Μηχανικών Πληροφορικής &
Ηλεκτρονικών Συστημάτων

ΔΙΕΘΝΕΣ
ΠΑΝΕΠΙΣΤΗΜΙΟ
ΤΗΣ ΕΛΛΑΔΟΣ

Αντικειμενοστρεφής Προγραμματισμός

Εισαγωγή (2ο μέρος)

Ασδρέ Κατερίνα
asdre@ihu.gr

Από το προηγούμενο μάθημα ξέρουμε ότι....

ΔΙΕΘΝΕΣ ΠΑΝΕΠΙΣΤΗΜΙΟ ΤΗΣ ΕΛΛΑΔΟΣ

Διαδικαστικός VS Αντικειμενοστρεφής

Διαδικασίες και συναρτήσεις που επεξεργάζονται τα δεδομένα είναι τα δομικά στοιχεία των προγραμμάτων	**σκεφτόμαστε με κέντρο το πρόβλημα, χρησιμοποιούμε αντικείμενα για αναπαράσταση οντοτήτων του προβλήματος, έχουμε υψηλότερο επίπεδο *αφαίρεσης***

Διαδικαστικός VS Αντικειμενοστρεφής

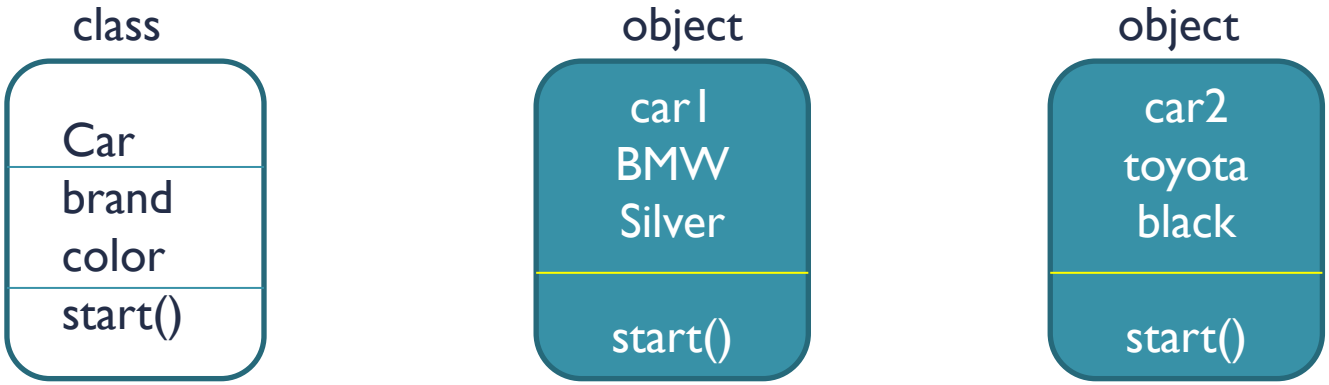
Ο προγραμματισμός στρέφεται προς τις έννοιες/αντικείμενα και όχι πια στις διαδικασίες.
Στα αντικείμενα ανατίθενται χαρακτηριστικά (*ιδιότητες*) που σχετίζονται με αυτά, τα οποία επεξεργάζεται με ειδικές συναρτήσεις (*μεθόδους*).

Δομητής (Constructor) – Κλάσεις Αντικειμένων

- Ειδική μέθοδος δημιουργίας αντικειμένων: **constructor**
- Αρχικοποίηση (μάρκα, χλμ=0, …)
- Αν θέλω να κατασκευάσω και φορτηγό? 'Η ποδήλατο?
- Θα περιγράψω το καθένα ξεχωριστά … έχουν όμως κοινά χαρακτηριστικά! Γι' αυτό δημιουργώ **πρότυπα** που συγκεντρώνουν τα κοινά χαρακτηριστικά και τις κοινές μεθόδους.
- Τα πρότυπα αυτά (σχέδια) ονομάζονται **classes**.

Δομητής (Constructor) – Κλάσεις Αντικειμένων

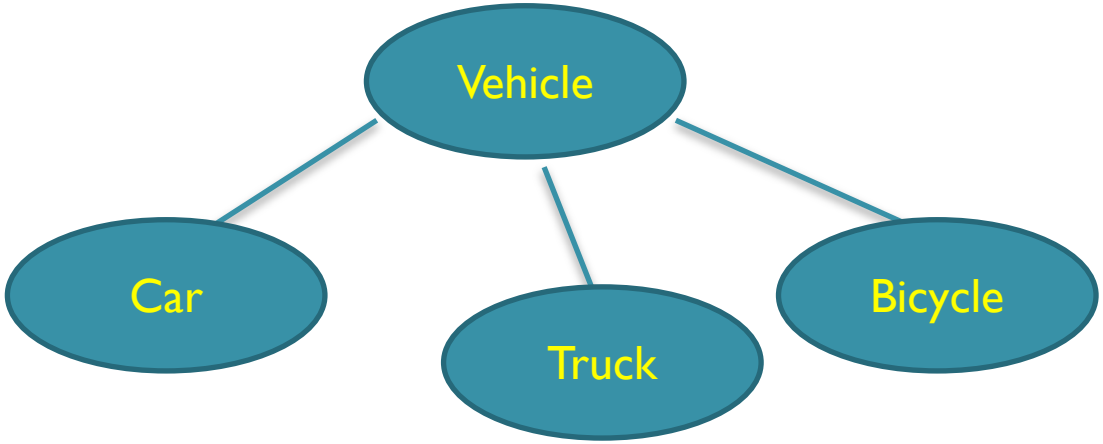
- Από την κλάση γνωρίζω για το αντικείμενο πριν τη δημιουργία του.
- Ουσιαστικά η κλάση ορίζει έναν νέο τύπο δεδομένων.
- Στις κλάσεις εμπεριέχονται και οι δομητές.
- Έχοντας τον τρόπο για να το φτιάξω, δεν σημαίνει ότι το έχω κι όλας!!
- Ένα αντικείμενο είναι ένα στιγμιότυπο της κλάσης.



Κλάσεις Αντικειμένων - Κληρονομικότητα

- Άρα για το αυτοκίνητο, το φορτηγό και το ποδήλατο θα μπορούσα να έχω μια κλάση **vehicle** (κλάση γονέας) η οποία θα περιέχει τα κοινά χαρακτηριστικά και τις μεθόδους των τριών (υποκλάσεις).
- Οι υποκλάσεις κληρονομούν ιδιότητες και μεθόδους.
- Αφού κληρονομούνται δεν τα ξαναορίζω. Μόνο ό,τι δεν υπάρχει στην υπερκλάση.

Κλάσεις Αντικειμένων - Κληρονομικότητα



Java

Μεταγλώττιση και εκτέλεση προγραμμάτων

- **Write Once, Run Anywhere**
- όταν ένα πρόγραμμα σε Java μεταγλωττίζεται, αυτό που παράγεται δεν είναι κώδικας μηχανής για έναν πραγματικό επεξεργαστή, αλλά ένα είδος κώδικα γνωστό ως **java bytecode**, που προορίζεται για μια εικονική μηχανή, **Java Virtual Machine**. Ο κώδικας αυτός εκτελείται από το περιβάλλον εκτέλεσης της Java (Java Runtime Environment, JRE), που είναι διαφορετικό για κάθε λειτουργικό σύστημα και είναι **διερμηνευτής** (interpreter).

Java

Το πρώτο μου πρόγραμμα…περιέχει κλάση…

```
public class HelloWorld {
    public static void main(String[] args){
       System.out.println ("Hello world");
    }
}
```

- Όνομα αρχείου: **HelloWorld.java** αφού μέσα στο πρόγραμμα μας έχουμε μια public κλάση με το όνομα HelloWorld.
- Η κλάση θα πρέπει να περιέχει μια μέθοδο main από την οποία θα ξεκινήσει η εκτέλεση του προγράμματος μας.
- String: κλάση για χειρισμό αλφαριθμητικών.

Java

Το πρώτο μου πρόγραμμα…περιέχει κλάση…

```
public class HelloWorld {
    public static void main(String[] args){
        System.out.println ("Hello world");
    }
}
```

- public, static
- System.out: αντικείμενο
- println(): μέθοδος που τυπώνει το String αντικείμενο που δίνεται ως όρισμα και αλλάζει γραμμή.
- Ομοιότητες με την C?

