6 ΠΙΝΑΚΕΣ

Πίνακας είναι μια διάταξη στοιχείων που το καθένα τους έχει κάποια συγκεκριμένη **θέση**. Καταλαμβάνει μια περιοχή μνήμης με το λογικό όνομα του πίνακα, ενώ κάθε στοιχείο του ξεχωρίζει με κάποιο **δείκτη**. Στη Java η αρίθμηση των δεικτών ξεκινάει από το 0 μέχρι το μέγεθος του πίνακα−1.

Παράδειγμα ενός πίνακα 5 θέσεων με περιεχόμενα τους αριθμούς 1−5.

1	2	3	4	5
Θέση 0	Θέση 1	Θέση 2	Θέση 3	Θέση 4

Δήλωση Πίνακα

Η Δήλωση ενός Πίνακα στη Java γίνεται με τη δήλωση του **τύπου** των στοιχείων που θα αποθηκεύσει, το **όνομά** του, το **μέγεθός** του και τον τελεστή **new**, επειδή οι πίνακες στη Java είναι αντικείμενα.

Παράδειγμα

```
int pin1[] = new int[5];
```

όπου δηλώνουμε τον πίνακα pin1, ο οποίος θα αποθηκεύσει 5 ακέραιους αριθμούς.

Το ίδιο αποτέλεσμα έχουμε με την εντολή

```
int[] pin1 = new int[5];
```

αρκεί να υπάρχουν οι αγκύλες στον τύπο ή στο όνομα του πίνακα.

Το μέγεθος του πίνακα μπορεί να είναι η τιμή μιας μεταβλητής. Οι επόμενες εντολές έχουν το ίδιο αποτέλεσμα :

```
int n = 5;
int pin1[] = new int[n];
```

Ένας πίνακας μπορεί να δημιουργηθεί με **δυναμική αρχικοποίηση**, όπου οι τιμές του περικλείονται σε άγκιστρα και χωρίζονται με κόμμα. Σ' αυτή την περίπτωση **δε** χρειάζεται ο τελεστής **new**.

Παράδειγμα

```
int pin2[] = {1, 2, 3, 4, 5};
```

όπου δηλώνουμε τον πίνακα pin2, ο οποίος θα αποθηκεύσει τους ακέραιους αριθμούς από το 1 μέχρι το 5. **Σαν στοιχεία μέσα στις αγκύλες, εκτός από σταθερούς αριθμούς, μπορεί να είναι ονόματα μεταβλητών, εκφράσεις ή κλήσεις μεθόδων.**

Η πρόσβαση σε κάποιο στοιχείο του πίνακα γίνεται με το όνομά του και τη θέση του (δείκτης) μέσα σε αγκύλες. Π.χ. το pin1[3] αναφέρεται στο 4^{ο} στοιχείο του pin1.

6.1 Παράδειγμα Δημιουργίας – Γεμίσματος 2 πινάκων με random τιμές και Δυναμική Αρχικοποίηση

- Να γραφεί Αλγόριθμος/Πρόγραμμα το οποίο γεμίζει έναν πίνακα με **δυναμική αρχικοποίηση** και δίνει τις τιμές από το 1 μέχρι το 5, δηλώνει έναν άλλον πίνακα 5 θέσεων, τον οποίο γεμίζει με τυχαίες τιμές και εμφανίζει τα περιεχόμενα των 2 πινάκων.

Αλγόριθμος

1. Δήλωση - Αρχικοποίηση πίνακα pin1
2. Δήλωση - Γέμισμα πίνακα pin2 με τυχαίες τιμές
   Για κάθε θέση i = 0 μέχρι (n – 1)
   Εκχωρούμε μια τυχαία τιμή μεταξύ 0 και 10 στο στοιχείο pin2[i]
3. Εμφάνιση στοιχείων πίνακα pin1
   Για κάθε θέση i = 0 μέχρι 4
   Εμφανίζουμε το στοιχείο pin1[i]
4. Εμφάνιση στοιχείων πίνακα pin2
   Για κάθε θέση i = 0 μέχρι (n – 1)
   Εμφανίζουμε το στοιχείο pin2[i]

Πρόγραμμα

```
public class Pin1 {
/*
Το πρόγραμμα δημιουργεί 2 πίνακες, τον πρώτο με δυναμική αρχικοποίηση
και τον γεμίζει με τις ακέραιες τιμές από 1 μέχρι 5, ενώ το δεύτερο τον
γεμίζει με τυχαίες τιμές από 0 μέχρι 10 και εμφανίζει τα στοιχεία των 2
πινάκων
*/
    public static void main(String[] args) {
        int i;

        // Εκχώρηση ακέραιας τιμής στο μέγεθος του πίνακα 2
        int n = 5;

        // Δήλωση - Αρχικοποίηση πίνακα 1
        int pin1[] = {1,2,3,4,5};

       // Δήλωση πίνακα 2
        double pin2[] = new double[n];
```

```
        // Γέμισμα πίνακα 2 με τυχαίες ακέραιες τιμές
        for (i = 0;i <= n-1;i++)
        {
         pin2[i] = Math.random()*10;
        }

        // Εμφάνιση στοιχείων πίνακα 1
        System.out.print("Πίνακας 1 = ");
        for (i = 0;i <= 4;i++)
        {
         System.out.print(pin1[i] + " " );
        }
        System.out.println();

        // Εμφάνιση στοιχείων πίνακα 2
        System.out.println("\nΠίνακας 2\n");
        for (i = 0;i <= n-1;i++)
        {
         System.out.println(pin2[i] + " " );
        }
        System.out.println();
    }

}
```

Έξοδος Προγράμματος :

```
run:

Πίνακας 1 = 1 2 3 4 5

Πίνακας 2

4.577221658514528
1.2033315418144874
6.063131490340102
9.267092004007694
1.1890233835178776
BUILD SUCCESSFUL (total time: 0 seconds)
```

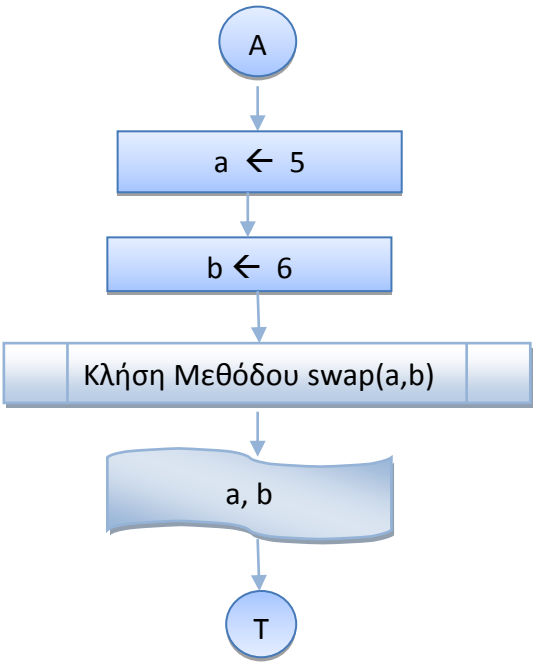
6.2 Πέρασμα Παραμέτρων σε Μεθόδους

Οι παράμετροι των βασικών τύπων περνάνε στην μέθοδο με **τιμή**, δηλαδή στη μέθοδο είναι διαθέσιμο ένα **αντίγραφο** της τιμής της μεταβλητής και **ΟΧΙ** η **διεύθυνσή** της, οπότε **δεν αλλάζει το περιεχόμενο** που είχαν πριν, ακόμα και αν αλλάζει η τιμή τους μέσα στη μέθοδο. Αυτό μπορεί να γίνει κατανοητό με το επόμενο παράδειγμα :

6.2.1 Παράμετροι με Τιμή – Ανταλλαγή των Τιμών Δυο Μεταβλητών

Να γραφεί πρόγραμμα που να καλεί μια μέθοδο η οποία **ανταλλάσει** τις τιμές 2 ακέραιων αριθμών.

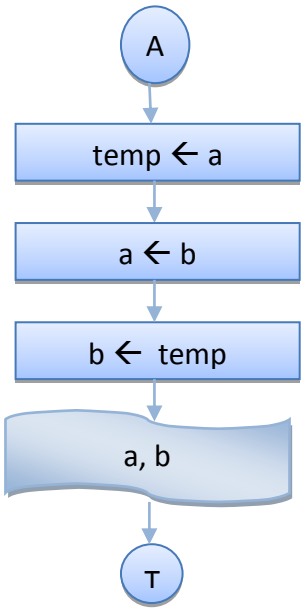
ΛΟΓΙΚΟ ΔΙΑΓΡΑΜΜΑ main()



ΑΛΓΟΡΙΘΜΟΣ main()

1. Δίνω την τιμή `5` στο `a` (**a ← 5**)
2. Δίνω την τιμή `6` στο `b` (**b ← 6**)
3. Κλήση Μεθόδου `swap(a, b)`
4. Εμφάνιση `a, b`

ΛΟΓΙΚΟ ΔΙΑΓΡΑΜΜΑ ΜΕΘΟΔΟΥ `swap(a, b)`



ΑΛΓΟΡΙΘΜΟΣ ΜΕΘΟΔΟΥ `swap(a,b)`

1. Αποθηκεύω το **a** στο **temp** (**temp←a**)
2. Αποθηκεύω το **b** στο **a** (**a ← b**)
3. Αποθηκεύω το **temp** στο **b** (**temp ← b**)
4. Εμφάνιση `a, b`

