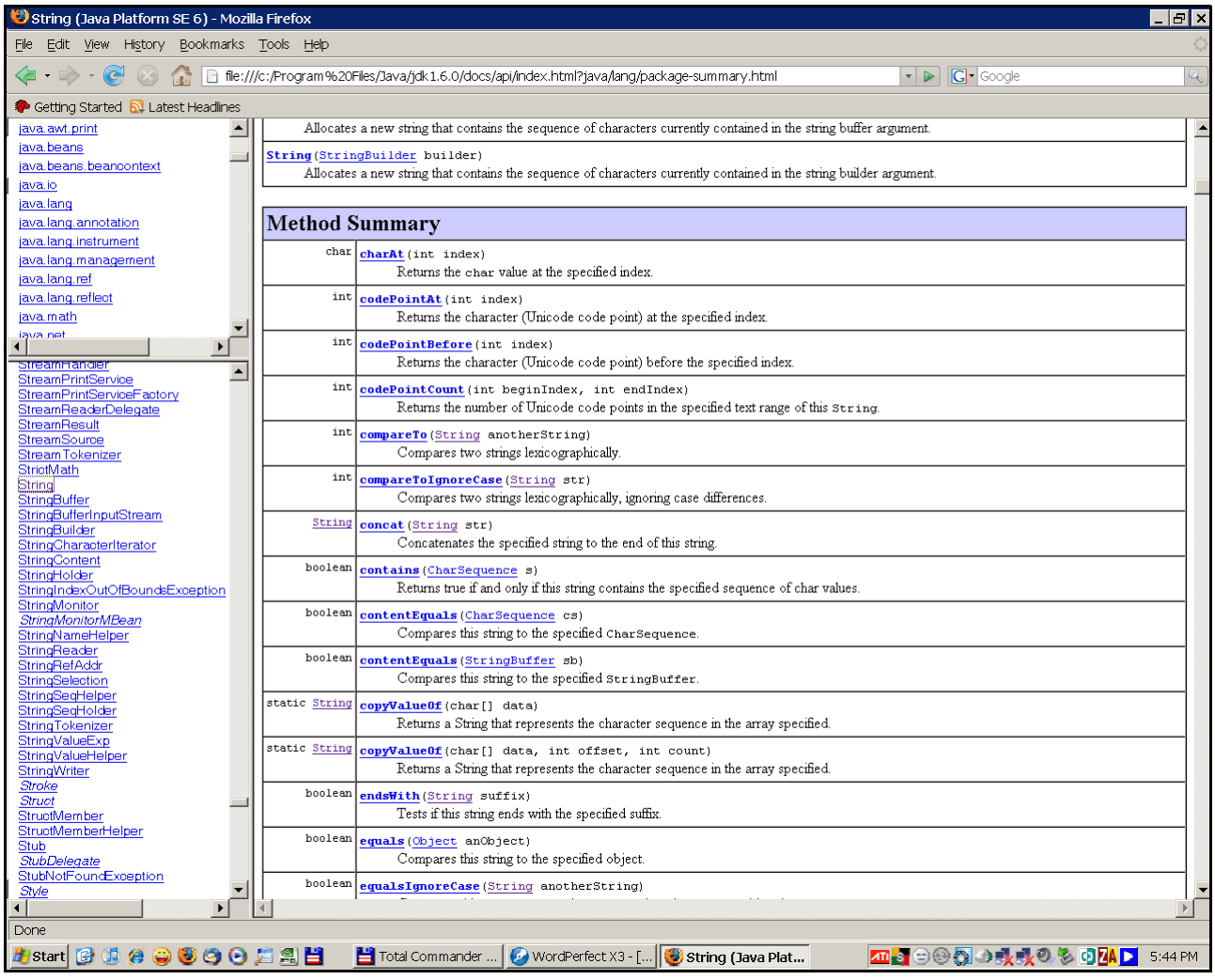
```
public class Siruri {
    public static void main(String[] args) {
        String s1, s2;
        String s3 = "Vasile"; //Vasile este un literal

        s2 ="Ion"; //Ion este un literal

        //concatenare
        s1 = s2 + s2;

        //lungimea unui sir de caractere
       System.out.println("Lungimea sirului: " +s1 +" este de " +s1.length()+"
caractere");

        //subsir
        System.out.println("Un subsir: " + s1.substring(0,3));

        //editarea sirurilor
        s1 = s1.substring(0,3) + "-" + s1.substring(3,s1.length());
        System.out.println("Editarea: " + s1);

        //testarea egalitatii sirurilor
        System.out.println("s1: " + s1+"\ns2: "+s2);
        System.out.println("Egalitatea lui s1 cu s2: " + s1.equals(s2));
        System.out.println("Egalitatea lui Ion cu s2: " + "Ion".equals(s2));

        /* comparatia sirurilor
```

```
            * intoarce < 0 daca s1 vine inainte de s2 in dictionar
            * intoarce 0 daca s1 si s2 sunt egale
            * intoarce > 0 daca s1 este dupa s2 in dictionar
            */
            System.out.println("Compararea lui s1 cu s2: " + s1.compareTo(s2));
    }
}
```

Rezultate:

```
Lungimea sirului: IonIon este de 6 caractere
Un subsir: Ion
Editarea: Ion-Ion
s1: Ion-Ion
s2: Ion
Egalitatea lui s1 cu s2: false
Egalitatea lui Ion cu s2: true
Compararea lui s1 cu s2: 4
```

Clasa `String` are mai bine de 50 de metode. Inspectarea acestora se face cel mai simplu utilizând documentaţia JDK.



Compilatorul Java aloca spaţiu pentru literalii şir în memorie iar operatorul de atribuire va stoca adresa respectivă în variabila şir. Din acest motiv, testarea egalităţii a două şiruri nu se poate face cu operatorul

==, deoarece el va compara adresele la ca sunt stocate cele două şiruri în RAM şi nu conţinutul respectivelor locaţii.

Clasa `String` nu are o metodă pentru modificarea unui caracter al şirului, se zice că obiectele clasei şir sunt imuabile (immutable) - stabile, de neschimbat. Pentru modficarea conţinutului unei variabile şir trebuie să creăm un şir nou.

C3- Întrebari

1. Care este codul Java pentru afişarea unui sir in mod text pe ecran si intr-o fereastra de ieşire predefinita?
2. Care este codul Java pentru citirea unui sir in mod text pe ecran sau dintr-o fereastra de ieşire predefinita?
3. Când se utilizează tablourile in locul variabilelor de tip primitiv?
4. Ce excepţii pot apare la manipularea greşita a elementelor de tablou?
5. Ce metode cunoaşteţi pentru manipularea şirurilor?

BIBLIOGRAFIE

1. http://www.oracle.com/technetwork/java/javase/documentation/index.html

2. http://docs.oracle.com/javase/6/docs/

3. Stefan Tanasa, Cristian Olaru, Stefan Andrei, Java de la 0 la expert, Polirom, 2003, ISBN: 973-681-201-4.

4. Herber Schild, Java 2 - The Complete Reference, Fourth Edition, Osborne, 2001, ISBN: 0-07-213084-9.

5. Deitel H.M., Deitel P. J., Java - How to programm, Fith Edition, Prentice Hall, 2003, ISBN: 0-13-120236-7.

6. http://www.east.utcluj.ro/mb/mep/antal/downloads.html