Java

Το δεύτερό μου πρόγραμμα…

Να γραφτεί πρόγραμμα που αθροίζει δύο ακεραίους

```
public class Sum2Integers
{
  public static void main(String[] args)
  {
    int num1 = 3;  //or int num1=3, num2=10;
    int num2 = 10;
    int result = num1 + num2;
    System.out.println("Result = " + result);
  }
}
```

ΔΙΕΘΝΕΣ ΠΑΝΕΠΙΣΤΗΜΙΟ ΤΗΣ ΕΛΛΑΔΟΣ

Java

Το δεύτερό μου πρόγραμμα…

Να γραφτεί πρόγραμμα που αθροίζει δύο ακεραίους

```
public class Sum2Integers
{
  public static void main(String[] args)
  {
    int num1 = 3;  //or int num1=3, num2=10;
    int num2 = 10;
    int result = num1 + num2;
    System.out.println("Result = " + result);
  }
}
```

- τελεστής «+»: ο βασικός τύπος (int) μετατρέπεται σε String και γίνεται η συνένωση με το String "**Result = **"

ΔΙΕΘΝΕΣ ΠΑΝΕΠΙΣΤΗΜΙΟ ΤΗΣ ΕΛΛΑΔΟΣ

Java

Βασικοί τύποι (πρωτογενείς, δεν είναι αντικείμενα κάποιας κλάσης)

Τύπος	Μέγεθος	Εύρος
`byte`	1 byte	-128 έως 127
`short`	2 bytes	-32768 έως 32767
`int`	4 bytes	-2^{31} έως $2^{31}-1$
`long`	8 bytes	-2^{63} έως $2^{63}-1$
`float`	4 bytes	-3.40292347E+38 έως 3.40292347E+38
`double`	8 bytes	-1.79769313486231570E+308 έως 1.79769313486231570E+308
`boolean`	1 bytes	true / false
`char`	2 bytes	οποιοδήποτε γράμμα ή ψηφίο ή άλλο σύμβολο του κώδικα *unicode*

Java

Τελεστές

Αριθμητικοί Τελεστές	Σχεσιακοί Τελεστές	Λογικοί Τελεστές
+	==	!
–	<	&&
*	>	||
/	<=	^
%	>=	
	!=	

Java

σταθερές

Για να οριστούν οι σταθερές, προηγείται η λέξη **final:**

π.χ. final int PRICE = 12;

Java

Διευρυμένη μετατροπή

- Όταν σε μια αριθμητική έκφραση έχουμε μεταβλητές διαφορετικού τύπου ο τύπος της μεταβλητής στο δεξί μέρος της έκφρασης μετατρέπεται στον τύπο της μεταβλητής στο αριστερό μέρος της έκφρασης. Η μετατροπή γίνεται από τον μικρότερο τύπο στο μεγαλύτερο με την παρακάτω σειρά:

byte -> short -> int -> long -> float -> double

- Για να γίνει διευρυμένη μετατροπή πρέπει ο τύπος αριστερά να είναι μεγαλύτερος από τον τύπο στα δεξιά της έκφρασης.

Java

Casting

- Αν θέλουμε να αποθηκεύσουμε το περιεχόμενο της μεταβλητής κάποιου τύπου σε μεταβλητή διαφορετικού τύπου:

πριν τη μεταβλητή ή την έκφραση βάζουμε μέσα σε παρενθέσεις τον τύπο στον οποίο θέλουμε να μετατραπεί η τιμή της μεταβλητής ή της έκφρασης. Προσοχή: μπορεί να μην χωράει η τιμή μιας μεταβλητής σε κάποια άλλη.

- Π.χ.
```
int a =10;
double d;
d = (double) a / 3;      d = (double) (a /3);
```

Java

Casting

- Αν θέλουμε να αποθηκεύσουμε το περιεχόμενο της μεταβλητής κάποιου τύπου σε μεταβλητή διαφορετικού τύπου:

πριν τη μεταβλητή ή την έκφραση βάζουμε μέσα σε παρενθέσεις τον τύπο στον οποίο θέλουμε να μετατραπεί η τιμή της μεταβλητής ή της έκφρασης. Προσοχή: μπορεί να μην χωράει η τιμή μιας μεταβλητής σε κάποια άλλη.

- Π.χ.
```
int a =10;
double d;
d = (double) a / 3;        d = (double) (a /3);
3.333                      3.0
```

Java

Το τρίτο μου πρόγραμμα…

```
public class Grade{
    static double FGrade(double lab, double theory){
        double result;
        result=0.3 * lab + 0.7 * theory;
        return result;
    }
   public static void main (String argv[]){
       double lab=6.5, theory=5.4;
       System.out.println("Grade = "+ FGrade(lab,theory));
   }
}
```

Java

Το τέταρτο πρόγραμμα...

```
public class Student{
        private int am;
        private String onoma;
        private int apousies;
        private double vathmos;

        public Student() { }
        public Student(int am,String onoma){
                this(am, onoma, 0,0.0);
       }
        public Student(int am,String onoma,int apousies,
                                                double vathmos){
              this.am=am;
              this.onoma=onoma;
              this.apousies=apousies;
              this.vathmos=vathmos;
        }
```

Java

Το τέταρτο πρόγραμμα…

```
public void setVathmos(double vathmos){
      this.vathmos=vathmos;
}

public void setApousies(int apousies){
      this.apousies=apousies;
}


public void setAm(int am){
      this.am=am;
}

public void setOnoma(String onoma){
      this.onoma=onoma;
}
```

Java

Το τέταρτο πρόγραμμα…

```
public int getAM(){
      return am;
}

public String getOnoma(){
      return onoma;
}

public double getVathmos(){
      return vathmos;
}

public int getApousies(){
      return apousies;
}
```

Java

Το τέταρτο πρόγραμμα…

```
    public String toString(){
        return("AM:"+am+"\nOnoma:"+onoma+"\nVathmos:"+
                                vathmos+"\nApousies:"+apousies);
    }

    public static void main(String[] args) {
            Student S1=new Student(181001, "Nikos");
            System.out.println(S1);
            S1.onoma="Katerina";
            System.out.println(S1.toString());
    }
}
```

Java

Το τέταρτο πρόγραμμα…

```
    public String toString(){
        return("AM:"+am+"\nOnoma:"+onoma+"\nVathmos:"+
                                vathmos+"\nApousies:"+apousies);
    }

    public static void main(String[] args) {
            Student S1;
            S1=new Student(181001, "Nikos");
            System.out.println(S1);
            S1.onoma="Katerina";
            System.out.println(S1);
    }
}
```

Java

Το τέταρτο πρόγραμμα…

Δημιουργία Αντικειμένου

- <όνομαΚλάσης> <όνομαΑντικειμένου>;
- <όνομαΑντικειμένου> = new <όνομαΚλάσης> ( ) ;

Π.χ.

```
Student S1;
S1=new Student(181001,"Nikos");
```

Δήλωση Αναφοράς Αντικειμένου

✓ Το αντικείμενο στο οποίο αναφέρεται το S1 περιέχει τα δικά του αντίγραφα των μεταβλητών και μεθόδων (μελών) της κλάσης Student.

Java

Το τέταρτο πρόγραμμα…

Δημιουργία Αντικειμένου

- Όταν χρησιμοποιούμε τα ίδια ονόματα για τις παραμέτρους του δομητή και τα πεδία της κλάσης χρησιμοποιείται το this:

```
public Student(int am,String onoma,int apousies,
                                        double vathmos){
            this.am=am;
            this.onoma=onoma;
            this.apousies=apousies;
            this.vathmos=vathmos;
      }
```

Java

Το τέταρτο πρόγραμμα…

Πρόσβαση σε μέλη του αντικειμένου:

Η πρόσβαση στα μέλη του αντικειμένου γίνεται με τον τελεστή '.'

<όνομαΑντικειμένου>.<μέλος>

π.χ.

```
S1.onoma="Katerina";
```

Java

Πρόσβαση στα δεδομένα του αντικειμένου

- Όταν ένα πεδίο ή μια μέθοδος σε μια κλάση έχει πριν τη δήλωση του τύπου το προσδιοριστικό πρόσβασης **public** υπάρχει ελεύθερη πρόσβαση σε αυτό από οποιαδήποτε άλλη κλάση, χρησιμοποιώντας το όνομα του αντικειμένου και το όνομα του πεδίου ή της μεθόδου.
- Με **private** πριν τη δήλωση του τύπου του πεδίου, ή της μεθόδου, γίνεται ιδιωτικό. Για την πρόσβαση απαιτούνται ειδικές μέθοδοι που πρέπει να είναι μέλη της κλάσης (set / get).

Java

Αν η main ήταν σε άλλη κλάση…

```
public class TestStudent{
	public static void main(String[] args) {
		Student S1=new Student(181001, "Nikos");
		System.out.println(S1);
		//S1.onoma="Katerina";
		S1.setOnoma("Katerina");
		System.out.println(S1);
	}
}
```

Java

Μέθοδος toString()

- Κληρονομείται με τη δημιουργία ενός αντικειμένου.
- Μπορούμε να εμφανίσουμε με τη μορφή που θέλουμε τα δεδομένα ενός αντικειμένου δίνοντας μόνο το όνομα (αναφορά) του.
- Επιστρέφει μια συμβολοσειρά που εμφανίζεται με την εντολή System.out.println().

