Laborator Nr. 5:
Structuri de date 1

Întocmit de: Dobrinaş Alexandra

Îndrumător: Asist. Drd. Danciu Gabriel

October 28, 2011

I. NOŢIUNI TEORETICE

A. Array

Un şir de date sau un *array* este o structură ce conţine valori multiple de acelaşi tip de data. Lungimea unui şir de date se stabileşte la crearea array-ului şi va fi fixa pe întreaga existenţă a acestuia. De regulă, primul element se află pe poziţia 0, iar ultimul pe poziţia $n-1$, unde n reprezintă numărul de elemente din şir.

1. Declararea unui şir de date

În Java, declararea unui array se face astfel:

```
tip_de_date [] nume_sir;
```

unde, *tip_de_date* reprezintă tipul de date atribuit şirului (*int, float, char, String*, sau orice tip referinţă), iar *nume_sir* reprezintă numele variabilei în care se stochează şirul.

Odată ce şirul de date a fost declarat, acesta se poate instanţia. Pentru instaţiere se foloseşte operatorul *new*.

De exemplu, pentru un şir de date de tipul *int*, cu 3 elemente, declararea şi instanţierea acestuia se face astfel:

```
int [] s;
s = new int[3];
```

sau:

```
int [] s=new int[3]
```

Setarea valorilor pentru elementele şirului s creat, se face astfel:

```
s[0]=14;
s[1]=2;
s[2]=12;
```

sau, elementelor li se pot atribui valori încă de la instanţiere:

```
int[] s = {14,2,12};
```

Pentru atribuirea de valori ale unui şir, se pot folosii şi instrucţiunile repetitive. De exemplu, în cazul În care se citeşte un şir de la tastatură, atribuirea valorilor se poate face astfel:

```
1  BufferReader stdin = new BufferReader (new InputStreamReader(System.in));
2  int [] s = new int[5];
3  for(int i=0;i<5;i++)
4  {
5     s[i]=Integer.parseInt(stdin.readLine());
6  }
```

Declararea unui array de un tip de date referinţă se face în acelaşi mod ca şi pentru şirurile de tipul de date primitive. Doar că un astfel de şir va avea ca elemente obiectele unei clase. Astfel că, dacă se dă o clasă cu numele *Country*, un şir de elemente din această clasă se declară astfel:

```
1  class Country
2  {
3     String name;
4     long population;
5  }
6
7  public class CountryDemo {
8     public static void main(String[] args)
9     {
10
```

```
11          Country[] sir = new Country[2];
12          sir[0].name = "Italia";
13          sir[0].population = 6000000000001;
14
15          sir[1].name = "Franta";
16          sir[1].population = 6500000000001;
17      }
18  }
```

2. Declararea şirurilor multidimensionale

Deoarece un şir de date poate conţine referinţe către alte obiecte, este posibil ca un şir să conţină referinţe şi catre alte şiruri. În acest caz, se spune că se folosesc şiruri multidimensionale. Cele mai comune şiruri multimensionale sunt matricile. Exemplu de sir multidimensional:

```
int matrix[][] = new int[3][2];
matrix[0][1] = 12;
```

În acest caz se dă un şir bidimensional, astfel: 3 şiruri în care sunt reţinute şiruri de câte două elemente de tip int. Accesarea elementelor unui astfel de şir se face prin precizarea tuturor indicilor (în cazul acesta sunt 2 indici).

Nu este obligatoriu ca toate subşirurile să aibă aceeaşi dimensiune. În exemplul următor este prezentat un şir multidimensional în care subşirurile au dimensiuni diferite:

```
1  float sir[][] = new float[4][];
2
3  sir[0] = new float[5];
4  sir[1] = new float[]{2.3f,5,6.7f,11};
5  sir[2] = new float[]{1f,4.2f,7f,4.1f,10f,9f};
6  sir[3] = new float[20];
```

B. Clasa Vector

Clasa Vector face parte din pachetul *java.util* şi reprezintă o clasă specială pentru lucrul cu şirurile de elemente. Cele mai uzuale metode folosite pentru declararea şi crearea unui astfel de şir sunt:

```
Vector v = new Vector();
//sau
Vector v = new Vector(5);//se creeaza un vector de 5 elemente
```

1. Metode şi operaţii specifice clasei Vector

Deoarece manipularea elementelor unui şir este o operaţie ce necesită multă muncă, clasa *Vector* are definite o serie de metode şi operaţii ce pot fi de folos:

Metoda	Operaţia specifică
add()	Adauga un element intr-un vector.
addAll()	Adauga o colectie de elemente intr-un vector.
addElement()	asemenea metodei add().
capacity()	Returneaza capacitatea adica marimea sirului intern.
clear()	sterge toate elementele unui vector.
clone()	Creeaza o clona a vectorului.
contains()	Verifica daca un vector contine un element.
containsAll()	Verifica daca un vector contine o colectie.
copyInto()	Copiaza elementele unui vector intr-un array.
elementAt()	Returneaza elementul de la pozitia specificata.
elements()	Returneaza un obiect al vectorului care permite vizitarea tuturor cheilor vectorului.
ensureCapacity()	Verifica si se asigura ca marimea buffer-ului intern sa fie de o anumita marime.
equals()	verifica egalitatea cu un obiect.
firstElement()	returneaza primul element.
get()	returneaza un element de la o anumita pozitie.
indexOf()	cauta prima aparitie a unui element in sir.
insertElementAt()	Insereaza un element in sir.
isEmpty()	verifica daca un vector este gol.
iterator()	returneaza un iterator, adica un obiect ce permite vizitarea elementelor din sir.
lastElement()	Returneaza ultimul element din sir.
lastIndexOf()	Cauta pozitia ultimului element din sir care este egal cu obiectul specificat.
listIterator()	Returneaza un obiect care permite ca toate elementele sa fie vizitate secvential.
remove()	Sterge un anumit element din vector.
removeAll()	Sterge toate elementele specificate in colectia data ca parametru.
removeAllElements()	Sterge toate elementele din sir si seteaza marimea acestui cu zero.
set()	Schimba un element de la o anumita pozitie.
setElementAt()	Acelasi lucru ca si set.
setSize()	modifica marimea buffer-ului intern.
size()	returneaza numarul de elemente din sir.
subList()	returneaza o sectiune din sir.
toArray()	returneaza elementele vectorului ca array.
trimToSize()	taie o portiune din sir, astfel ca el sa ramana de marimea specificata.

C. Clasa Stack

Stiva este o structură de date de tipul *LIFO (Last In, First Out)*.

Clasa *Stack* este considerată a fi ,,copilul" clasei *Vector*. însă are definite o serie de operaţii specifice stivelor.

- Adăugarea elementelor.

  Aceasta operatiune se face prin metoda push():

`public Object push(Object element)`

- Ştergerea unui element Aceasta operatiune se va face prin apelul functiei pop():

  `public Object pop()`

- Verificare dacă stiva este goală

  `public boolean empty()`

- Preluarea elementului din vârful stivei

  `public Object peek()`

- Căutarea unui element în stivă

  `public int search(Object element)`

Exemplu de utilizare a clasei *Stack*:

```
1  import java.util.Stack;
2  public class Stiva
3  {
4     public static void main (String args[])
5     {
6        Stack s = new Stack();
7        s.push("Primul element");
8        s.push("Al doilea element");
9        s.push("Al treilea element");
10       System.out.println("Next: " + s.peek());
11
12       s.push("Al patrulea element");
13       System.out.println(s.pop());
14       s.push("Al cincilea element");
15       s.push("Al saselea element");
16       System.out.println(s);
17
18       int count = s.search("Al doilea element");
19       while (count != -1 && count > 1)
20       {
21          s.pop();
22          count--;
23       }
24
25       System.out.println(s.pop());
26
27       System.out.println(s.empty());
28       System.out.println(s);
29    }
30 }
```

II. TEME DE LABORATOR

1. Se consideră un şir de elemente ce conţine n numere reale. Se spune că două elemente ale sale formează o ,,pereche în dezordine" dacă sunt îndeplinite simultan condiţiile:

   - $i < j$
   - $a[i] > a[j], unde 1 \leq i < n$ şi $1 < j \leq n$

   Să se creeze un program care afişează perechile în dezordine din şir şi numărul lor. Exemplu: Pentru n=4 şi şirul (1, 13,2,4), se va afişa: 13 2 13 4 2

2. Se consideră două tablouri bidimensionale de dimensiuni identice (nxm). Să se afişeze transpusa matricei sumă. Transpusa unei matrice se obţine prin schimbarea liniilor cu coloanele.

3. Determinaţi suma maximă care se poate forma cu m numere distincte dintr-un vector ce conţine n valori întregi. Dacă şirul conţine mai puţin de m valori distincte se va afişa mesajul *Imposibil*. *Pentru rezolvarea problemei se va folosi clasa Vector.*

4. Folosind o stivă, realizaţi şi afişaţi programul dumneavoastră dintr-o zi, pe ore. Eliminaţi acţiunile pe care le efectuaţi până la ora 14 şi afişaţi ceea ce urmează să faceţi la ora 15.