Διανύσματα - Vectors

Η κλάση Vector μας βοηθά να δημιουργήσουμε δυναμικούς πίνακες, δηλαδή πίνακες των οποίων το μέγεθος μπορεί να αλλάξει δυναμικά. Ένας τέτοιος δυναμικός πίνακας αποθηκεύει μια λίστα αναφορών σε **αντικείμενα.** Η αλλαγή του μεγέθους του επιτυγχάνεται με τα πεδία *"capacity"* και *"capacityIncrement"*. Δηλαδή, όταν ορίζεται ένα Vector στην ουσία δημιουργείται ένας πίνακας αντικειμένων με αρχική χωρητικότητα *("initialCapacity")*. Όταν γεμίζει αυτός ο πίνακας, τότε αυξάνει το μέγεθος του buffer σύμφωνα με την τιμή του πεδίου capacityIncrement.

Σε ένα Vector, δεν μπορεί να αποθηκευτεί με άμεσο τρόπο ένας βασικός τύπος της Java, αλλά με έμμεσο τρόπο μόνο αν τοποθετηθεί σε ένα αντικείμενο. Για παράδειγμα, για την αποθήκευση ενός ακέραιου αριθμού πρέπει να δημιουργήσουμε ένα αντικείμενο με τη χρήση της κλάσης Integer (π.χ. new Integer(10)). Βέβαια καλύτερος τρόπος είναι η δημιουργία δικής μας κλάσης, ώστε να μπορούμε να αλλάξουμε την τιμή του βασικού τύπου.

Για να ορίσουμε και να χρησιμοποιήσουμε ένα Vector πρέπει να εισάγουμε πρώτα την εντολή:

import java.util.Vector; ή **import java.util.*;**

Οι δομητές του Vector

1) *Δομητής χωρίς παράμετρο,* κατασκευάζει ένα κενό Vector:

```
Vector v;                 //ορισμός μεταβλητής τύπου Vector
v = new Vector();         // ορισμός ενός κενού Vector αντικειμένων
```

ή

```
Vector v = new Vector();
```

2) *Δομητής με παράμετρο,* κατασκευάζει ένα Vector με αρχικό μέγεθος:

Vector v = new Vector(200); ***// initialCapacity = 200***

3) *Δομητής με παράμετρους,* κατασκευάζει ένα Vector με αρχικό μέγεθος και βήμα αύξησης:

Vector v = new Vector(200, 1); ***// initialCapacity = 200, capacityIncrement =1***

4) Δομητής για δημιουργία Vector από Strings.

Vector<String> v = new Vector<String>();

Οι κυριότερες μέθοδοι επεξεργασίας των στοιχείων ενός Vector

Μέθοδος	Περιγραφή
v.add(o)	Προσθέτει ένα αντικείμενο **o** στο τέλος του Vector v.
v.add(i, o)	Εισάγει ένα αντικείμενο **o** στη θέση **i**, μετακινώντας τα στοιχεία προς τα επάνω.
v.addElement(o)	Προσθέτει ένα αντικείμενο **o,** αυξάνοντας το μέγεθος κατά 1.
v.capacity()	Επιστρέφει την τρέχουσα χωρητικότητα του Vector v.
v.clear()	Διαγράφει όλα τα στοιχεία από ένα Vector v.
v.clone()	Επιστρέφει ένα αντίγραφο του Vector.
v.contains(o)	Επιστρέφει true να ένα Vector v περιέχει το αντικείμενο **o.**
v.elementAt(i)	Επιστρέφει το στοιχείο στη συγκεκριμένη θέση **i**.
v.firstElement()	Επιστρέφει το πρώτο στοιχείο.
v.get(i)	Επιστρέφει το αντικείμενο στη θέση **i**.
v.isEmpty()	Ελέγχει αν ένα Vector έχει στοιχεία ή όχι.
v.lastElement()	Επιστρέφει το τελευταίο στοιχείο.
v.remove(i)	Διαγράφει το στοιχείο στη θέση **i**, μετακινώντας τα στοιχεία προς τα κάτω.
v.remove(o)	Διαγράφει το πρώτο στοιχείο (αντικείμενο o) που θα βρει στο Vector.
v.removeElementAt(i)	Διαγράφει το στοιχείο στη θέση **i.**
v.removeRange(i, j)	Διαγράφει από τη θέση **i** έως τη θέση **j**.
v.removeAllElements()	Διαγράφει όλα τα στοιχεία του Vector.
v.set(i, o)	Αντικαθιστά το αντικείμενο **o** στη θέση **i**.
v.setElementAt(o, i)	Τοποθετεί το αντικείμενο o στη θέση **i**.
v.setSize(i)	Ορίζει νέο μέγεθος του Vector v.
v.size()	Επιστρέφει τον αριθμό των στοιχείων του Vector v.
v.toString()	Επιστρέφει μία συμβολοσειρά που αναπαριστά το στοιχεία του Vector (τα στοιχεία σε μορφή – Strings).