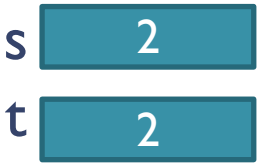
Java

Αναφορές αντικειμένων

- Στους κλασικούς τύπους δεδομένων με το όνομα της μεταβλητής έχουμε πρόσβαση στα περιεχόμενά της.

π.χ. `int s = 2;`

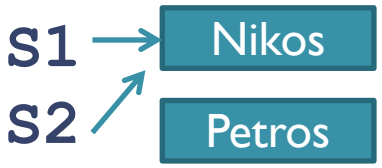
`int t = s;`



- Όταν δημιουργείται αντικείμενο S1 της κλάσης Student το S1 περιέχει τη διεύθυνση –αναφορά στο αντικείμενο.

π.χ.

```
Student S1=new Student(181001, "Nikos");
Student S2=new Student(180000, "Petros");
S2=S1;
```



ΔΙΕΘΝΕΣ ΠΑΝΕΠΙΣΤΗΜΙΟ ΤΗΣ ΕΛΛΑΔΟΣ

Java
Πίνακες

Ένας πίνακας (array) είναι ένα σύνολο μεταβλητών ιδίου τύπου, με ένα όνομα, αποθηκευμένες σε διαδοχικές θέσεις μνήμης. Το μέγεθος μνήμης που δεσμεύεται εξαρτάται από το πλήθος και τον τύπο των στοιχείων του πίνακα.

Δήλωση πίνακα: elementType[ ] arrayRefVariable;

π.χ. int [ ] mymatrix;

Δημιουργία πίνακα: arrayRefVariable = **new** elementType[arraySize];

π.χ. mymatrix = **new** int[10];

ή

int [ ] mymatrix = **new** int[10];

Java
Πίνακες

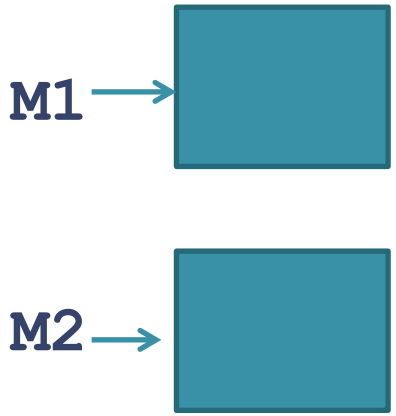
- ✓ Δυναμική αρχικοποίηση: `int mymatrix[ ]= {1,2,3,4,5};`

- ✓ Οι πίνακες διαθέτουν μία μεταβλητή μέλος με όνομα **length** που περιέχει το πλήθος των στοιχείων του πίνακα.

```
System.out.print("Elements= ");
for (i = 0; i < mymatrix.length; i++) {
      System.out.print(pin[i] + " " );
}
System.out.println();
```

Java
Πίνακες

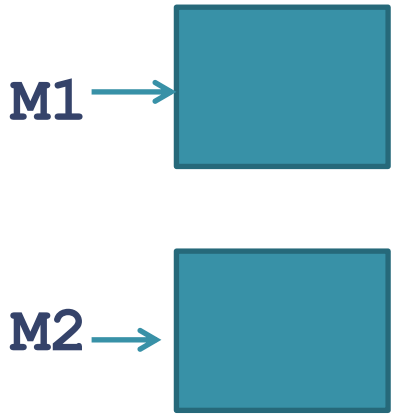
Αντιγραφή πίνακα σε άλλον πίνακα?



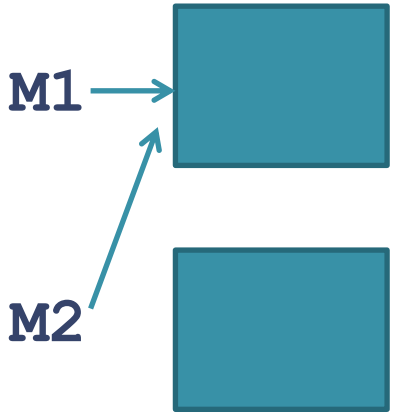
Το αποτέλεσμα της εντολής `M2 = M1;`

Java
Πίνακες

Αντιγραφή πίνακα σε άλλον πίνακα?