ΠΡΟΓΡΑΜΜΑ

```
public class SwapAB {
/* Πρόγραμμα το οποίο δίνει τις τιμές 5 και 6 σε δύο μεταβλητές a και b και
καλεί τη μέθοδο swap(), η οποία ανταλάσσει τις τιμές των a και b, τις οποίες
και εμφανίζει.
*/
static void swap(int a, int b){
    // Μέθοδος ανταλλαγής των τιμών 2 ακέραιων μεταβλητών
    int temp = a;
    a = b;
    b = temp;
    System.out.print("Τιμές a, b μέσα στη Μέθοδο : ");
    System.out.println("a = " + a + ", b = " + b);
    }

   public static void main(String[] args) {
    int a = 5, b = 6;
    swap( a,  b);
    System.out.print("Τιμές a, b μετά την κλήση της μεθόδου : ");
    System.out.println("a = " + a + ", b = " + b);
    }
}
```

Έξοδος Προγράμματος

```
run:
Τιμές a, b μέσα στη Μέθοδο : a = 6, b = 5
Τιμές a, b μετά την κλήση της μεθόδου : a = 5, b = 6
BUILD SUCCESSFUL (total time: 0 seconds)
```

Παρατηρήσεις

- Τα ορίσματα (οι μεταβλητές `a, b` της `main()`) περνάνε στην μέθοδο **με τιμή**, δηλαδή ένα αντίγραφο του περιεχομένου των μεταβλητών `a, b` **και <u>όχι</u> οι διευθύνσεις** των μεταβλητών.
- Με τις αλλαγές στην μέθοδο το περιεχόμενο των μεταβλητών `a, b` της `main()` **δεν επηρεάζεται.**
- Μετά την κλήση της μεθόδου οι μεταβλητές `a, b` περιέχουν **τις ίδιες τιμές** που είχαν και **πριν** την κλήση της μεθόδου.

6.3 Οι Πίνακες σαν Παράμετροι σε Μεθόδους

Σε αντίθεση με της μεταβλητές απλού τύπου, οι πίνακες περνάνε **με αναφορά** σαν παράμετροι στις μεθόδους, οπότε μπορεί να αλλάξει το περιεχόμενό τους.

Η μέθοδος μπορεί επίσης να **επιστρέφει πίνακα**, πράγμα που πρέπει να δηλωθεί στην υπογραφή της μεθόδου.

Παράδειγμα

Η παρακάτω μέθοδος

```
static double[] fillPin2a(int n ){
// Μέθοδος δημιουργίας στοιχείων πίνακα 2, επιστρεφόμενος τύπος = Πίνακας
        double pin2[] = new double[n];
        for (int i = 0;i <= n-1;i++)
        {
         pin2[i] = Math.random()*10;
        }
        return pin2;
    }
```

επιστρέφει πίνακα με την εντολή **`return pin2 - double pin2[] = new double[n]`**, ο οποίος περιέχεται στη δήλωσή της μεθόδου (**`double[]`** `fillPin2a(int n ){`).

6.3.1 Παράδειγμα Δημιουργίας – Γεμίσματος 2 πινάκων με random τιμές και Δυναμική Αρχικοποίηση – Χρήση Μεθόδων για Γέμισμα - Εμφάνιση

- Να γραφεί Αλγόριθμος/Πρόγραμμμα το οποίο γεμίζει έναν πίνακα με **δυναμική αρχικοποίηση** και δίνει τις τιμές από το 1 μέχρι το 5, δηλώνει έναν άλλον πίνακα 5 θέσεων, τον οποίο γεμίζει με **τυχαίες τιμές** καλώντας τη μέθοδο **`fillPin2()`** και τη μέθοδο **`fillPin2a()`**, η οποία επιστρέφει πίνακα και εμφανίζει τα περιεχόμενα των 2 πινάκων καλώντας τις μεθόδους **`showPin1()`** και **`showPin2()`**. Η μέθοδος `showPin2()` καλείται επίσης με παράμετρο την κλήση της μεθόδου `fillPin2a()` για να εμφανίσει τα στοιχεία του πίνακα που επιστρέφει η μέθοδος `fillPin2a()`.

Αλγόριθμος

1. Δήλωση - Αρχικοποίηση πίνακα 1
2. Δήλωση – Κλήση μεθόδου **`fillPin2()`** για το γέμισμα του πίνακα 2 με τυχαίες τιμές
3. Εμφάνιση στοιχείων πίνακα 1 - Κλήση μεθόδου **`showPin1()`**
4. Εμφάνιση στοιχείων πίνακα 2 - Κλήση μεθόδου **`showPin2()`**
5. Γέμισμα πίνακα 2 με τυχαίες τιμές - Επιστροφή πίνακα - Κλήση μεθόδου `fillPin2a()`
6. Εμφάνιση στοιχείων πίνακα 2 - Κλήση μεθόδου **`showPin2()`**
7. Γέμισμα πίνακα 2 με τυχαίες τιμές - Επιστροφή πίνακα - Κλήση μεθόδου `fillPin2a()` – παράμετρος στην εντολή `System.out.println`
8. Εμφάνιση στοιχείων πίνακα 2 - Κλήση μεθόδου **`showPin2()`**

Πρόγραμμα

```
public class PinMethods {
/*
    Το πρόγραμμα δημιουργεί 2 πίνακες, τον πρώτο με δυναμική αρχικοποίηση,
    και τον γεμίζει με τις ακέραιες τιμές από 1 μέχρι 5, ενώ το δεύτερο τον
    γεμίζει με τυχαίες τιμές από 0 μέχρι 10 και εμφανίζει τα στοιχεία των 2
    πινάκων. Για το γέμισμα του πίνακα 2 και την εμφάνιση των στοιχείων των
    2 πινάκων χρησιμοποιεί τις static μεθόδους showPin1(), showPin2(),
    fillPin2() και fillPin2a() η οποία επιστρέφει πίνακα.
    */
    static void showPin1(int pin1[] ){
    // Μέθοδος Εμφάνισης στοιχείων πίνακα 1
        System.out.print("Πίνακας 1 = ");
        for (int i = 0;i <= 4;i++)
        {
         System.out.print(pin1[i] + " " );
        }
        System.out.println();
     }

    static void fillPin2(int n, double pin2[] ){
    // Μέθοδος δημιουργίας στοιχείων πίνακα1, πέρασμα μεταβλητής pin2 με
αναφορά
        for (int i = 0;i <= n-1;i++)
        {
         pin2[i] = Math.random()*10;
        }
    }

    static double[] fillPin2a(int n ){
    // Μέθοδος δημιουργίας στοιχείων πίνακα 2, επιστρεφόμενος τύπος = Πίνακας
        double pin2[] = new double[n];
        for (int i = 0;i <= n-1;i++)
        {
         pin2[i] = Math.random()*10;
        }
        return pin2;
     }

    static void showPin2(int n, double pin2[] ){
    // Μέθοδος Εμφάνισης στοιχείων πίνακα 2
       for (int i = 0;i <= n-1;i++)
        {
         System.out.println(pin2[i] + " " );
        }
        System.out.println();
    }

    public static void main(String[] args) {
        int i;

        // Εκχώρηση ακέραιας τιμής στο μέγεθος του πίνακα 2
        int n = 5;

        // Δήλωση - Αρχικοποίηση πίνακα 1
        int pin1[] = {1,2,3,4,5};

        // Δήλωση πίνακα 2
        double pin2[] = new double[n];

        // Γέμισμα πίνακα 2 με τυχαίες  τιμές
        fillPin2( n,  pin2);

        // Εμφάνιση στοιχείων πίνακα 1
```

```
        showPin1(pin1);
        // Εμφάνιση στοιχείων πίνακα 2
        System.out.println("\nΠίνακας 2\n ");
        showPin2(n, pin2);

        // Γέμισμα πίνακα 2 με τυχαίες τιμές - Επιστροφή πίνακα
        pin2 = fillPin2a( n );

        // Εμφάνιση στοιχείων πίνακα 2 - Επιστροφή πίνακα
        System.out.println("Πίνακας 2 - Επιστροφή πίνακα\n ");
        showPin2(n, pin2);

        // Εμφάνιση στοιχείων πίνακα 2 - Γέμισμα πίνακα 2 με τυχαίες τιμές -
        // Επιστροφή πίνακα
       System.out.println("Πίνακας 2 - Επιστροφή πίνακα - κλήση μεθόδου στην"
                          + " εντολή System.out.println\n ");
        showPin2(n, fillPin2a( n ));
    }
}
```

Έξοδος Προγράμματος :

```
run:

Πίνακας 1 = 1 2 3 4 5

Πίνακας 2

3.6518718113025295
9.028672546645574
9.803978963288898
5.312458213990624
7.522460445626495

Πίνακας 2 - Επιστροφή πίνακα

5.432465106350786
1.4239093219143784
4.4804895751466685
5.617742872661365
2.666166716016672

Πίνακας 2 - Επιστροφή πίνακα - κλήση μεθόδου στην εντολή System.out.println

0.5326041961274375
9.610193943893067
7.385668259217066
2.919175861736183
6.606580446346813

BUILD SUCCESSFUL (total time: 0 seconds)
```

6.4 Το μέλος length

Το `length` είναι **μέλος** του αντικειμένου τύπου πίνακας και παίρνει την τιμή του μεγέθους του πίνακα που δηλώνουμε με `new` ή με δυναμική αρχικοποίηση.

Παράδειγμα 1

```
int pin1[] = {1, 2, 3, 4,5};          // pin1.length = 5
int pin2[] = new int[5];              // pin2.length = 5
```

Παράδειγμα 2

Ο αριθμός 4 (η θέση του τελευταίου στοιχείου του πίνακα `pin1`) στη μέθοδο `showPin1(int pin1[])` θα μπορούσε να αντικατασταθεί από το `pin1.length-1`, οπότε η μέθοδος `showPin1` θα μπορούσε να γραφεί :

```
static void showPin1(int pin1[] ){
// Μέθοδος Εμφάνισης στοιχείων πίνακα 1
    System.out.print("Πίνακας 1 = ");
    for (int i = 0;i <= pin1.length-1;i++)
    {
     System.out.print(pin1[i] + " " );
    }
    System.out.println();

}
```

Παράδειγμα 3

Το `n-1` (η θέση του τελευταίου στοιχείου του πίνακα `pin2`) στη μέθοδο `showPin2(int n, double pin2[])` θα μπορούσε να αντικατασταθεί από το `pin2.length-1`, οπότε η μέθοδος `showPin2` θα μπορούσε να γραφεί :

```
static void showPin2( double pin2[] ){
// Μέθοδος Εμφάνισης στοιχείων πίνακα 2
   for (int i = 0;i <= pin2.length -1;i++)
    {
     System.out.println(pin2[i] + " " );
    }
    System.out.println();
}
```

Παράδειγμα 4

Το ίδιο μπορεί να γίνει και για τη μέθόδο `fillPin2(int n, double pin2[])`

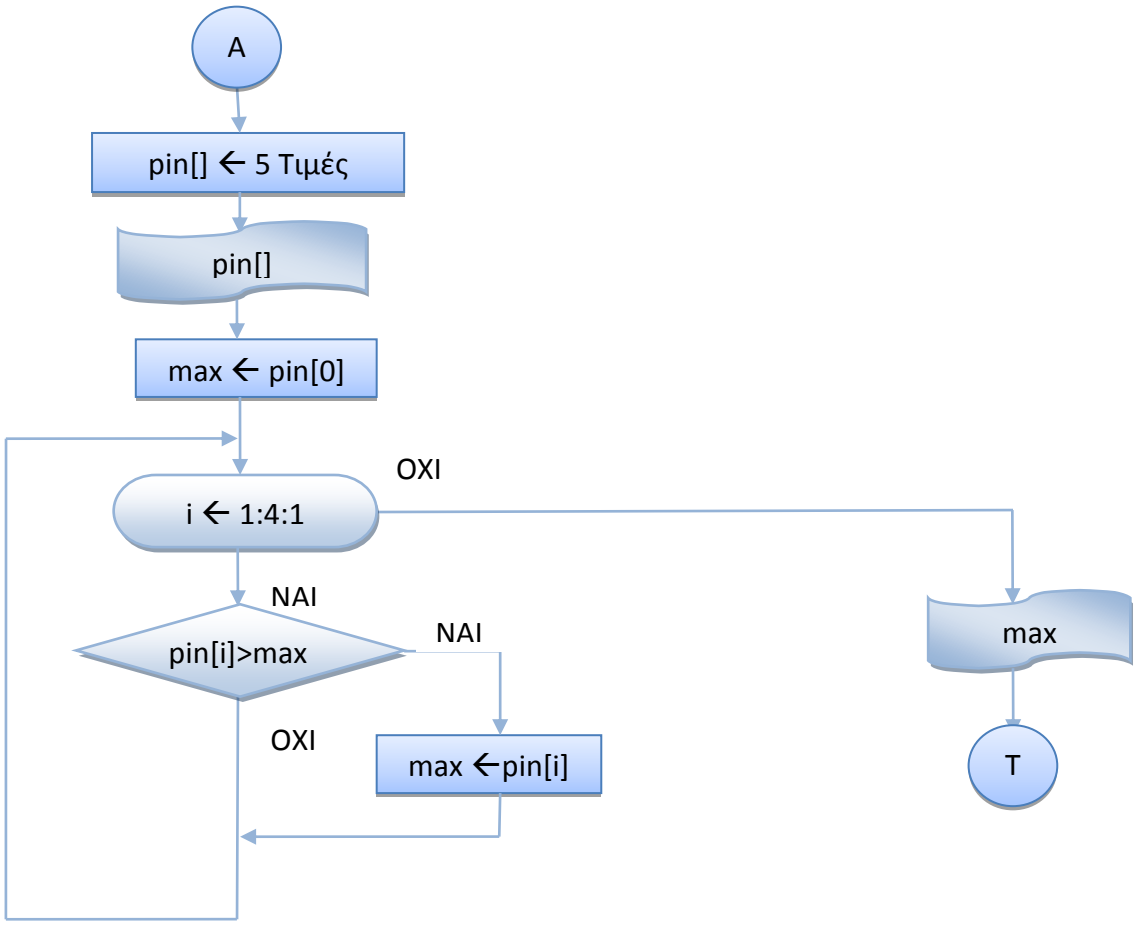
6.5 Εύρεση στοιχείου με τη Μέγιστη ή Ελάχιστη Τιμή σε έναν Πίνακα

Σε πολλά προβλήματα, χρειάζεται να βρούμε το στοιχείο με τη μεγαλύτερη ή μικρότερη τιμή σε έναν πίνακα. Ο πιο συνηθισμένος τρόπος είναι να ελέγξουμε **όλα** τα στοιχεία του πίνακα και να κρατάμε κάθε φορά το μεγαλύτερο, όπως φαίνεται στο επόμενο παράδειγμα :

6.5.1 Εύρεση στοιχείου με τη Μέγιστη Τιμή σε έναν Πίνακα

- Να γραφεί Πρόγραμμα που να δημιουργεί και να γεμίζει έναν πίνακα ακεραίων με δυναμική αρχικοποίηση, να εμφανίζει τα στοιχεία του, να βρίσκει το στοιχείο με τη μεγαλύτερη τιμή και να το εμφανίζει.

ΛΟΓΙΚΟ ΔΙΑΓΡΑΜΜΑ



Αλγόριθμος

1. Δήλωση - Αρχικοποίηση πίνακα `pin[]` με `5` θέσεις
2. Εμφάνιση στοιχείων πίνακα `pin`
3. Θέτουμε σαν μέγιστο στοιχείο `max` το πρώτο στοιχείο `pin[0]` (αφού δεν υπάρχει άλλο μέχρι στιγμής να συγκριθεί)
4. **Για** τα υπόλοιπα στοιχεία `pin[i]`, για `i = 1` μέχρι `4` :
   Αν το στοιχείο `pin[i]` είναι μεγαλύτερο από το `max`
   Θέτουμε σαν μέγιστο στοιχείο `max` το `pin[i]`
5. Εμφάνιση της τιμής του μεγίστου στοιχείου `max`

Πρόγραμμα

```
public class PinMax {
/* Πρόγραμμα που δημιουργεί και γεμίζει έναν πίνακα ακεραίων με δυναμική
αρχικοποίηση, εμφανίζει τα στοιχεία του, Βρίσκει το στοιχείο με τη
μεγαλύτερη τιμή και το Εμφανίζει
*/
    public static void main(String[] args) {
        int i, max;

        // Δήλωση - Αρχικοποίηση πίνακα
        int pin[] = {10,2,13,14,5};

        // Εμφάνιση στοιχείων πίνακα
        System.out.print("Πίνακας = ");
        for (i = 0;i <= pin.length-1;i++)
        {
         System.out.print(pin[i] + " " );
        }
        System.out.println();

        // Εύρεση μεγίστου στοιχείου του πίνακα
        max = pin[0];
        for (i = 1;i <= pin.length-1;i++)
        {
         if (pin[i] > max ) {
             max = pin[i];
         }
        }

        // Εμφάνιση μεγίστου στοιχείου του πίνακα
        System.out.println("Μέγιστο στοιχείο = " + max);
       }
}
```

Έξοδος Προγράμματος :

```
run:
Πίνακας = 10 2 13 14 5
Μέγιστο στοιχείο = 14
BUILD SUCCESSFUL (total time: 0 seconds)
```

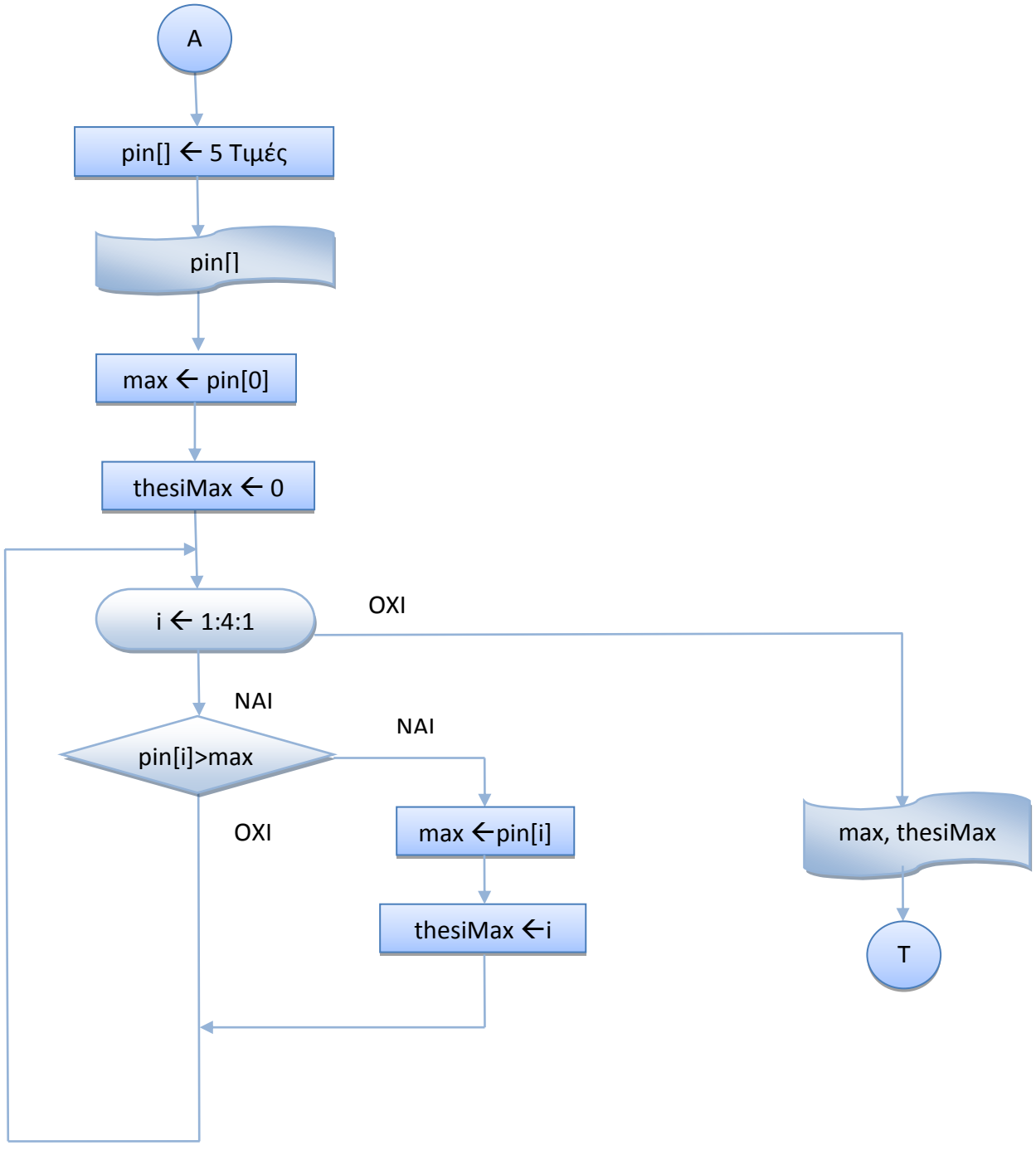
Παρατήρηση :