Η είσοδος των στοιχείων

Η χρήση της μεθόδου **addElement()** μπορεί να χρησιμοποιηθεί για την είσοδο αντικειμένων βασικών τύπων. Παραδείγματα:

v.addElement(new Integer(24));

v.addElement(new Float(3.210));

Για να τοποθετήσουμε ένα στοιχείο σε κάποια θέση του Vector αλλά και να λάβουμε ένα στοιχείο του από συγκεκριμένη θέση, χρησιμοποιούμε επίσης τις μεθόδους **setElementAt()** και **elementAt()**.

Ο παρακάτω πίνακας δείχνει τις διαφορές στο 'στυλ' χειρισμού των Vectors σε σχέση με τους πίνακες.

array	vector
s = a[i];	s = v.elementAt(i);
a[i] = s;	v.setElementAt(s, i);

Ας δούμε ένα μικρό πρόγραμμα που εισάγει αριθμητικά αντικείμενα στο Vector. Με την εντολή: **System.out.println(v)**, θα εμφανίσουμε τα στοιχεία του Vector.

```
import java.util.*;
class VectorTest {
  public static void main(String[] args) {
        Vector v = new Vector();
        v.addElement(new Integer(24));
        v.addElement(new Float(3.210));
        System.out.println(v);
  }
}
```

Το αποτέλεσμα:

Ένας άλλος τρόπος εμφάνισης των στοιχείων του Vector (ο πιο συνηθισμένος…) είναι η χρήση της εντολής **for**:

```
for (int i=0; i<v.size(); i++) {
   System.out.println("v[" + i + "] = " + v.elementAt(i));
}
```

(προσέξτε τη χρήση της size())

Παράδειγμα με την εντολή for

```
import java.util.*;
class VectorTest3 {
   public static void main(String[] args) {
          Vector v = new Vector(10);
          v.addElement(new Integer(24));
                    v.addElement(new Float(3.210));
         for (int i=0; i<v.size(); i++) {
                    System.out.println("v[" + i + "] = " + v.elementAt(i));
    }
   }
}
```

Το αποτέλεσμα:

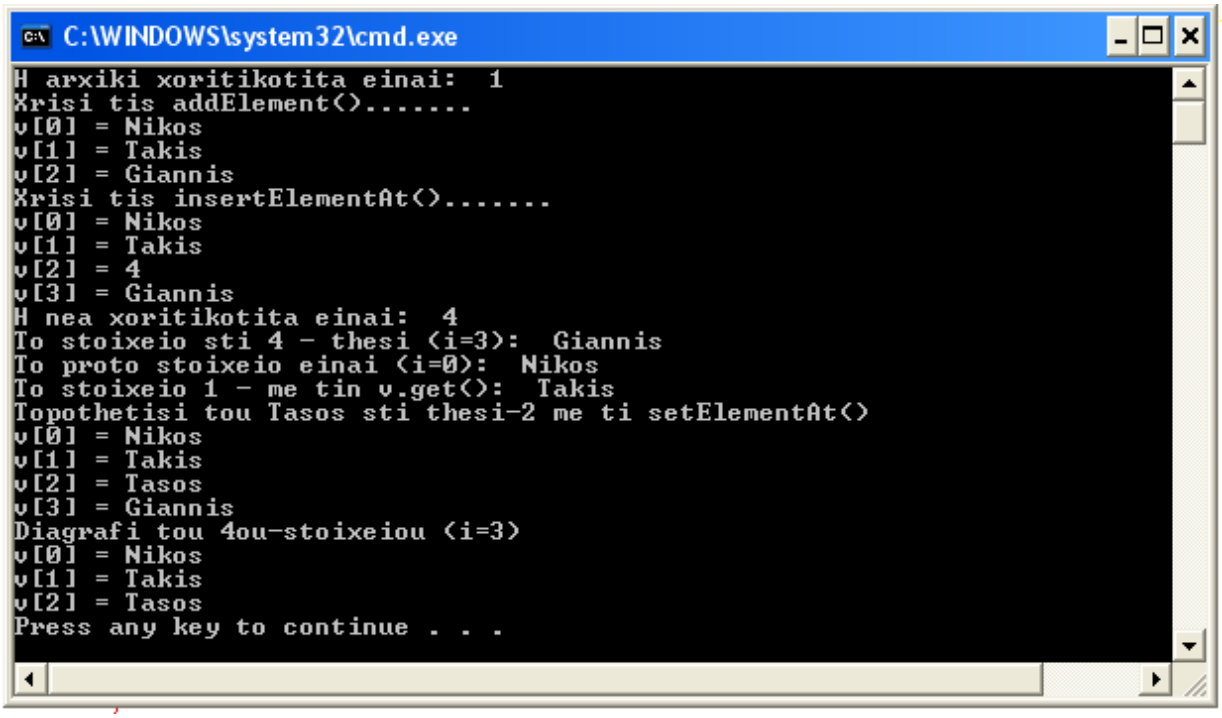
Παράδειγμα:

Στο παρακάτω παράδειγμα θα δούμε την υλοποίηση μερικών από τις σημαντικότερες μεθόδους της Vector.

```
import java.util.*;
class VectorTest1 {
   public static void main(String[] args) {
          Vector v = new Vector(1);
          System.out.println("H arxiki xoritikotita einai:  " + v.capacity());
          String s1 = "Nikos";
          String s2 = "Takis";
          String s3 = "Giannis";
          Integer t = new Integer(4);

          System.out.println("Xrisi tis addElement().....");
          v. addElement(s1);
          v. addElement(s2);
          v. addElement(s3);

          for (int i=0; i<v.size(); i++) {
```

```
            System.out.println("v[" + i + "] = " + v.elementAt(i));
        }
        System.out.println("Xrisi tis insertElementAt().....");
        v.insertElementAt(t, 2);
        for (int i=0; i<v.size(); i++) {
                System.out.println("v[" + i + "] = " + v.elementAt(i));
        }
        System.out.println("H nea xoritikotita einai:  " + v.capacity());

        System.out.println("To stoixeio sti 4 - thesi (i=3):  " + v.elementAt(3));
        System.out.println("To proto stoixeio einai (i=0):  " + v.firstElement());
        System.out.println("To stoixeio 1 - me tin v.get():  " + v.get(1));
        System.out.println("Topothetisi tou Tasos sti thesi-2 me ti setElementAt() " );
        v.setElementAt("Tasos", 2);
        for (int i=0; i<v.size(); i++) {
                System.out.println("v[" + i + "] = " + v.elementAt(i));
        }
        System.out.println("Diagrafi tou 4ou-stoixeiou (i=3) ");
        v.remove(3);
        for (int i=0; i<v.size(); i++) {
           System.out.println("v[" + i + "] = " + v.elementAt(i));
        }
    }
}
```

Τα αποτελέσματα:



Άλλοι τρόποι προσπέλασης των στοιχείων του Vector

Μπορεί η εντολή - **for** να χρησιμοποιείται περισσότερο για την προσπέλαση των στοιχείων ενός Vector, αλλά υπάρχουν και δύο άλλοι τρόποι προσπέλασης που θα αναφέρουμε εδώ. Ο πρώτος τρόπος είναι με τη χρήση ενός **ListIterator**. Αυτός ο τρόπος αντιστοιχεί στη χρήση της εντολής **foreach** άλλων γλωσσών προγραμματισμού για την προσπέλαση στοιχείων συλλογών. Στο παρακάτω παράδειγμα θα εμφανίσουμε όλα τα στοιχεία ενός Vector (αντικειμένων Strings) χρησιμοποιώντας δύο μεθόδους: την **hasNext()**, που επιστρέφει true αν υπάρχουν περισσότερα στοιχεία, και την **next()**, που επιστρέφει το επόμενο στοιχείο.

```
ListIterator it = v.listIterator();
while (it.hasNext())
{    System.out.println((String)it.next()); }
```