Το αποτέλεσμα της εντολής `M2 = M1;`



Java
Πίνακες

Αντιγραφή πίνακα σε άλλον πίνακα:

```
int[] sourceArray = {2, 3, 1, 5, 10};
int[] targetArray = new int[sourceArray.length];
for (int i = 0; i < sourceArray.length; i++) {
    targetArray[i] = sourceArray[i];
}
```

Ή arraycopy(sourceArray, srcPos, targetArray, tarPos, length);

Δηλ.

```
System.arraycopy(sourceArray,0,targetArray,0,
                                    sourceArray.length);
```

Java
Πίνακες αντικειμένων

Πολλές φορές απαιτείται η δημιουργία πίνακα αντικειμένων, όταν τα αντικείμενα που χρειαζόμαστε είναι πολλά:

- Δημιουργούμε το αντικείμενο τύπου πίνακα.
  Δημιουργούμε τα αντικείμενα του πίνακα.

π.χ.

Δημιουργία πίνακα

```
Student Lab[] = new Student [10];
```

Δημιουργία φοιτητή

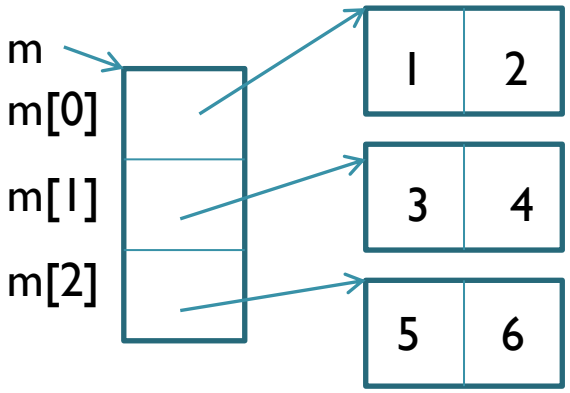
```
Lab[0] = new Student(181001, "Nikos");
```

ΔΙΕΘΝΕΣ ΠΑΝΕΠΙΣΤΗΜΙΟ ΤΗΣ ΕΛΛΑΔΟΣ

Java
Πολυδιάστατοι Πίνακες

Ένας πολυδιάστατος πίνακας είναι ένας μονοδιάστατος πίνακας, τα στοιχεία του οποίου αποθηκεύουν αναφορές σε κάποιον άλλον πίνακα. Τα στοιχεία δηλαδή ενός δισδιάστατου πίνακα αποθηκεύουν αναφορές σε έναν μονοδιάστατο πίνακα, ενός τρισδιάστατου σε έναν δισδιάστατο κ.ο.κ.

- `int m[][]= {{1, 2}, {3, 4}, {5,6}};`



m[0][0]=1 m[0][1]=2

m[1][0]=3 m[1][1]=4

m[2][0]=5 m[2][1]=6

Java
Πολυδιάστατοι Πίνακες

Ένας πολυδιάστατος πίνακας είναι ένας μονοδιάστατος πίνακας, τα στοιχεία του οποίου αποθηκεύουν αναφορές σε κάποιον άλλον πίνακα. Τα στοιχεία δηλαδή ενός δισδιάστατου πίνακα αποθηκεύουν αναφορές σε έναν μονοδιάστατο πίνακα, ενός τρισδιάστατου σε έναν δισδιάστατο κ.ο.κ.

- ```
  int m[][]= {{1, 2}, {3, 4}, {5,6}};
  ```

- ```
  int m1[][] = new int[3][];
  m1[0] = new int[3];
  m1[1] = new int[2];
  m1[2] = new int[1];
  ```

Java
Πολυδιάστατοι Πίνακες

- `int m[][]= {{1, 2}, {3, 4}, {5,6}};`

  m.length=3

- `int m1[][] = new int[3][];`

  `m1[0] = new int[3];` *m1[0].length=3*

  `m1[1] = new int[2];` *m1[1].length=2*

  `m1[2] = new int[1];` *m1[2].length=1*

Java

Μέθοδοι χειρισμού πινάκων της κλάσης Arrays

```
import java.util.Arrays;
```

- **`Arrays.fill(name,val)`**: γεμίζει τις θέσεις του πίνακα name με την τιμή val
- **`Arrays.equals(name1,name2)`**: ελέγχει αν οι πίνακες name1 και name2 έχουν τα ίδια περιεχόμενα
- **`Arrays.sort(name)`**: ταξινομεί τα στοιχεία του πίνακα name