Πολλές φορές μας ενδιαφέρει να βρούμε όχι μόνο το στοιχείο με τη μεγαλύτερη τιμή, αλλά και τη **θέση** αυτού του στοιχείου στον πίνακα. Έχοντας τη θέση, μπορούμε να βρούμε και το αντίστοιχο στοιχείο. Αυτό φαίνεται στο επόμενο παράδειγμα :

6.5.2 Εύρεση στοιχείου με τη Μέγιστη Τιμή σε έναν Πίνακα και της θέσης του στον πίνακα

- Να γραφεί Πρόγραμμα που να δημιουργεί και να γεμίζει έναν πίνακα ακεραίων με δυναμική αρχικοποίηση, να εμφανίζει τα στοιχεία του, να βρίσκει το στοιχείο με τη μεγαλύτερη τιμή και τη θέση του στον πίνακα και να εμφανίζει το στοιχείο με τη μεγαλύτερη τιμή και **τη θέση του** στον πίνακα

ΛΟΓΙΚΟ ΔΙΑΓΡΑΜΜΑ



Αλγόριθμος

1. Δήλωση - Αρχικοποίηση πίνακα `pin[]` με 5 θέσεις
2. Εμφάνιση στοιχείων πίνακα `pin`
3. Θέτουμε σαν μέγιστο στοιχείο `max` το πρώτο στοιχείο `pin[0]` (αφού δεν υπάρχει άλλο μέχρι στιγμής να συγκριθεί)
4. **Θέτουμε σαν θέση μεγίστου στοιχείου `thesiMax` το 0**
5. **Για** τα υπόλοιπα στοιχεία `pin[i]`, για `i = 1` μέχρι 4 :
   Αν το στοιχείο `pin[i]` είναι μεγαλύτερο από το `max`
   Θέτουμε σαν μέγιστο στοιχείο `max` την τιμή του `pin[i]`
   Θέτουμε την τιμή του `i` στο `thesiMax`
6. Εμφάνιση της τιμής του μεγίστου στοιχείου `max` και **της θέσης του `thesiMax`**

Πρόγραμμα

```
public class PinMaxThesi {
/* Πρόγραμμα που δημιουργεί και γεμίζει έναν πίνακα ακεραίων με δυναμική
αρχικοποίηση, εμφανίζει τα στοιχεία του, Βρίσκει το στοιχείο με τη μεγαλύτερη
τιμή και τη θέση του στον πίνακα και Εμφανίζει το στοιχείο μεγαλύτερη τιμή και
τη θέση του στον πίνακα
*/
    public static void main(String[] args) {
        int i, max;

        // Δήλωση - Αρχικοποίηση πίνακα
        int pin[] = {10,2,13,14,5};

        // Εμφάνιση στοιχείων πίνακα
        System.out.print("Πίνακας = ");
        for (i = 0;i <= pin.length-1;i++)
        {
         System.out.print(pin[i] + " " );
        }
        System.out.println();

        // Εύρεση μεγίστου στοιχείου του πίνακα και της θέσης του στον πίνακα
        max = pin[0];
        int thesiMax = 0;
        for (i = 1;i <= pin.length-1;i++)
        {
         if (pin[i] > max ) {
             max = pin[i];
             thesiMax = i;
        }
        }
        // Εμφάνιση μεγίστου στοιχείου και της θέσης του στον πίνακα
        System.out.println("Μέγιστο στοιχείο = " + max + " στη θέση " +
thesiMax );
    }
}
```

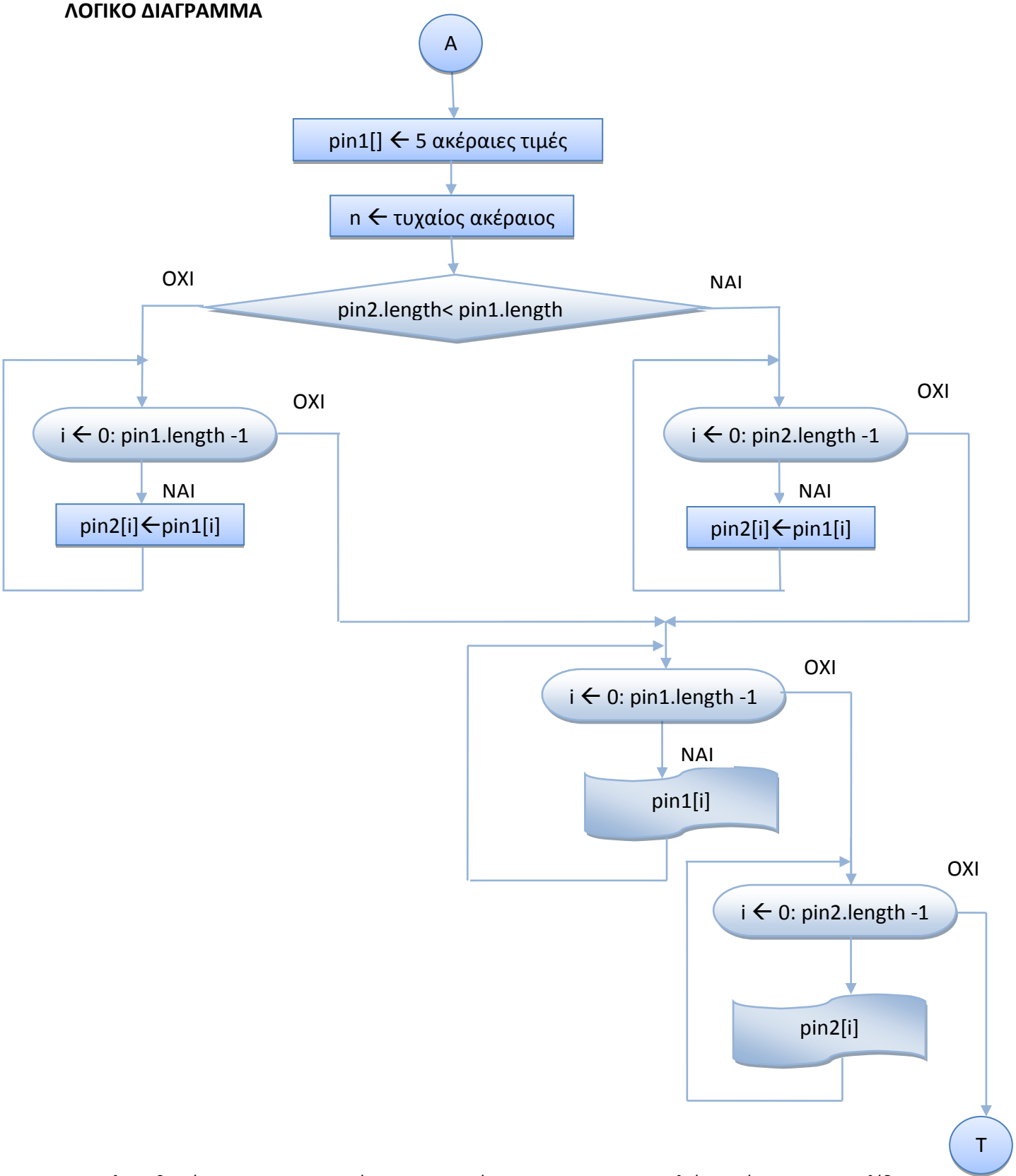
Έξοδος Προγράμματος :

```
run:
Πίνακας = 10 2 13 14 5
Μέγιστο στοιχείο = 14 στη θέση 3
BUILD SUCCESSFUL (total time: 0 seconds)
```

6.5.3 Αντιγραφή Στοιχείων ενός Πίνακα σε κάποιον άλλο Πίνακα

- Να γραφεί Πρόγραμμα που να δημιουργεί `2` πίνακες, τον πρώτο με δυναμική αρχικοποίηση και τον γεμίζει με τις ακέραιες τιμές από `1` μέχρι `5`, ενώ για το δεύτερο δημιουργεί με τη μέθοδο `Math.random()` μια τυχαία ακέραια τιμή για το μέγεθος του πίνακα `2` μεταξύ του `1` και `10` και αντιγράφει ΟΣΑ στοιχεία του πίνακα `1` χωράνε στον πίνακα `2` και εμφανίζει τα στοιχεία των `2` πινάκων.

ΛΟΓΙΚΟ ΔΙΑΓΡΑΜΜΑ



Αλγόριθμος

1. Δήλωση - Αρχικοποίηση πίνακα `pin1[]` με 5 θέσεις τους αριθμούς `1` μέχρι `5`
2. Εκχώρηση τυχαίας ακέραιας τιμής `n` στο μέγεθος του πίνακα `2`
3. Δήλωση πίνακα `pin2[]` με `n` θέσεις
4. Αντιγραφή στον πίνακα `2` των στοιχείων του πίνακα `1`
   <u>Αν</u> το μέγεθος του `pin2` < μέγεθος του `pin1` (`pin2.length < pin1.length`)
   Αντιγραφή **όσων** στοιχείων του πίνακα `pin1` χωράνε στον πίνακα `pin2`
   <u>Διαφορετικά</u>
   Αντιγραφή **ΟΛΩΝ** των στοιχείων του πίνακα `pin1` στον πίνακα `pin2`
5. Εμφάνιση των στοιχείων του πίνακα `pin1`
6. Εμφάνιση των στοιχείων του πίνακα `pin2`

Πρόγραμμα

```
public class PinCopy {
/* Το πρόγραμμα δημιουργεί 2 πίνακες, τον πρώτο με δυναμική αρχικοποίηση
και τον γεμίζει με τις ακέραιες τιμές από 1 μέχρι 5, ενώ για το δεύτερο
δημιουργεί με τη μέθοδο Math.random() μια τυχαία ακέραια τιμή για το
μέγεθος του πίνακα 2 μεταξύ του 1 και 10 και αντιγράφει ΟΣΑ στοιχεία του
πίνακα 1 χωράνε στον πίνακα 2 και εμφανίζει τα στοιχεία των 2 πινάκων.
*/
    public static void main(String[] args) {
        int i, j;

        // Δήλωση - Αρχικοποίηση πίνακα 1
        int pin1[]= {1,2,3,4,5};

         // Εκχώρηση τυχαίας ακέραιας τιμής στο μέγεθος του πίνακα 2
        int n = (int) ( 1 + Math.random()*10);
        System.out.println("n = " + n);

        // Δήλωση πίνακα 2
        int pin2[]= new int[n];

        // Αντιγραφή στον πίνακα 2 των στοιχείων του πίνακα 1
        if ( pin2.length < pin1.length)
         for (i = 0;i < pin2.length;i++)
          pin2[i] = pin1[i];
        else
         for (i = 0;i < pin1.length;i++)
          pin2[i] = pin1[i];

        // Εμφάνιση στοιχείων πίνακα 1
        System.out.print("Πίνακας 1 = ");
        for (i = 0;i < pin1.length;i++)
          System.out.print(pin1[i] + " " );
        System.out.println();

        // Εμφάνιση στοιχείων πίνακα 2
        System.out.print("Πίνακας 2 = ");
        for (i = 0;i < pin2.length;i++)
          System.out.print(pin2[i] + " " );
        System.out.println();
        }
}
```

Έξοδος Προγράμματος

```
run:
n = 3
Πίνακας 1 = 1 2 3 4 5
Πίνακας 2 = 1 2 3

n = 5
Πίνακας 1 = 1 2 3 4 5
Πίνακας 2 = 1 2 3 4 5

n = 8
Πίνακας 1 = 1 2 3 4 5
Πίνακας 2 = 1 2 3 4 5 0 0 0
BUILD SUCCESSFUL (total time: 0 seconds)
```

- Όπως φαίνεται στην έξοδο του προγράμματος, για **`n = 5`** και οι 2 πίνακες έχουν τα **ΙΔΙΑ** περιεχόμενα. Είναι όμως **ΔΙΑΦΟΡΕΤΙΚΟΙ** πίνακες. Η αλλαγή στα στοιχεία ενός πίνακα **ΔΕΝ** επηρεάζει τον άλλο.

6.5.4 Αντιγραφή Στοιχείων ενός Πίνακα σε κάποιον άλλο Πίνακα με κλήση μεθόδων

- Να γραφεί Πρόγραμμα που να δημιουργεί 2 πίνακες, τον πρώτο με δυναμική αρχικοποίηση και τον γεμίζει με τις ακέραιες τιμές από 1 μέχρι 5, ενώ για το δεύτερο δημιουργεί με τη μέθοδο `Math.random()` μια τυχαία ακέραια τιμή για το μέγεθος του πίνακα 2 μεταξύ του 1 και 10 και αντιγράφει ΟΣΑ στοιχεία του πίνακα 1 χωράνε στον πίνακα 2 και εμφανίζει τα στοιχεία των 2 πινάκων.

Αλγόριθμος

1. Δήλωση - Αρχικοποίηση πίνακα `pin1[]` με 5 θέσεις τους αριθμούς 1 μέχρι 5
2. Εκχώρηση τυχαίας ακέραιας τιμής `n` στο μέγεθος του πίνακα 2
3. Δήλωση πίνακα `pin2[]` με `n` θέσεις
4. Αντιγραφή στον πίνακα 2 των στοιχείων του πίνακα 1 καλώντας τη μέθοδο `copyPin1toPin2()`
5. Εμφάνιση των στοιχείων του πίνακα `pin1` καλώντας τη μέθοδο `showPin()`
6. Εμφάνιση των στοιχείων του πίνακα `pin2` καλώντας τη μέθοδο `showPin()`

Πρόγραμμα

```
public class PinCopyMethods {
/* Το πρόγραμμα δημιουργεί 2 πίνακες, τον πρώτο με δυναμική αρχικοποίηση
και τον γεμίζει με τις ακέραιες τιμές από 1 μέχρι 5, ενώ για το δεύτερο
δημιουργεί με τη μέθοδο Math.random() μια τυχαία ακέραια τιμή μεταξύ του
1 και 10 για το μέγεθος του πίνακα 2 και αντιγράφει καλώντας τη μέθοδο
copyPin1toPin2() ΟΣΑ στοιχεία του πίνακα 1 χωράνε στον πίνακα 2 και
εμφανίζει τα στοιχεία των 2 πινάκων καλώντας τη μέθοδο showPin() */
```

```
        static void showPin(int pin[] ){
        // Μέθοδος Εμφάνισης στοιχείων πίνακα
            for (int i = 0;i <= pin.length-1;i++)
            {
             System.out.print(pin [i] + " " );
            }
            System.out.println();
         }

        static int[] copyPin1toPin2(int n, int pin1[]){
        // Μέθοδος Αντιγραφής στον πίνακα 2 των στοιχείων του πίνακα 1
            int pin2[] = new int[n];
            if ( pin2.length < pin1.length)
             for (int i = 0;i <= pin2.length -1;i++)
              pin2[i] = pin1[i];
            else
             for (int i = 0;i <= pin1.length -1;i++)
              pin2[i] = pin1[i];
            return pin2;
        }


        public static void main(String[] args) {
            int i, j;

            // Δήλωση - Αρχικοποίηση πίνακα 1
            int pin1[]= {1,2,3,4,5};

            // Εκχώρηση τυχαίας ακέραιας τιμής στο μέγεθος του πίνακα 2
            int n = (int) ( 1 + Math.random()*10);
            System.out.println("n = " + n);

            // Δήλωση πίνακα 2
            int pin2[]= new int[n];

            // Αντιγραφή στον πίνακα 2 των στοιχείων του πίνακα 1
            pin2 = copyPin1toPin2( n,  pin1);

            //  Εμφάνιση στοιχείων πίνακα 1
          System.out.print("Πίνακας 1 = ");
          showPin( pin1);

            // Εμφάνιση στοιχείων πίνακα 2
          System.out.print("Πίνακας 2 = ");
            showPin( pin2);
          }
    }
```

Έξοδος Προγράμματος

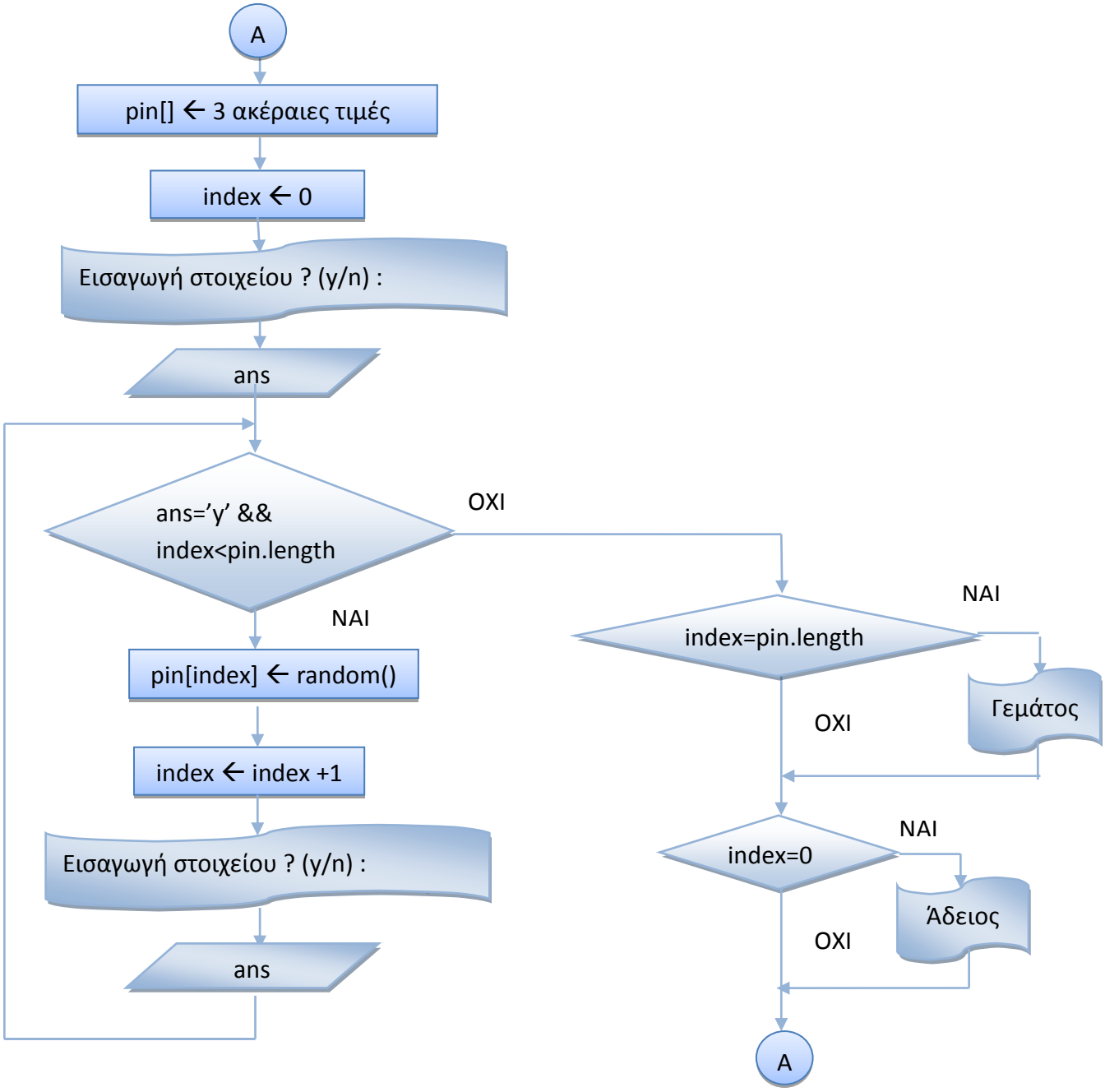
```
run:
n = 2
Πίνακας 1 = 1 2 3 4 5
Πίνακας 2 = 1 2

n = 5
Πίνακας 1 = 1 2 3 4 5
Πίνακας 2 = 1 2 3 4 5

n = 10
Πίνακας 1 = 1 2 3 4 5
Πίνακας 2 = 1 2 3 4 5 0 0 0 0 0
BUILD SUCCESSFUL (total time: 0 seconds)
```

6.5.5 Δυναμικό Γέμισμα ενός Πίνακα – Έλεγχος αν Είναι Άδειος ή Γεμάτος

Σε αρκετές εφαρμογές θα χρειαστεί να γεμίζουμε δυναμικά τις θέσεις ενός πίνακα, οπότε θα πρέπει να ξέρουμε αν ο πίνακας έχει γεμίσει ή αν δεν έχει μπει κανένα στοιχείο.

- Να γραφεί Πρόγραμμα που να δημιουργεί έναν πίνακα που το μέγεθός του δίνεται με μια τυχαία ακέραια τιμή. Για όσο ο χρήστης επιλέγει να βάλει στοιχεία στον πίνακα δημιουργεί τυχαίες ακέραιες τιμές. Όταν ο πίνακας γεμίσει, εμφανίζει το μήνυμα "`Ο Πίνακας είναι γεμάτος`", ενώ αν ο χρήστης επιλέξει να μη βάλει κανένα στοιχείο στον πίνακα, εμφανίζει το μήνυμα "`Ο Πίνακας είναι άδειος`". Η απάντηση του χρήστη δίνεται με την κλήση της μεθόδου `readchar()`.

ΛΟΓΙΚΟ ΔΙΑΓΡΑΜΜΑ



Αλγόριθμος

1. Δήλωση πίνακα `pin[3]` με 3 θέσεις
2. Αρχική τιμή για τη θέση του τελευταίου στοιχείων του πίνακα `pin = 0 ( index = 0)`
3. Εμφάνιση μηνύματος "`Εισαγωγή στοιχείου ? (y/n) : `"
4. Διάβασμα απάντησης του χρήστη με την κλήση της μεθόδου `readchar()`.
5. **Για όσο** η απάντηση του είναι χρήστη είναι `'y' (ans == 'y')` και το `index` δεν είναι ίσο με το μέγεθος του πίνακα (`index < pin.length)`)
   Βάζουμε στη θέση `index` του πίνακα `pin` μια τυχαία ακέραια τιμή
   Αυξάνουμε το `index` κατά `1`
   Διάβασμα νέας απάντησης του χρήστη με την κλήση της μεθόδου `readchar()`.
6. Αν `index = pin.length,` Εμφάνιση μηνύματος "`Ο Πίνακας είναι γεμάτος`"
7. **Αν** `index = 0,` Εμφάνιση μηνύματος "`Ο Πίνακας είναι άδειος`"
   Διαφορετικά
   Εμφάνιση των στοιχείων του πίνακα

Πρόγραμμα

```
public class PinEmptyFull {
/* Το πρόγραμμα δημιουργεί έναν πίνακα 3 στοιχείων. Για όσο ο χρήστης επιλέγει
να βάλει στοιχεία στον πίνακα δημιουργεί τυχαίες ακέραιες τιμές. Όταν ο πίνακας
γεμίσει, εμφανίζει το μήνυμα "Ο Πίνακας είναι γεμάτος", ενώ αν ο χρήστης
επιλέξει να μη βάλει κανένα στοιχείο στον πίνακα, εμφανίζει το μήνυμα "Ο
Πίνακας είναι άδειος". Η απάντηση του χρήστη δίνεται με την κλήση της μεθόδου
readchar().
*/
    static char readchar() throws java.io.IOException{
        // Μήνυμα - Απάντηση για Εισαγωγή νέου στοιχείου
        char ans;
        System.out.print("Εισαγωγή στοιχείου ? (y/n) : ");
        do {
          ans = (char)System.in.read();
        }
        while (ans == '\n' | ans == '\r');
        return ans;
    }

    public static void main(String[] args)
                        throws java.io.IOException{

        // Εκχώρηση ακέραιας τιμής στο μέγεθος του πίνακα 2
        int n = 3;
        System.out.println("n =  " + n  );
        char ans = 'y';

        // Δήλωση πίνακα 2
        int pin[] = new int[n];
        int index = 0;

        // Γέμισμα πίνακα με τυχαίες ακέραιες τιμές

        ans = readchar();
        while ((ans == 'y') && ( index < pin.length)) {

           pin[index] = (int) ( Math.random()*10);
           System.out.println("Στοιχείο " + index + " = " + pin[index] );
           index++;
           // Μήνυμα - Απάντηση για Εισαγωγή νέου στοιχείου
```

```
            ans = readchar();
        }

        if ( index == pin.length)
            System.out.println("Ο Πίνακας είναι γεμάτος");

        if (index == 0)
            System.out.println("Ο Πίνακας είναι άδειος");
        else {
        // Εμφάνιση στοιχείων πίνακα 2
        System.out.print("Πίνακας = ");
        for (int i = 0;i < n;i++)
          System.out.print(pin[i] + " " );
        System.out.println();
        }
        }
    }
```

Έξοδος Προγράμματος :

```
run:
Εισαγωγή στοιχείου ? (y/n) : y
Στοιχείο 0 = 6
Εισαγωγή στοιχείου ? (y/n) : y
Στοιχείο 1 = 9
Εισαγωγή στοιχείου ? (y/n) : y
Στοιχείο 2 = 9
Εισαγωγή στοιχείου ? (y/n) : y
Ο Πίνακας είναι γεμάτος
Πίνακας = 6 9 9


n =  3
Εισαγωγή στοιχείου ? (y/n) : n
Ο Πίνακας είναι άδειος

n =  3
Εισαγωγή στοιχείου ? (y/n) : y
Στοιχείο 0 = 1
Εισαγωγή στοιχείου ? (y/n) : y
Στοιχείο 1 = 5
Εισαγωγή στοιχείου ? (y/n) : n
Πίνακας = 1 5 0
BUILD SUCCESSFUL (total time: 0 seconds)
```

6.6 Πίνακες 2 Διαστάσεων

Πίνακας 2 διαστάσεων είναι μια διάταξη στοιχείων σε **γραμμές** και **στήλες** που το καθένα τους έχει κάποια θέση στην αντίστοιχη γραμμή και στήλη. Καταλαμβάνει μια περιοχή μνήμης με το λογικό όνομα του πίνακα, ενώ κάθε στοιχείο του ξεχωρίζει με **δύο δείκτες**, γραμμή - στήλη.

Παράδειγμα

Πίνακας 2 διαστάσεων, 3 γραμμών και 2 στηλών με περιεχόμενα τους αριθμούς 1-6.

	Στήλη 0	Στήλη 1
Γραμμή 0	1	2
Γραμμή 1	3	4
Γραμμή 2	5	6

Δήλωση Πίνακα 2 διαστάσεων

Η Δήλωση ενός Πίνακα 2 διαστάσεων στη Java γίνεται με τη δήλωση του **τύπου** των στοιχείων που θα αποθηκεύσει, το **όνομά** του, το **μέγεθός** του (αριθμός γραμμών και στηλών σε ξεχωριστά ζεύγη αγκυλών) και τον τελεστή **new**, επειδή οι πίνακες στη Java είναι αντικείμενα.

Παράδειγμα

```
int pin1[][] = new int[3][2];
```

όπου δηλώνουμε τον πίνακα pin1, ο οποίος θα αποθηκεύσει 6 ακέραιους αριθμούς.

Το ίδιο αποτέλεσμα έχουμε με την εντολή

```
int[][] pin1 = new int[3][2];
```

αρκεί να υπάρχουν οι αγκύλες στον τύπο ή στο όνομα του πίνακα.

Ο αριθμός των γραμμών και στηλών του πίνακα μπορεί να είναι οι τιμές δύο μεταβλητών. Οι επόμενες εντολές έχουν το ίδιο αποτέλεσμα :

```
int m = 3, n = 2;
int pin1[][] = new int[m][n];
```

Δημιουργία πίνακα 2 διαστάσεων με δυναμική αρχικοποίηση

Ένας πίνακας 2 διαστάσεων μπορεί να δημιουργηθεί με **δυναμική αρχικοποίηση**, όπου οι τιμές του περικλείονται σε ζεύγη αγκίστρων για την κάθε γραμμή και χωρίζονται με κόμμα. Σ' αυτή την περίπτωση **δε** χρειάζεται ο τελεστής **new**.

Παράδειγμα

```
int pin2[][] = {{1, 2}, {3, 4}, {5, 6}};
```

όπου δηλώνουμε τον πίνακα pin2, ο οποίος θα αποθηκεύσει τους ακέραιους αριθμούς από το 1 μέχρι το 6 σε 3 γραμμές από 2 στοιχεία η καθεμιά.

Πρόσβαση σε κάποιο στοιχείο πίνακα 2 διαστάσεων

Η **πρόσβαση** σε κάποιο στοιχείο του πίνακα γίνεται με το όνομά του και τη θέση του (δύο δείκτες γραμμής και στήλης) μέσα σε αγκύλες. Π.χ. το pin1[2][1] αναφέρεται στο στοιχείο του pin1 που βρίσκεται στην τρίτη γραμμή και δεύτερη στήλη, δηλαδή το 6.

Ακανόνιστοι Πίνακες

Στους πίνακας 2 διαστάσεων είναι **υποχρεωτική η δήλωση ΜΟΝΟ της 1^{ης} διάστασης** (αριθμός γραμμών). Μ' αυτό τον τρόπο μπορούμε να έχουμε πίνακες με διαφορετικό αριθμό στοιχείων στην κάθε στήλη (ακανόνιστοι πίνακες).

Παράδειγμα

Με τις παρακάτω εντολές :

```
int pin4[][] = new int[3][];
pin4[0] = new int[3];
pin4[1] = new int[2];
pin4[2] = new int[1];
```

δηλώνουμε τον πίνακα pin4, ο οποίος θα αποθηκεύσει ακέραιους αριθμούς σε 3 γραμμές, με 3 στοιχεία στην πρώτη γραμμή, 2 στοιχεία στη δεύτερη γραμμή και 1 στοιχείο στην τρίτη γραμμή.

Στο επόμενο παράδειγμα δηλώνεται ένας ακανόνιστος πίνακας με δυναμική αρχικοποίηση :

Παράδειγμα

```
int pin3[][] = {{1, 2, 3}, {4, 5}, {6}};
```

όπου δηλώνουμε τον πίνακα pin3, ο οποίος θα αποθηκεύσει τους ακέραιους αριθμούς από το 1 μέχρι το 6 σε 3 γραμμές, με 3 στοιχεία στην πρώτη γραμμή, 2 στοιχεία στη δεύτερη γραμμή και 1 στοιχείο στην τρίτη γραμμή.

6.6.1 Παράδειγμα Δημιουργίας – Γεμίσματος 2 πινάκων 2 διαστάσεων με random τιμές και Δυναμική Αρχικοποίηση

- Να γραφεί Αλγόριθμος/Πρόγραμμμα το οποίο γεμίζει έναν πίνακα 2 διαστάσεων (3x2), με **δυναμική αρχικοποίηση** και δίνει τις τιμές από το 1 μέχρι το 6, δηλώνει έναν άλλον πίνακα (3x2), θέσεων, τον οποίο γεμίζει με τυχαίες τιμές και εμφανίζει τα περιεχόμενα των 2 πινάκων.

Αλγόριθμος

1. Δήλωση - Αρχικοποίηση πίνακα pin1
2. Δήλωση - Γέμισμα πίνακα pin2 με τυχαίες τιμές
   Για κάθε γραμμή i = 0 μέχρι (n – 1)
      Για κάθε στήλη j = 0 μέχρι (n – 1)
         Εκχωρούμε μια τυχαία τιμή μεταξύ 0 και 10 στο στοιχείο pin2[i][j]
3. Εμφάνιση στοιχείων πίνακα pin1
   Για κάθε γραμμή i = 0 μέχρι (n – 1)
      Για κάθε στήλη j = 0 μέχρι (n – 1)
         Εμφανίζουμε το στοιχείο pin1[i]
4. Εμφάνιση στοιχείων πίνακα pin2
   Για κάθε γραμμή i = 0 μέχρι (n – 1)
      Για κάθε στήλη j = 0 μέχρι (n – 1)
         Εμφανίζουμε το στοιχείο pin2[i]

Πρόγραμμα

```
public class Pin2 {
/*
    Το πρόγραμμα δημιουργεί 2 πίνακες 2 διαστάσεων ( 3x2 ), τον πρώτο με
δυναμική αρχικοποίηση και τον γεμίζει με τις ακέραιες τιμές από 1 μέχρι 6,
ενώ το δεύτερο τον γεμίζει με τυχαίες τιμές από 0 μέχρι 10 και εμφανίζει
τα στοιχεία των 2 πινάκων.
    */
    public static void main(String[] args) {
        int i, j;

        // Δήλωση - Αρχικοποίηση πίνακα 1
        int pin1[][] = {{1,2},{3,4},{5,6}};

        // Εκχώρηση ακέραιας τιμής στα μεγέθη γραμμών-στηλών του πίνακα 2
        int m = 3, n = 2;

        // Δήλωση πίνακα 2
        int pin2[][] = new int[m][n];

        // Γέμισμα πίνακα 2 με τυχαίες ακέραιες τιμές
        for (i = 0;i < m;i++)
         for (j = 0;j < n;j++)
           pin2[i][j] = (int) ( Math.random()*10);


        // Εμφάνιση στοιχείων πίνακα 1
        System.out.println("Πίνακας 1");
        for (i = 0;i < 3;i++)
```

```
        {
         for (j = 0;j < 2;j++)
          System.out.print(pin1[i][j] + " " );
        System.out.println();
        }
        System.out.println();

        // Εμφάνιση στοιχείων πίνακα 2
        System.out.println("Πίνακας 2");
        for (i = 0;i < m;i++)
        {
         for (j = 0;j < n;j++)
          System.out.print(pin2[i][j] + " " );
        System.out.println();
        }
     }
}
```

Έξοδος Προγράμματος :

```
run:
Πίνακας 1
1 2
3 4
5 6

Πίνακας 2
7 5
3 6
6 0
BUILD SUCCESSFUL (total time: 0 seconds)
```

6.6.2 Παράδειγμα Δημιουργίας – Γεμίσματος 2 πινάκων 2 διαστάσεων με random τιμές και Δυναμική Αρχικοποίηση με τη χρήση μεθόδων

- Να γραφεί Αλγόριθμος/Πρόγραμμμα το οποίο γεμίζει έναν πίνακα 2 διαστάσεων (3x2), με **δυναμική αρχικοποίηση** και δίνει τις τιμές από το 1 μέχρι το 6, δηλώνει έναν άλλον πίνακα (3x2), θέσεων, τον οποίο γεμίζει με τυχαίες τιμές και εμφανίζει τα περιεχόμενα των 2 πινάκων καλώντας μεθόδους για το γέμισμα του πίνακα των τυχαίων τιμών και την εμφάνιση των 2 πινάκων.

Αλγόριθμος

1. Δήλωση - Αρχικοποίηση πίνακα 1
2. Δήλωση - Γέμισμα πίνακα 2 με τυχαίες τιμές καλώντας τη μέθοδο `fillPin2()`
3. Εμφάνιση στοιχείων πίνακα 1 καλώντας τη μέθοδο `showPin1(pin1)`
4. Εμφάνιση στοιχείων πίνακα 2 καλώντας τη μέθοδο `showPin2(m, n, pin2)`

Πρόγραμμα

```
public class Pin2Methods {
/* Το πρόγραμμα δημιουργεί 2 πίνακες 2 διαστάσεων ( 3x2 ), τον πρώτο με
δυναμική αρχικοποίηση και τον γεμίζει με τις ακέραιες τιμές από 1 μέχρι 6,
ενώ το δεύτερο τον γεμίζει με τυχαίες τιμές από 0 μέχρι 10 καλώντας τη
μέθοδο fillPin2() και εμφανίζει τα στοιχεία των 2 πινάκων καλώντας τις
μεθόδους  showPin1(pin1) και showPin2(m, n, pin2)   */

    static void showPin1(int pin1[][] ){
    // Μέθοδος Εμφάνισης στοιχείων πίνακα 1
        System.out.println("Πίνακας 1 = ");
        for (int i = 0;i < 3;i++) {
         for (int j = 0;j < 2;j++)
          System.out.print(pin1[i][j] + " " );
         System.out.println();
        }
     }

    static void fillPin2(int m, int n, int pin2[][] ){
    // Μέθοδος δημιουργίας στοιχείων πίνακα1, πέρασμα μεταβλητής pin2 με
αναφορά
        for (int i = 0;i < m;i++)
         for (int j = 0;j < n;j++)
           pin2[i][j] = (int) ( Math.random()*10);
        }

    static void showPin2(int m, int n, int pin2[][] ){
    // Μέθοδος Εμφάνισης στοιχείων πίνακα 2
        System.out.println("Πίνακας 2 = ");
        for (int i = 0;i < 3;i++) {
         for (int j = 0;j < 2;j++)
          System.out.print(pin2[i][j] + " " );
         System.out.println();
        }
     }

    public static void main(String[] args) {
        int i, j;

        // Δήλωση - Αρχικοποίηση πίνακα 1
        int pin1[][] = {{1,2},{3,4},{5,6}};
```

```
        // Εκχώρηση ακέραιας τιμής στα μεγέθη γραμμών-στηλών του πίνακα 2
        int m = 3, n = 2;

        // Δήλωση πίνακα 2
        int pin2[][] = new int[m][n];

        // Γέμισμα πίνακα 2 με τυχαίες ακέραιες τιμές
        fillPin2( m,  n,  pin2);


        // Εμφάνιση στοιχείων πίνακα 1
        showPin1(pin1);
        System.out.println();

        // Εμφάνιση στοιχείων πίνακα 2
        showPin2( m,  n,  pin2);
    }
}
```

Έξοδος Προγράμματος :

```
run:
Πίνακας 1 =
1 2
3 4
5 6

Πίνακας 2 =
5 3
6 6
1 6
BUILD SUCCESSFUL (total time: 0 seconds)
```

6.6.3 Το length για τους Πίνακες 2 Διαστάσεων

Το `length` και για τους πίνακες 2 διαστάσεων παίρνει την τιμή του **αριθμού** των **γραμμών** του πίνακα που δηλώνουμε με `new` ή με δυναμική αρχικοποίηση. Στους ακανόνιστους πίνακες, το όνομα του πίνακα με τον αριθμό γραμμής και το `length` δίνει τον αριθμό των **στοιχείων** της γραμμής.

Παράδειγμα 1

```
int pin3[][] = {{1,2},{3,4},{5,6}}; // pin3.length = 3
```

Παράδειγμα 2

```
int pin4[][] = new int[4][];        // pin4.length = 4
pin4[0] = new int[31];              // pin4[0].length = 31
pin4[1] = new int[28];              // pin4[1].length = 28
pin4[2] = new int[31];              // pin4[2].length = 31
pin4[3] = new int[30];              // pin4[3].length = 30
```

6.7 Μέθοδοι Χειρισμού Πινάκων των Κλάσεων System και Arrays

Η Java περιέχει τις κλάσεις `System` και `Arrays`, οι οποίες περιέχουν μεθόδους γεμίσματος, αντιγραφής, σύγκρισης και ταξινόμησης πινάκων.

Η κλάση `Arrays` περιέχει τις παρακάτω μεθόδους :

`Arrays.fill(<όνομα_πίνακα>,<τιμή>)`

η οποία γεμίζει όλες τις θέσεις ενός πίνακα `<όνομα_πίνακα>` με την τιμή `<τιμή>`

`Arrays.equals(<όνομα_πίνακα_1>,<όνομα_πίνακα_2>)`

η οποία ελέγχει αν οι πίνακες `<όνομα_πίνακα_1>` και `<όνομα_πίνακα_2>` έχουν τα ίδια περιεχόμενα

`Arrays.sort(<όνομα_πίνακα>)`

η οποία ταξινομεί τα στοιχεία του πίνακα `<όνομα_πίνακα>`

ενώ η κλάση `System` περιέχει τη μέθοδο :

`System.arraycopy(<όνομα_πίνακα_1>, <αρχή_1>,>,<όνομα_πίνακα_2>,>, <αρχή_2>,<αριθμός_στοιχείων>)`

η οποία αντιγράφει όσα στοιχεία θέλουμε `<αριθμός_στοιχείων>` του πίνακα `<όνομα_πίνακα_1>` αρχίζοντας από τη θέση `<αρχή_1>` στον πίνακα `<όνομα_πίνακα_2>` αρχίζοντας από τη θέση `<αρχή_2>`

Θα πρέπει να γίνει εισαγωγή της βιβλιοθήκης `java.util.Arrays` με την εντολή

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

6.7.1 Παράδειγμα Χρήσης Μεθόδων Χειρισμού Πινάκων των Κλάσεων System και Arrays

- Να γραφεί Πρόγραμμα που να γεμίζει έναν πίνακα `pin1 5` θέσεων με το `-999` με την κλήση της μεθόδου `Arrays.fill()`, γεμίζει έναν πίνακα `pin2 5` θέσεων με τυχαίες ακέραιες τιμές, εμφανίζει τα περιεχόμενα των `2` πινάκων και ελέγχει αν είναι ίδια με την κλήση της μεθόδου `Arrays.equals()`, Ταξινομεί τα στοιχεία του πίνακα `2` με την κλήση της μεθόδου `Arrays.sort()`, Αντιγράφει τα περιεχόμενα του πίνακα `2` στον πίνακα `1` με την κλήση της μεθόδου `System.arraycopy()`, εμφανίζει τα περιεχόμενα των `2` πινάκων και ελέγχει αν είναι ίδια με την κλήση της μεθόδου `Arrays.equals()`.

Αλγόριθμος

1. Γέμισμα ενός πίνακα `pin1 5` θέσεων με το `-999` με την κλήση της μεθόδου `Arrays.fill()`
2. Γέμισμα ενός πίνακα `pin2` 5 θέσεων με με τυχαίες ακέραιες τιμές
3. Εμφάνιση των στοιχείων του πίνακα `pin1`
4. Εμφάνιση των στοιχείων του πίνακα `pin2`
5. Έλεγχος αν τα περιεχόμενα των `2` πινάκων είναι ίδια με την κλήση της μεθόδου `Arrays.equals()`
6. Ταξινόμηση των στοιχείων του πίνακα 2 με την κλήση της μεθόδου `Arrays.sort()`
7. Εμφάνιση των στοιχείων του πίνακα `pin2`
8. Αντιγραφή των περιεχομένων του πίνακα `2` στον πίνακα `1` με την κλήση της μεθόδου `System.arraycopy()`
9. Εμφάνιση των στοιχείων του πίνακα `pin1`
10. Έλεγχος αν τα περιεχόμενα των `2` πινάκων είναι ίδια με την κλήση της μεθόδου `Arrays.equals()`

Πρόγραμμα

```
import java.util.Arrays;

public class PinArrays {
/* Πρόγραμμα που γεμίζει έναν πίνακα pin1 5 θέσεων με το -999 με την κλήση
της μεθόδου Arrays.fill, γεμίζει έναν πίνακα pin2 5 θέσεων με τυχαίες
ακέραιες τιμές, εμφανίζει τα περιεχόμενα των 2 πινάκων και ελέγχει αν είναι
ίδια με την κλήση της μεθόδου Arrays.equals, Ταξινομεί τα στοιχεία του πίνακα 2
με την κλήση της μεθόδου Arrays.sort, Αντιγράφει τα περιεχόμενα του πίνακα 2
στον πίνακα 1 με την κλήση της μεθόδου System.arraycopy, εμφανίζει τα
περιεχόμενα των 2 πινάκων και ελέγχει αν είναι ίδια με την κλήση της μεθόδου
Arrays.equals
 */
    public static void main(String[] args) {
        int i, j;

        // Δήλωση - Αρχικοποίηση πίνακα 1 - Arrays.fill
        int pin1[]= new int[5];
        Arrays.fill(pin1, -999);

        // Δήλωση - Αρχικοποίηση πίνακα 2 - Math.random
        int pin2[]= new int[5];
```

```
        for (i = 0;i < pin2.length;i++)
          pin2[i] = (int) (Math.random()*10);

        // Εμφάνιση στοιχείων πίνακα 1
        System.out.print("Πίνακας 1 = ");
        for (i = 0;i < pin1.length;i++)
          System.out.print(pin1[i] + " " );
        System.out.println();

        // Εμφάνιση στοιχείων πίνακα 2
        System.out.print("Πίνακας 2 = ");
        for (i = 0;i < pin2.length;i++)
          System.out.print(pin2[i] + " " );
        System.out.println();

        // Έλεγχος αν οι Πίνακες 1,2 έχουν ίδια περιεχόμενα Arrays.equals
        if ( Arrays.equals(pin1, pin2))
          System.out.println("Οι πίνακες 1, 2 έχουν τα ίδια περιεχόμενα" );
        else
          System.out.println("Οι πίνακες 1,2 δεν έχουν τα ίδια "
                  + "περιεχόμενα" );

        // Ταξινόμηση των στοιχείων του πίνακα 2 - Arrays.sort
        Arrays.sort(pin2);

         // Εμφάνιση στοιχείων πίνακα 2
        System.out.print("\nΠίνακας 2 μετά την Ταξινόμηση = ");
        for (i = 0;i < pin2.length;i++)
          System.out.print(pin2[i] + " " );
        System.out.println();

        // Αντιγραφή των στοιχείων του πίνακα 2 στον πίνακα 1
        // System.arraycopy
        System.arraycopy(pin2, 0, pin1, 0, 5);

        // Εμφάνιση στοιχείων πίνακα 1
        System.out.print("Πίνακας 1 μετά την Αντιγραφή = ");
        for (i = 0;i < pin1.length;i++)
          System.out.print(pin1[i] + " " );
        System.out.println();

        // Έλεγχος αν οι Πίνακες 1,2 έχουν  ίδια περιεχόμενα Arrays.equals
        if ( Arrays.equals(pin1, pin2))
          System.out.println("Οι πίνακες 1, 2 έχουν τα ίδια περιεχόμενα" );
        else
          System.out.println("Οι πίνακες 1, 2 δεν έχουν τα ίδια "
                  + "περιεχόμενα" );

      }
}
```

Έξοδος Προγράμματος :

```
run:
Πίνακας 1 = -999 -999 -999 -999 -999
Πίνακας 2 = 0 6 2 7 4
Οι πίνακες 1, 2 δεν έχουν τα ίδια περιεχόμενα

Πίνακας 2 μετά την Ταξινόμηση = 0 2 4 6 7
Πίνακας 1 μετά την Αντιγραφή = 0 2 4 6 7
Οι πίνακες 1, 2 έχουν τα ίδια περιεχόμενα
BUILD SUCCESSFUL (total time: 0 seconds)
```