Ο δεύτερος τρόπος παρέχεται από τη **διεπαφή Enumeration**:

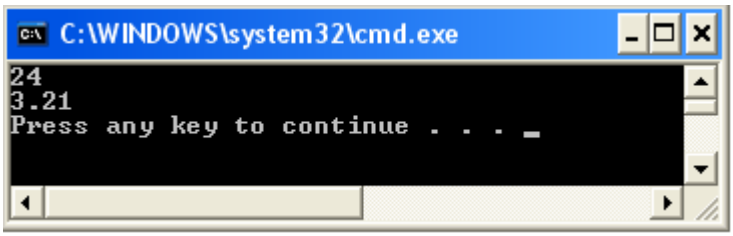
```
Enumeration e=vector.elements();
while(e.hasMoreElements()) {
  System.out.println("The elements are: " +  e.nextElement());
}
```

Παραδείγματα

Στο πρώτο παράδειγμα αυτό θα δημιουργήσουμε το Vector όπως και στο πρώτο παράδειγμα αλλά η εμφάνιση των στοιχείων θα γίνει με τη χρήση της **ListIterator**.

```
import java.util.*;
class VectorTest3 {
  public static void main(String[] args) {

        Vector v = new Vector(10);
       v.addElement(new Integer(24));
       v.addElement(new Float(3.210));

        ListIterator it = v.listIterator();
        while (it.hasNext())
        {
          System.out.println(it.next());
        }
  }
}
```

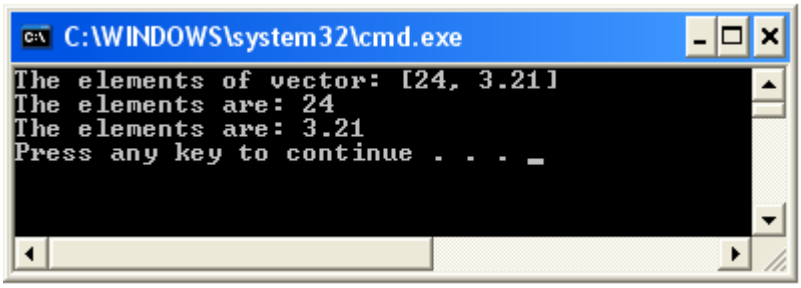
Το αποτέλεσμα:



Στο δεύτερο παράδειγμα θα χρησιμοποιήσουμε το ίδιο Vector (ίδια δεδομένα), όμως η εμφάνιση των στοιχείων θα γίνει με την χρήση της διεπαφής Enumeration.

```
import java.util.*;
class VectorTest3 {
  public static void main(String[] args) {

        Vector v = new Vector(10);
       v.addElement(new Integer(24));
       v.addElement(new Float(3.210));

       Enumeration e=v.elements();
       System.out.println("The elements of vector: " +  v);
       while(e.hasMoreElements()){
           System.out.println("The elements are: " + e.nextElement());
   }
  }
}
```

Το αποτέλεσμα:



Αντιγραφή των Vectors

Η αντιγραφή των Vectors γίνεται με την μέθοδο **clone(),** για την αντιγραφή των στοιχείων από το ένα στο άλλο.

```
Vector a = new Vector();
Vector b = a.clone();
```

Παράδειγμα:

```
import java.util.Vector;

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

    Vector a = new Vector();
    a.addElement(new Integer(100));
    a.addElement(new Integer(200));
    a.addElement(new Integer(300));

    Vector b = (Vector) a.clone();

    System.out.println("O Vector b einai to antigrafo (clone) tou a: ");
    System.out.println(b);

    System.out.println("O Vector c = a: ");
    Vector c = b;
    System.out.println(c);
  }
}
```

Το αποτέλεσμα: