# 7 ΚΛΑΣΕΙΣ – ΑΝΤΙΚΕΙΜΕΝΑ

- Να γραφεί Αλγόριθμος/Πρόγραμμμα το οποίο δίνει τιμές στον Αριθμό Μητρώου, τον Κωδικό Μαθήματος, το Βαθμό Θεωρίας και το Βαθμό Εργαστηρίου για 2 φοιτητές και Υπολογίζει και Εμφανίζει τον Τελικό Βαθμό του κάθε φοιτητή.

**Αλγόριθμος**

1. Εισαγωγή Στοιχείων 1ου φοιτητή
2. Υπολογισμός - Εμφάνιση Τελικού Βαθμού 1ου φοιτητή
3. Εισαγωγή Στοιχείων 2ου φοιτητή
4. Υπολογισμός - Εμφάνιση Τελικού Βαθμού 2ου φοιτητή

**Πρόγραμμα**

```
public class Objects1 {
/*    Πρόγραμμμα το οποίο δίνει τιμές στον Αριθμό Μητρώου, τον Κωδικό
Μαθήματος,το Βαθμό Θεωρίας και το Βαθμό Εργαστηρίου για 2 φοιτητές και
Υπολογίζει και Εμφανίζει τον Τελικό Βαθμό του κάθε φοιτητή */
    public static void main(String[] args) {
    int AM;          // Αριθμός Μητρώου Φοιτητή
    int KM;          // Κωδικός Μαθήματος
    double Theory;   // Βαθμός Θεωρίας
    double Erg;      // Βαθμός Εργαστηρίου
    double Telikos; // Τελικός Βαθμός

    // Εισαγωγή Στοιχείων 1ου φοιτητή
    AM = 14013;
    KM = 6000232;
    Theory = 6.5;
    Erg = 8.2;
    // Υπολογισμός - Εμφάνιση Τελικού Βαθμού 1ου φοιτητή
    Telikos = Theory * 0.6 + Erg * 0.4;
    System.out.println ( "Τελικός Βαθμός Φοιτητή 1 = " + Telikos );

    // Εισαγωγή Στοιχείων 2ου φοιτητή
    AM = 14014;
    KM = 6000232;
    Theory = 8.6;
    Erg = 8.0;
    // Υπολογισμός - Εμφάνιση Τελικού Βαθμού 1ου φοιτητή
    Telikos = Theory * 0.6 + Erg * 0.4;
    System.out.println ( "Τελικός Βαθμός Φοιτητή 2 = " + Telikos );
   }
}
```

**Έξοδος Προγράμματος :**

```
run:
Τελικός Βαθμός Φοιτητή 1 = 7.18
Τελικός Βαθμός Φοιτητή 2 = 8.36
BUILD SUCCESSFUL (total time: 0 seconds)
```

## 7.1 Δημιουργία Κλάσης η οποία περιέχει ΜΟΝΟ Δεδομένα σαν μέλη

Στο προηγούμενο πρόγραμμα, όλα τα στοιχεία Αριθμός Μητρώου, Κωδικός Μαθήματος, Βαθμός Θεωρίας, Βαθμός Εργαστηρίου και Τελικός Βαθμός αφορούν τους φοιτητές. Θα μπορούσαμε να ορίσουμε μια κατηγορία, κλάση με το όνομα `Foititis`, η οποία να περιέχει τα παραπάνω στοιχεία και να δημιουργούμε στιγμιότυπα – αντικείμενα, τα οποία και έχουν **φυσική υπόσταση**, καταλαμβάνουν μνήμη.

Κλάση είναι ένα **αφηρημένο πρότυπο**, το οποίο περιγράφει – καθορίζει τη μορφή ενός αντικειμένου. Μπορεί εκτός από δεδομένα να περιέχει και μεθόδους που ενεργούν πάνω στα δεδομένα. Η κλάση ορίζει ένα **ΝΕΟ** τύπο δεδομένων.

Τα δεδομένα και οι μέθοδοι πρόσβασης στα δεδομένα λέγονται **μέλη** της κλάσης. Η κλάση αποκτά πραγματική υπόσταση με τη δυναμική δημιουργία του αντικειμένου.

### Ορισμός Κλάσης

```
public class <όνομα_κλάσης> {
Δηλώσεις μεταβλητών
...
Μέθοδοι
...
}
```

**Παράδειγμα**

```
public class Foititis {
      int AM;          // Αριθμός Μητρώου Φοιτητή
      int KM;          // Κωδικός Μαθήματος
      double Theory;   // Βαθμός Θεωρίας
      double Erg;      // Βαθμός Εργαστηρίου
}
```

Η κλάση `Foititis` περιέχει σαν μέλη – δεδομένα ΜΟΝΟ τις μεταβλητές που χρησιμοποιήσαμε στο προηγούμενο πρόγραμμα και αφορούν τον κάθε φοιτητή.

### Δημιουργία Αντικειμένου

Για τη δημιουργία ενός αντικειμένου πρέπει να δηλώσουμε ότι το αντικείμενο που θα δημιουργήσουμε είναι τύπου της κλάσης στην οποία ανήκει και να το δημιουργήσουμε με τον τελεστή **new**. Ο γενικός τύπος για τη δημιουργία του είναι :

```
<όνομα_ κλάσης>  <όνομα_αντικειμένου>;
<όνομα_αντικειμένου> = new <όνομα_κλάσης>();
```

**Παράδειγμα**

```
Foititis f;
f = new Foititis ();
```

Η πρώτη εντολή δηλώνει πως το **f** είναι τύπου `Foititis` ( Δήλωση Αναφοράς Αντικειμένου ). Με τη δεύτερη εντολή **δημιουργείται** το αντικείμενο - στιγμιότυπο της κλάσης `Foititis`.

Το ίδιο αποτέλεσμα θα έχουμε και με την εντολή :

```
Foititis f = new Foititis ();
```

Το αντικείμενο στο οποίο αναφέρεται το `f` θα περιέχει **τα δικά του αντίγραφα** των μεταβλητών και μεθόδων ( **μελών** ) της κλάσης `Foititis`.

## Πρόσβαση στα μέλη του αντικειμένου

Η πρόσβαση στα μέλη του αντικειμένου γίνεται με τον τελεστή **'.'**. Η γενική της μορφή είναι `<όνομα_αντικειμένου>.<μέλος>`.

**Παράδειγμα**

Με την εντολή

```
f.AM = 14013;
```

δίνουμε την τιμή `14013` στο πεδίο `AM` ( Αριθμός Μητρώου ) του αντικειμένου `f`.

### 7.1.1 Παράδειγμα Δημιουργίας Κλάσης η οποία περιέχει ΜΟΝΟ Δεδομένα σαν μέλη

- Να γραφεί Αλγόριθμος/Πρόγραμμα το οποίο δημιουργεί `2` αντικείμενα τύπου `Foititis`, δίνει τιμές στον Αριθμό Μητρώου, τον Κωδικό Μαθήματος, το Βαθμό Θεωρίας και το Βαθμό Εργαστηρίου για τους `2` φοιτητές και Υπολογίζει και Εμφανίζει τον Τελικό Βαθμό του κάθε φοιτητή.

#### Κλάση `Foititis`

Η Κλάση `Foititis` περιέχει τα πεδία ( μέλη ) Αριθμός Μητρώου Φοιτητή, Κωδικός Μαθήματος, Βαθμός Θεωρίας και Βαθμός Εργαστηρίου.

```
public class Foititis {
    int AM;          // Αριθμός Μητρώου Φοιτητή
    int KM;          // Κωδικός Μαθήματος
    double Theory;   // Βαθμός Θεωρίας
    double Erg;      // Βαθμός Εργαστηρίου
}
```

## Αλγόριθμος κλάσης `Objects2 – main()`

1. Δημιουργία Αντικειμένου Φοιτητή 1
2. Δημιουργία Αντικειμένου Φοιτητή 2
3. Εισαγωγή Στοιχείων 1ου φοιτητή
4. Υπολογισμός - Εμφάνιση Τελικού Βαθμού 1ου φοιτητή
5. Εισαγωγή Στοιχείων 2ου φοιτητή
6. Υπολογισμός - Εμφάνιση Τελικού Βαθμού 2ου φοιτητή

### Πρόγραμμα

```
public class Objects2 {
/*  Πρόγραμμμα το οποίο δημιουργεί 2 αντικείμενα τύπου Foititis, δίνει τιμές
στον Αριθμό Μητρώου, τον Κωδικό Μαθήματος, το Βαθμό Θεωρίας και το Βαθμό
Εργαστηρίου για τους 2 φοιτητές και Υπολογίζει και Εμφανίζει τον Τελικό Βαθμό
του κάθε φοιτητή
*/
    public static void main(String[] args) {
    double Telikos;

    // Δημιουργία Αντικειμένου Φοιτητή 1
    Foititis f1 = new Foititis();

    // Δημιουργία Αντικειμένου Φοιτητή 2
    Foititis f2;
    f2 = new Foititis();

    // Εισαγωγή Στοιχείων Φοιτητή 1
    f1.AM = 14013;
    f1.KM = 6000232;
    f1.Theory = 6.5;
    f1.Erg = 8.2;

    // Υπολογισμός - Εμφάνιση Τελικού Βαθμού Φοιτητή 1
    Telikos = f1.Theory * 0.6 + f1.Erg * 0.4;
    System.out.println ("Τελικός Βαθμός Φοιτητή f1 = " + Telikos );

    // Εισαγωγή Στοιχείων Φοιτητή 2
    f2.AM = 14014;
    f2.KM = 6000232;
    f2.Theory = 8.6;
    f2.Erg = 8.0;

    // Υπολογισμός - Εμφάνιση Τελικού Βαθμού Φοιτητή 2
    Telikos = f2.Theory * 0.6 + f2.Erg * 0.4;
    System.out.println ("Τελικός Βαθμός Φοιτητή f2 = " + Telikos );
   }
}
```

### Έξοδος Προγράμματος :

```
run:
Τελικός Βαθμός Φοιτητή f1 = 7.18
Τελικός Βαθμός Φοιτητή f2 = 8.36
BUILD SUCCESSFUL (total time: 0 seconds)
```

## 7.2 Επεξεργασία Δεδομένων της Κλάσης η οποία περιέχει Δεδομένα σαν μέλη με Μεθόδους στη `main()`

Τα δεδομένα μιας κλάσης, αφού είναι προσβάσιμα στην κλάση που βρίσκεται η `main()` μπορούμε να τα επεξεργαστούμε με μεθόδους, οι οποίες βρίσκονται στην ίδια κλάση που βρίσκεται η `main()`.

### 7.2.1 Παράδειγμα Δημιουργίας Κλάσης η οποία περιέχει Δεδομένα και επεξεργασία τους με Μεθόδους στην κλάση που περιέχει τη `main()`

- Να γραφεί Αλγόριθμος/Πρόγραμμα το οποίο δημιουργεί 2 αντικείμενα τύπου `Foititis`, δίνει τιμές στον Αριθμό Μητρώου, τον Κωδικό Μαθήματος, το Βαθμό Θεωρίας και το Βαθμό Εργαστηρίου για τους 2 φοιτητές και Υπολογίζει και Εμφανίζει τον Τελικό Βαθμό του κάθε φοιτητή **με την κλήση των μεθόδων `findTelikos1()` και `findTelikos2()`** που έχουν οριστεί στην κλάση που βρίσκεται η `main()`. Η μέθοδος `findTelikos1()` θα **υπολογίζει** και θα **εμφανίζει** τον Τελικό Βαθμό του 1ου φοιτητή, ενώ η μέθοδος `findTelikos2()` θα **υπολογίζει** και θα **επιστρέφει** στη `main()` τον Τελικό Βαθμό του 2ου φοιτητή, τον οποίο και θα εμφανίζει.

#### Κλάση `Foititis`

Η Κλάση `Foititis` περιέχει :

- Τα πεδία ( μέλη της κλάσης `Foititis`) Αριθμός Μητρώου Φοιτητή, Κωδικός Μαθήματος, Βαθμός Θεωρίας και Βαθμός Εργαστηρίου.

```
public class Foititis {
    int AM;          // Αριθμός Μητρώου Φοιτητή
    int KM;          // Κωδικός Μαθήματος
    double Theory;   // Βαθμός Θεωρίας
    double Erg;      // Βαθμός Εργαστηρίου
```

#### Κλάση `Objects2aMethods`

Η Κλάση `Objects2aMethods` περιέχει :

- **Τη Μέθοδο `main()`.**
- **Τη Μέθοδο Υπολογισμού - Εμφάνισης Τελικού Βαθμού `findTelikos1()`.**
- **Τη Μέθοδο Υπολογισμού - Επιστροφής Τελικού Βαθμού `findTelikos2()`.**

## Αλγόριθμος `main()` κλάσης `Objects2amethods`

1. Δημιουργία Αντικειμένου Φοιτητή 1
2. Εισαγωγή Στοιχείων 1ου φοιτητή
3. **Υπολογισμός - Εμφάνιση Τελικού Βαθμού 1ου φοιτητή με κλήση μεθόδου `findTelikos1()`**
4. Δημιουργία Αντικειμένου Φοιτητή 2
5. Εισαγωγή Στοιχείων 2ου φοιτητή
6. **Υπολογισμός - Εμφάνιση Τελικού Βαθμού 2ου φοιτητή με κλήση μεθόδου `findTelikos2()`**

### Πρόγραμμα

```
public class Objects2aMethods {
/*  Πρόγραμμμα το οποίο δημιουργεί 2 αντικείμενα τύπου Foititis, δίνει τιμές
στον Αριθμό Μητρώου, τον Κωδικό Μαθήματος, το Βαθμό Θεωρίας και το Βαθμό
Εργαστηρίου για τους 2 φοιτητές και Υπολογίζει και Εμφανίζει τον Τελικό Βαθμό
του κάθε φοιτητή με την κλήση των μεθόδων findTelikos1() και findTelikos2()*/
    // Μέθοδος Υπολογισμού - Εμφάνισης Τελικού Βαθμού
    static void findTelikos1(Foititis f){
          System.out.println( "Τελικός Βαθμός Φοιτητή = "
                + (f.Theory * 0.6 + f.Erg * 0.4));
    }

    // Μέθοδος Υπολογισμού - Επιστροφής Τελικού Βαθμού
    static double findTelikos2(Foititis f){
        return f.Theory * 0.6 + f.Erg * 0.4;
    }
    public static void main(String[] args) {

    // Δημιουργία Αντικειμένου Φοιτητή 1
    Foititis f1 = new Foititis();

    // Εισαγωγή Στοιχείων Φοιτητή 1
    f1.AM = 14013;
    f1.KM = 6000232;
    f1.Theory = 6.5;
    f1.Erg = 8.2;

// Υπολογισμός-Εμφάνιση Τελικού Βαθμού Φοιτητή 1 - κλήση μεθόδου findTelikos1
    findTelikos1(f1);

    // Δημιουργία Αντικειμένου Φοιτητή 2
    Foititis f2 = new Foititis();

    // Εισαγωγή Στοιχείων Φοιτητή 2
    f2.AM = 14014;
    f2.KM = 6000232;
    f2.Theory = 8.6;
    f2.Erg = 8.0;

// Υπολογισμός-Εμφάνιση Τελικού Βαθμού Φοιτητή 2 - κλήση μεθόδου findTelikos2
    System.out.println ("Τελικός Βαθμός Φοιτητή f2 = " + findTelikos2(f2) );
    }
}
```

### Έξοδος Προγράμματος :

```
run:
Τελικός Βαθμός Φοιτητή = 7.18
Τελικός Βαθμός Φοιτητή f2 = 8.36
BUILD SUCCESSFUL (total time: 0 seconds)
```

## 7.3 Δημιουργία Κλάσης η οποία περιέχει Δεδομένα και Μεθόδους σαν μέλη

Στο προηγούμενο παράδειγμα, οι μέθοδοι **`findTelikos1()`** και **`findTelikos2()`** επεξεργάζονται άμεσα **δεδομένα - μέλη** της κλάσης `Foititis` . Η `Java` δίνει τη δυνατότητα στην κλάση, εκτός από δεδομένα, να μπορεί να περιέχει και μεθόδους, οι οποίες ενεργούν πάνω στα δεδομένα, όπως φαίνεται στο επόμενο παράδειγμα. Ο κώδικας των μεθόδων γράφεται **μετά** τα δεδομένα. Τα δεδομένα είναι **ορατά** μέσα στην κλάση, οπότε **δεν απαιτείται** η χρήση του τελεστή '.', ο οποίος χρησιμοποιείται για την πρόσβαση σε κάποιο μέλος της κλάσης.

### 7.3.1 Παράδειγμα Δημιουργίας Κλάσης η οποία περιέχει Δεδομένα και Μεθόδους σαν μέλη

- Να γραφεί Αλγόριθμος/Πρόγραμμα το οποίο δημιουργεί 2 αντικείμενα τύπου `Foititis`, δίνει τιμές στον Αριθμό Μητρώου, τον Κωδικό Μαθήματος, το Βαθμό Θεωρίας και το Βαθμό Εργαστηρίου για τους 2 φοιτητές και Υπολογίζει και Εμφανίζει τον Τελικό Βαθμό του κάθε φοιτητή **με την κλήση των μεθόδων `findTelikos1()` και `findTelikos2()`** που έχουν οριστεί στην κλάση `Foititis`. Η μέθοδος `findTelikos1()` θα **υπολογίζει** και θα **εμφανίζει** τον Τελικό Βαθμό του 1ου φοιτητή, ενώ η μέθοδος `findTelikos2()` θα **υπολογίζει** και θα **επιστρέφει** τον Τελικό Βαθμό του 2ου φοιτητή, τον οποίο και θα εμφανίζει.

#### Κλάση `Foititis`

Η Κλάση **`Foititis`** περιέχει :

- Τα πεδία ( μέλη ) Αριθμός Μητρώου Φοιτητή, Κωδικός Μαθήματος, Βαθμός Θεωρίας και Βαθμός Εργαστηρίου.
- **Τη Μέθοδο Υπολογισμού - Εμφάνισης Τελικού Βαθμού `findTelikos1()` .**
- **Τη Μέθοδο Υπολογισμού - Επιστροφής Τελικού Βαθμού `findTelikos2()` .**

```
public class Foititis {
    int AM;          // Αριθμός Μητρώου Φοιτητή
    int KM;          // Κωδικός Μαθήματος
    double Theory;   // Βαθμός Θεωρίας
    double Erg;      // Βαθμός Εργαστηρίου

    // Μέθοδος Υπολογισμού - Εμφάνισης Τελικού Βαθμού
    void findTelikos1(){
        System.out.println( "Τελικός Βαθμός Φοιτητή = "
                                + (Theory * 0.6 + Erg * 0.4));
    }

    // Μέθοδος Υπολογισμού - Επιστροφής Τελικού Βαθμού
    double findTelikos2(){
        return Theory * 0.6 + Erg * 0.4;
    }
}
```

## Αλγόριθμος κλάσης `Objects3Methods` – **`main()`**

1. Δημιουργία Αντικειμένου Φοιτητή 1
2. Εισαγωγή Στοιχείων 1ου φοιτητή
3. **Υπολογισμός - Εμφάνιση Τελικού Βαθμού 1ου φοιτητή με κλήση μεθόδου `findTelikos1()`**
4. Δημιουργία Αντικειμένου Φοιτητή 2
5. Εισαγωγή Στοιχείων 2ου φοιτητή
6. **Υπολογισμός - Εμφάνιση Τελικού Βαθμού 2ου φοιτητή με κλήση μεθόδου `findTelikos2()`**

### Πρόγραμμα

```
public class Objects3Methods {
/*  Πρόγραμμμα το οποίο δημιουργεί 2 αντικείμενα τύπου Foititis, δίνει τιμές
στον Αριθμό Μητρώου, τον Κωδικό Μαθήματος, το Βαθμό Θεωρίας και το Βαθμό
Εργαστηρίου για τους 2 φοιτητές και Υπολογίζει και Εμφανίζει τον Τελικό Βαθμό
του κάθε φοιτητή με την κλήση των μεθόδων findTelikos1() και findTelikos2()
που έχουν οριστεί στην κλάση Foititis.
*/

    public static void main(String[] args) {

    // Δημιουργία Αντικειμένου Φοιτητή 1
    Foititis f1 = new Foititis();

    // Εισαγωγή Στοιχείων Φοιτητή 1
    f1.AM = 14013;
    f1.KM = 6000232;
    f1.Theory = 6.5;
    f1.Erg = 8.2;

     // Υπολογισμός-Εμφάνιση Τελικού Βαθμού Φοιτητή 1 με κλήση μεθόδου
     // findTelikos1()
    f1.findTelikos1();

    // Δημιουργία Αντικειμένου Φοιτητή 2
    Foititis f2 = new Foititis();

    // Εισαγωγή Στοιχείων Φοιτητή 2
    f2.AM = 14014;
    f2.KM = 6000232;
    f2.Theory = 8.6;
    f2.Erg = 8.0;

     // Υπολογισμός-Εμφάνιση Τελικού Βαθμού Φοιτητή 2 με κλήση μεθόδου
     // findTelikos2()
    System.out.println ( "Τελικός Βαθμός Φοιτητή f2 = "
                          + f2.findTelikos2() );
}
}
```

### Έξοδος Προγράμματος :

```
run:
Τελικός Βαθμός Φοιτητή = 7.18
Τελικός Βαθμός Φοιτητή f2 = 8.36
BUILD SUCCESSFUL (total time: 0 seconds)
```

## 7.4 Υπερφόρτωση Μεθόδων

Η Java έχει το πλεονέκτημα σε σχέση με άλλες γλώσσες προγραμματισμού να μη χρειάζεται διαφορετικά ονόματα μεθόδων στην περίπτωση που είναι διαφορετικός ο τύπος των παραμέτρων. Π.χ. για τον υπολογισμό της απόλυτης τιμής ενός αριθμού, η C διαθέτει στη Μαθηματική της Βιβλιοθήκη τις μεθόδους abs και fabs για τον υπολογισμό της απόλυτης τιμής ενός ακεραίου ή πραγματικού αριθμού αντίστοιχα. Στη Java μπορεί να χρησιμοποιηθεί το **ίδιο όνομα** της μεθόδου ( **υπερφόρτωση** μεθόδων ), αρκεί να διαφέρει η **υπογραφή** της μεθόδου, δηλαδή ο τύπος επιστροφής ή και ο τύπος των παραμέτρων. Θα μπορούσαν στην ίδια κλάση να υπάρχουν όλες οι εκδόσεις της μεθόδου add() που χρησιμοποιήθηκαν στο Κεφάλαιο 5 – **δεν μπορούν να συνυπάρχουν οι μέθοδοι void add(int a, int b)** και **int add(int a, int b) γιατί ενώ έχουν παραμέτρους του ιδίου τύπου διαφέρουν ΜΟΝΟ στον τύπο επιστροφής** - και η αντίστοιχη μέθοδος που δέχεται και επιστρέφει τιμές τύπου **double** :

```
static double add(double a, double b) {
    // Υπολογισμός - Επιστροφή Τιμής Αθροίσματος (a + b)
    return (a + b);
}
```

οπότε, ανάλογα με τους **τύπους** των ορισμάτων που περνάμε στη main() θα εκτελείται και ο κώδικας της **αντίστοιχης** μεθόδου.

### 7.4.1 Εφαρμογή Υπερφόρτωσης Μεθόδων

Να γραφεί πρόγραμμα που να καλεί τις μεθόδους void add(),int add(int a, int b) και double add(double a, double b) και να εμφανίζει το άθροισμα και το Μέσο Όρο.

**ΠΡΟΓΡΑΜΜΑ**

```
public class YperfortoshAdd {
/*
Πρόγραμμα που  καλεί τις μεθόδους void add(), int add(int a, int b) και
double add(double a, double b) και εμφανίζει το άθροισμα και το Μέσο Όρο.
*/
    static void add() {

    // Δήλωση-Ανάθεση των τιμών 5, 6 στις ακέραιες μεταβλητές a, b
        int a = 5, b = 6;
        // Δήλωση - Υπολογισμός του αθροίσματος sum
        int sum = a + b;

        // Εμφάνιση  του αθροίσματος sum
        System.out.println("Κλήση void add() - Άθροισμα = " + sum);
        }
```

```
    static int add(int a, int b) {

        // Δήλωση - Υπολογισμός του αθροίσματος sum
        int sum = a + b;

        // Επιστροφή Τιμής Αθροίσματος sum
        return sum;
        }

    static double add(double a, double b) {

        // Υπολογισμός - Επιστροφή Τιμής Αθροίσματος (a + b)
        return (a + b);
    }

    public static void main(String[] args) {

    // Κλήση Μεθόδου add()
    add();


    // Δήλωση-Ανάθεση των τιμών 5, 6 στις ακέραιες μεταβλητές a, b
    int a = 5, b = 6;

    // Κλήση Μεθόδου int add(a, b)
    int isum = add(a, b);

    // Δήλωση - Υπολογισμός του Μέσου Όρου mo
    double mo = (double) isum/2;

    // Εμφάνιση  του αθροίσματος isum και του Μέσου Όρου mo
    System.out.println("Κλήση int add() - Άθροισμα = " + isum
                       + " Μέσος Όρος = " + mo );

    // Κλήση Μεθόδου double add(a, b)
    double dsum = add(a, b);

    // Δήλωση - Υπολογισμός του Μέσου Όρου mo
    mo = dsum/2;

    // Εμφάνιση  του αθροίσματος dsum και του Μέσου Όρου mo
    System.out.println("Κλήση double add() - Άθροισμα = " + dsum
                       + " Μέσος Όρος = " + mo );

    }
}
```

## Έξοδος Προγράμματος

```
run:
Κλήση void add() - Άθροισμα = 11
Κλήση int add() - Άθροισμα = 11 Μέσος Όρος = 5.5
Κλήση double add() - Άθροισμα = 11 Μέσος Όρος = 5.5
BUILD SUCCESSFUL (total time: 0 seconds)
```

## 7.5 Μέθοδοι Κατασκευής - Δομητές

Όταν δημιουργείται ένα αντικείμενο με τον τελεστή `new`, ενεργοποιείται μια εξ ορισμού ( `default` ) μέθοδος κατασκευής-δομητής του αντικειμένου, η οποία γεμίζει τα πεδία με το μηδενικό στοιχείο του κάθε τύπου δεδομένων που έχει οριστεί σαν πεδίο της κλάσης. Αυτή την έννοια έχει π.χ. το `Foititis()` στην εντολή `Foititis f = new Foititis();`

Μπορούμε να παρέμβουμε στον εξ ορισμού τρόπο δημιουργίας του αντικειμένου και να χρησιμοποιήσουμε μεθόδους κατασκευής-δομητές, οι οποίες δίνουν τιμές ( σταθερές ή παραμετρικά ) σε όσα πεδία ενός αντικειμένου της κλάσης επιθυμούμε.

Οι μέθοδοι κατασκευής-δομητές έχουν όλες **το όνομα της κλάσης** ( υπερφόρτωση ) και **δεν έχουν τύπο** ( ούτε `void` ).

Οι τιμές που περνάνε παραμετρικά στις μεθόδους κατασκευής-δομητές μπορεί να είναι σταθερές ή μεταβλητές.

**Παράδειγμα**

Μέθοδος κατασκευής-δομητής που γεμίζει παραμετρικά όλα τα πεδία ενός αντικειμένου τύπου `Foititis` με τη δημιουργία του.

```
Foititis(int am, int km, double th, double erg){
        AM = am;
        KM = km;
        Theory = th;
        Erg = erg;
 }
```

Η εντολή για τη δημιουργία του αντικειμένου `f` μπορεί να έχει τώρα την παρακάτω μορφή :

```
Foititis f = new Foititis(14014, 6000232, 8.6, 8.0);

Foititis f = new Foititis(am, km, theory, erg);
```

όπου οι μεταβλητές παράμετροι **`am`**, **`km`**, **`theory`**, **`erg`** έχουν πάρει κάποιες τιμές στην καλούσα μέθοδο.

❖ Όταν χρησιμοποιούμε τα ίδια ονόματα για τις παραμέτρους και τα πεδία της κλάσης χρησιμοποιείται το `this`.

**Παράδειγμα**

```
Foititis(int AM, int KM, double Theory, double Erg){
        this.AM = AM;
        this.KM = KM;
        this.Theory = Theory;
        this.Erg = Erg;
 }
```

### 7.5.1 Παράδειγμα Δημιουργίας Κλάσης η οποία περιέχει Δεδομένα και Μεθόδους σαν μέλη και Μεθόδους Κατασκευής - Δομητές

- Να γραφεί Αλγόριθμος/Πρόγραμμα το οποίο δημιουργεί 3 αντικείμενα τύπου `Foititis`. **Ο πρώτος φοιτητής δημιουργείται με το δεύτερο δομητή περνώντας τις τιμές, τις οποίες δίνουμε σε μεταβλητές στη `main()`, ο δεύτερος φοιτητής δημιουργείται με το δεύτερο δομητή περνώντας απ' ευθείας τις τιμές και ο τρίτος φοιτητής δημιουργείται με τον πρώτο δομητή ενώ οι υπόλοιπες τιμές δίνονται απ' ευθείας στα αντίστοιχα πεδία του αντικειμένου**. Και για τους 3 φοιτητές το πρόγραμμα εμφανίζει τον Αριθμό Μητρώου, τον Κωδικό Μαθήματος, το Βαθμό Θεωρίας και το Βαθμό Εργαστηρίου και Υπολογίζει και Εμφανίζει τον Τελικό Βαθμό του κάθε φοιτητή με την κλήση των μεθόδων `findTelikos1()` και `findTelikos2()` που έχουν οριστεί στην κλάση `Foititis`.

**Κλάση** `Foititis`

Η Κλάση **`Foititis`** περιέχει :

- Τα πεδία ( μέλη ) Αριθμός Μητρώου Φοιτητή, Κωδικός Μαθήματος, Βαθμός Θεωρίας και Βαθμός Εργαστηρίου.
- **Τη Μέθοδο Κατασκευής/Δομητή `Foititis()`, η οποία δίνει μια συγκεκριμμένη τιμή στον Κωδικό Μαθήματος.**
- **Τη Μέθοδο Κατασκευής `Foititis(int am, int km, double th, double erg)`, η οποία δίνει παραμετρικά τις τιμές σε όλα τα πεδία του αντικειμένου.**
- Τη Μέθοδο Υπολογισμού - Εμφάνισης Τελικού Βαθμού `findTelikos1()`.
- Τη Μέθοδο Υπολογισμού - Επιστροφής Τελικού Βαθμού `findTelikos2()`.

```
public class Foititis {
    int AM;          // Αριθμός Μητρώου Φοιτητή
    int KM;          // Κωδικός Μαθήματος
    double Theory;   // Βαθμός Θεωρίας
    double Erg;      // Βαθμός Εργαστηρίου

    // Μέθοδος Κατασκευής - Δομητής 1
    Foititis(){
        KM = 6000232;
    }

    // Μέθοδος Κατασκευής - Δομητής 2
    Foititis(int am, int km, double th, double erg){
        AM = am;
        KM = km;
        Theory = th;
        Erg = erg;
    }

    // Μέθοδος Υπολογισμού - Εμφάνισης Τελικού Βαθμού
    void findTelikos1(){
        System.out.println( "Τελικός Βαθμός Φοιτητή = "
                + (Theory * 0.6 + Erg * 0.4));
    }

    // Μέθοδος Υπολογισμού - Επιστροφής Τελικού Βαθμού
```

```
    double findTelikos2(){
        return Theory * 0.6 + Erg * 0.4;
    }
}
```

## Αλγόριθμος κλάσης ObjectsDomhtes – main()

1. Εισαγωγή Στοιχείων 1ου Φοιτητή σε Μεταβλητές
2. **Δημιουργία Αντικειμένου 1ου Φοιτητή με το Δομητή 2**
3. Εμφάνιση στοιχείων 1ου Φοιτητή
4. Υπολογισμός - Εμφάνιση Τελικού Βαθμού 1ου φοιτητή με την κλήση της μεθόδου findTelikos1()
5. **Δημιουργία Αντικειμένου 2ου Φοιτητή με ταυτόχρονη Εισαγωγή Στοιχείων με το Δομητή 2**
6. Εμφάνιση στοιχείων 2ου Φοιτητή
7. Υπολογισμός - Εμφάνιση Τελικού Βαθμού 2ου Φοιτητή με την κλήση της μεθόδου findTelikos2()
8. **Δημιουργία Αντικειμένου 3ου Φοιτητή με τον πρώτο δομητή**
9. Εισαγωγή Υπολοίπων Στοιχείων 3ου Φοιτητή με Πρόσβαση στα δεδομένα του αντικειμένου
10. Εμφάνιση στοιχείων 3ου Φοιτητή
11. Υπολογισμός - Εμφάνιση Τελικού Βαθμού 3ου φοιτητή με την κλήση της μεθόδου findTelikos2()

### Πρόγραμμα

```
public class ObjectsDomhtes {
/*  Πρόγραμμα το οποίο δημιουργεί 3 αντικείμενα τύπου foititis. Ο πρώτος
φοιτητής δημιουργείται με το δεύτερο δομητή περνώντας τις τιμές, τις οποίες
δίνουμε σε μεταβλητές στη main(), ο δεύτερος φοιτητής δημιουργείται με το
δεύτερο δομητή περνώντας απ' ευθείας τις τιμές και ο τρίτος φοιτητής
δημιουργείται με τον πρώτο δομητή και οι υπόλοιπες τιμές δίνονται απ' ευθείας
στα αντίστοιχα πεδία του αντικειμένου. Και για τους 3 φοιτητές το πρόγραμμα
εμφανίζει τον Αριθμό Μητρώου, τον Κωδικό Μαθήματος, το Βαθμό Θεωρίας και το
Βαθμό Εργαστηρίου και Υπολογίζει και Εμφανίζει τον Τελικό Βαθμό του κάθε
φοιτητή.
*/
    public static void main(String[] args) {
    double Telikos;

    // Εισαγωγή Στοιχείων Φοιτητή 1 -> Μεταβλητές -> Δομητής 2
    int am = 14013;
    int km = 6000232;
    double theory = 6.5;
    double erg = 8.2;

    // Δημιουργία Αντικειμένου Φοιτητή 1 με το δεύτερο δομητή
    Foititis f1 = new Foititis(am, km, theory, erg);

    // Εμφάνιση στοιχείων Φοιτητή 1
    System.out.println ( "\n\nΦοιτητής f1\n\nΑριθμός Μητρώου = " + f1.AM
                + "\nΚωδικός Μαθήματος = " + f1.KM + "\nΒαθμός Θεωρίας = "
                + f1.Theory + "\nΒαθμός Εργαστηρίου = "  + f1.Erg);

    // Υπολογισμός-Εμφάνιση Τελικού Βαθμού Φοιτητή 1
    // Κλήση μεθόδου findTelikos1()
    f1.findTelikos1();
```

```
        // Δημιουργία Αντικειμένου Φοιτητή 2 με το δεύτερο δομητή
        // + Εισαγωγή Στοιχείων -> Δομητής
        Foititis f2 = new Foititis(14014, 6000232, 8.6, 8.0);

        // Εμφάνιση στοιχείων Φοιτητή 2
        System.out.println ( "\n\nΦοιτητής f2\n\nΑριθμός Μητρώου = " + f2.AM
                    + "\nΚωδικός Μαθήματος = " + f2.KM + "\nΒαθμός Θεωρίας = "
                    + f2.Theory + "\nΒαθμός Εργαστηρίου = "  + f2.Erg);

        // Υπολογισμός-Εμφάνιση Τελικού Βαθμού Φοιτητή 2
        // Κλήση μεθόδου findTelikos2()
        System.out.println ( "Τελικός Βαθμός Φοιτητή f2 = "
              + f2.findTelikos2() );

        // Δημιουργία Αντικειμένου Φοιτητή 3 με τον πρώτο δομητή
        Foititis f3 = new Foititis();

        // Εισαγωγή Υπολοίπων Στοιχείων Φοιτητή 3
        // Πρόσβαση στα δεδομένα του αντικειμένου
        f3.AM = 14015;
        f3.Theory = 4.6;
        f3.Erg = 4.0;

        // Εμφάνιση στοιχείων Φοιτητή 3
        System.out.println ( "\n\nΦοιτητής f3\n\nΑριθμός Μητρώου = " + f3.AM
                    + "\nΚωδικός Μαθήματος = " + f3.KM + "\nΒαθμός Θεωρίας = "
                    + f3.Theory + "\nΒαθμός Εργαστηρίου = "  + f3.Erg);

       // Υπολογισμός-Εμφάνιση Τελικού Βαθμού Φοιτητή 3
       // Κλήση μεθόδου findTelikos2
        System.out.println ( "Τελικός Βαθμός Φοιτητή f3 = "
              + f3.findTelikos2() );
      }
    }
```

**Έξοδος Προγράμματος :**

```
run:
Φοιτητής f1

Αριθμός Μητρώου = 14013
Κωδικός Μαθήματος = 6000232
Βαθμός Θεωρίας = 6.5
Βαθμός Εργαστηρίου = 8.2
Τελικός Βαθμός Φοιτητή = 7.18


Φοιτητής f2

Αριθμός Μητρώου = 14014
Κωδικός Μαθήματος = 6000232
Βαθμός Θεωρίας = 8.6
Βαθμός Εργαστηρίου = 8.0
Τελικός Βαθμός Φοιτητή f2 = 8.36


Φοιτητής f3

Αριθμός Μητρώου = 14015
Κωδικός Μαθήματος = 6000232
Βαθμός Θεωρίας = 4.6
Βαθμός Εργαστηρίου = 4.0
Τελικός Βαθμός Φοιτητή f3 = 4.359999999999999
BUILD SUCCESSFUL (total time: 0 seconds)
```

## 7.6 Εμφάνιση των δεδομένων με τη Μέθοδο toString()

Με τη δημιουργία ενός αντικειμένου κληρονομείται και η μέθοδος **toString(),** η οποία μας δίνει τη δυνατότητα να εμφανίζουμε – δίνοντας απλώς το όνομα ( αναφορά ) του αντικειμένου - όσα από τα δεδομένα του επιθυμούμε και με τον τρόπο που εμείς επιλέγουμε να εμφανισθούν. Η μέθοδος `toString()` είναι τύπου `String` (συμβολοσειρά) και επιστρέφει μια συμβολοσειρά που περιέχει μηνύματα και ονόματα μεταβλητών της κλάσης, τα οποία θα θέλαμε να εμφανιστούν με την εντολή `System.out.println()`.

**Παράδειγμα**

```
public String toString() {
      String s = "\nΑριθμός Μητρώου = " + AM + "\nΚωδικός
Μαθήματος = ";
      s +=  KM + "\nΒαθμός Θεωρίας = " + Theory + "\nΒαθμός
Εργαστηρίου = "
            + Erg;
      return s;
    }
```

Το περιεχόμενο της μεταβλητής `s` που επιστρέφει η μέθοδος θα εμφανιστεί με την εντολή :

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

Το ίδιο αποτέλεσμα θα είχαμε, αν γράφαμε την παρακάτω **toString()** :

```
public String toString() {
  return "\nΑριθμός Μητρώου = " + AM + "\nΚωδικός Μαθήματος = " +
  KM + "\nΒαθμός Θεωρίας = " + Theory + "\nΒαθμός Εργαστηρίου = "
  + Erg;
}
```

### 7.6.1 Παράδειγμα Δημιουργίας Κλάσης η οποία περιέχει Δεδομένα και Μεθόδους σαν μέλη, Μεθόδους Κατασκευής – Δομητές και τη Μέθοδο toString()

- Να γραφεί Αλγόριθμος/Πρόγραμμα το οποίο δημιουργεί `3` αντικείμενα τύπου `Foititis`. Ο πρώτος φοιτητής δημιουργείται με το δεύτερο δομητή περνώντας τις τιμές, τις οποίες δίνουμε σε μεταβλητές στη `main()`, ο δεύτερος φοιτητής δημιουργείται με το δεύτερο δομητή περνώντας απ' ευθείας τις τιμές και ο τρίτος φοιτητής δημιουργείται με τον πρώτο δομητή και οι υπόλοιπες τιμές δίνονται απ' ευθείας στα αντίστοιχα πεδία του αντικειμένου. Και για τους `3` φοιτητές το πρόγραμμα εμφανίζει τον Αριθμό Μητρώου, τον Κωδικό Μαθήματος, το Βαθμό Θεωρίας και το Βαθμό Εργαστηρίου **με την κλήση της μεθόδου toString()** και Υπολογίζει και Εμφανίζει τον Τελικό Βαθμό του κάθε φοιτητή με την κλήση των μεθόδων `findTelikos1()` και `findTelikos2()` που έχουν οριστεί στην κλάση `Foititis`.

## Κλάση `Foititis`

Η Κλάση **`Foititis`** περιέχει :

- Τα πεδία ( μέλη ) Αριθμός Μητρώου Φοιτητή, Κωδικός Μαθήματος, Βαθμός Θεωρίας και Βαθμός Εργαστηρίου.
- Τη Μέθοδο Κατασκευής/Δομητή `Foititis()`, η οποία δίνει μια συγκεκριμμένη τιμή στον Κωδικό Μαθήματος.
- Τη Μέθοδο Κατασκευής `Foititis(int am, int km, double th, double erg)`, η οποία δίνει παραμετρικά τις τιμές σε όλα τα πεδία του αντικειμένου.
- Τη Μέθοδο Υπολογισμού - Εμφάνισης Τελικού Βαθμού `findTelikos1()`.
- Τη Μέθοδο Υπολογισμού - Επιστροφής Τελικού Βαθμού `findTelikos2()`.
- **Τη Μέθοδο Εμφάνισης των Στοιχείων του κάθε Φοιτητή `toString()`.**

```
public class Foititis {
    int AM;          // Αριθμός Μητρώου Φοιτητή
    int KM;          // Κωδικός Μαθήματος
    double Theory;   // Βαθμός Θεωρίας
    double Erg;      // Βαθμός Εργαστηρίου

    // Μέθοδος Κατασκευής - Δομητής 1
    Foititis(){
        KM = 6000232;
    }

    // Μέθοδος Κατασκευής - Δομητής 2
    Foititis(int am, int km, double th, double erg){
        AM = am;
        KM = km;
        Theory = th;
        Erg = erg;
    }

    // Μέθοδος Υπολογισμού - Εμφάνισης Τελικού Βαθμού
    void findTelikos1(){
        System.out.println( "Τελικός Βαθμός Φοιτητή = "
                + (Theory * 0.6 + Erg * 0.4));
    }

    // Μέθοδος Υπολογισμού - Επιστροφής Τελικού Βαθμού
    double findTelikos2(){
        return Theory * 0.6 + Erg * 0.4;
    }


    // Μέθοδος Εμφάνισης Στοιχείων Φοιτητή
    public String toString() {
      String s = "\nΑριθμός Μητρώου = " + AM + "\nΚωδικός Μαθήματος = ";
      s +=  KM + "\nΒαθμός Θεωρίας = " + Theory + "\nΒαθμός Εργαστηρίου = "
            + Erg;
      return s;
    }
}
```

## Αλγόριθμος κλάσης Objects5toString – main()

1. Εισαγωγή Στοιχείων 1ου Φοιτητή σε Μεταβλητές
2. Δημιουργία Αντικειμένου 1ου Φοιτητή με το Δομητή 2
3. **Εμφάνιση στοιχείων 1ου Φοιτητή με τη μέθοδο `toString()`**
4. Υπολογισμός - Εμφάνιση Τελικού Βαθμού 1ου φοιτητή με την κλήση της μεθόδου `findTelikos1()`
5. Δημιουργία Αντικειμένου 2ου Φοιτητή με ταυτόχρονη Εισαγωγή Στοιχείων με το Δομητή 2
6. **Εμφάνιση στοιχείων 2ου Φοιτητή με τη μέθοδο `toString()`**
7. Υπολογισμός - Εμφάνιση Τελικού Βαθμού 2ου φοιτητή με κλήση μεθόδου `findTelikos2()`
8. Δημιουργία Αντικειμένου 3ου Φοιτητή με τον πρώτο δομητή
9. Εισαγωγή Υπολοίπων Στοιχείων 3ου Φοιτητή με Πρόσβαση στα δεδομένα του αντικειμένου
10. **Εμφάνιση στοιχείων 3ου Φοιτητή με τη μέθοδο `toString()`**
11. Υπολογισμός - Εμφάνιση Τελικού Βαθμού 3ου φοιτητή με την κλήση της μεθόδου `findTelikos2()`

### Πρόγραμμα

```
public class Objects5toString {
/*  Πρόγραμμμα το οποίο δημιουργεί 3 αντικείμενα τύπου foititis. Ο πρώτος
φοιτητής δημιουργείται με το δεύτερο δομητή περνώντας τις τιμές,τις οποίες
δίνουμε σε μεταβλητές στη main(), ο δεύτερος φοιτητής δημιουργείται με το
δεύτερο δομητή περνώντας απ' ευθείας τις τιμές και ο τρίτος φοιτητής
δημιουργείται με τον πρώτο δομητή και οι υπόλοιπες τιμές δίνονται απ' ευθείας
στα αντίστοιχα πεδία του αντικειμένου. Και για τους 3 φοιτητές το πρόγραμμα
εμφανίζει τον Αριθμό Μητρώου, τον Κωδικό Μαθήματος, το Βαθμό Θεωρίας και το
Βαθμό Εργαστηρίου με την κλήση της μεθόδου toString() και Υπολογίζει και
Εμφανίζει τον Τελικό Βαθμό του κάθε φοιτητή.
*/
    public static void main(String[] args) {

    // Εισαγωγή Στοιχείων Φοιτητή 1 -> Μεταβλητές -> Δομητής
    int am = 14013;
    int km = 6000232;
    double theory = 6.5;
    double erg = 8.2;

    // Δημιουργία Αντικειμένου Φοιτητή 1 με το δεύτερο δομητή
    Foititis f1 = new Foititis (am, km, theory, erg);

    // Εμφάνιση Στοιχείων Φοιτητή 1 με κλήση μεθόδου toString
    System.out.println ( "\nΣτοιχεία Φοιτητή 1");
    System.out.println ( f1 );

    // Υπολογισμός-Εμφάνιση Τελικού Βαθμού Φοιτητή 1
    // Κλήση μεθόδου findTelikos1()
    f1.findTelikos1();

    // Δημιουργία Αντικειμένου Φοιτητή 2 με το δεύτερο δομητή
    // + Εισαγωγή Στοιχείων -> Δομητής
    Foititis f2 = new Foititis(14014, 6000232, 8.6, 8.0);



    // Εμφάνιση Στοιχείων Φοιτητή 2 με κλήση μεθόδου toString
```

```
        System.out.println ( "\nΣτοιχεία Φοιτητή 2");
        System.out.println ( f2 );

        // Υπολογισμός-Εμφάνιση Τελικού Βαθμού Φοιτητή 2
        // Κλήση μεθόδου findTelikos2()
        System.out.println ( "Τελικός Βαθμός Φοιτητή f2 = "
              + f2.findTelikos2() );

        // Δημιουργία Αντικειμένου Φοιτητή 3 με τον πρώτο δομητή
        Foititis f3 = new Foititis();

        // Εισαγωγή Υπολοίπων Στοιχείων Φοιτητή 3
        // Πρόσβαση στα δεδομένα του αντικειμένου
        f3.AM = 14015;
        f3.Theory = 4.6;
        f3.Erg = 4.0;

        // Εμφάνιση Στοιχείων Φοιτητή 3 με κλήση μεθόδου toString
        System.out.println ( "\nΣτοιχεία Φοιτητή 3");
        System.out.println ( f3 );

       // Υπολογισμός-Εμφάνιση Τελικού Βαθμού Φοιτητή 3
       // Κλήση μεθόδου findTelikos2
        System.out.println ( "Τελικός Βαθμός Φοιτητή f3 = "
              + f3.findTelikos2() );
      }
    }
```

## Έξοδος Προγράμματος :

```
run:
Στοιχεία Φοιτητή 1

Αριθμός Μητρώου = 14013
Κωδικός Μαθήματος = 6000232
Βαθμός Θεωρίας = 6.5
Βαθμός Εργαστηρίου = 8.2
Τελικός Βαθμός Φοιτητή = 7.18

Στοιχεία Φοιτητή 2

Αριθμός Μητρώου = 14014
Κωδικός Μαθήματος = 6000232
Βαθμός Θεωρίας = 8.6
Βαθμός Εργαστηρίου = 8.0
Τελικός Βαθμός Φοιτητή f2 = 8.36

Στοιχεία Φοιτητή 3

Αριθμός Μητρώου = 14015
Κωδικός Μαθήματος = 6000232
Βαθμός Θεωρίας = 4.6
Βαθμός Εργαστηρίου = 4.0
Τελικός Βαθμός Φοιτητή f3 = 4.359999999999999
BUILD SUCCESSFUL (total time: 0 seconds)
```

## 7.7 Προσδιοριστικά Πρόσβασης στα Δεδομένα ενός Αντικειμένου

Ένα από τα χαρακτηριστικά της Java είναι η δυνατότητα να προστατεύσει τα δεδομένα μιας κλάσης από την αυθαίρετη χρήση κάποιου, ο οποίος θα χρησιμοποιήσει σε μια εφαρμογή τη συγκεκριμμένη κλάση δημιουργώντας τα αντίστοιχα αντικείμενα.

Όταν ένα πεδίο ή μια μέθοδος σε μια κλάση δεν έχει πριν τη δήλωση του τύπου του κανένα προσδιοριστικό πρόσβασης, θεωρείται **δημόσιο** ( **public** ) και έχει ελεύθερη πρόσβαση από οποιαδήποτε άλλη κλάση, χρησιμοποιώντας το **όνομα του αντικειμένου** και το **όνομα του πεδίου ή της μεθόδου**. Μπορεί κάποιος να δώσει οποιαδήποτε τιμή σε μια μεταβλητή ( ακόμη και έναν αρνητικό Βαθμό Θεωρίας για παράδειγμα ). Αν δεν ελέγξει στο πρόγραμμά του ότι η τιμή που δίνει σε μια μεταβλητή είναι αποδεκτή, θα προκύψουν απρόσμενα αποτελέσματα.

Με το προσδιοριστικό **private** πριν τη δήλωση του τύπου του πεδίου ή της μεθόδου, το πεδίο ή η μέθοδος γίνεται ιδιωτικό και προστατεύεται από την αυθαίρετη χρήση του. Η πρόσβαση σ' αυτή την περίπτωση **ΔΕΝ** μπορεί να γίνει με τη χρήση του ονόματός του, όπως στο public ή το default και απαιτούνται ειδικές **μέθοδοι πρόσβασης**, οι οποίες **ΠΡΕΠΕΙ** να είναι **μέλη** της κλάσης.

Σ' αυτές της μεθόδους μπορούν να μπουν και οι έλεγχοι που απαιτούνται, ώστε οι τιμές που θα δίνονται να είναι **έγκυρες**. Ο σχεδιαστής της κλάσης μ' αυτό τον τρόπο διασφαλίζει τη σωστή χρήση του πεδίου ή της μεθόδου.

Οι μέθοδοι πρόσβασης συνήθως έχουν το όνομα **get<όνομα_μέλους_κλάσης>()** για τη χρήση της τιμής ενός μέλους και **set<όνομα_μέλους_κλάσης>()** για την ανάθεση της τιμής σε ένα μέλος της κλάσης.

**Παράδειγμα**

Η μέθοδος

```
double getTheory(){
   return Theory;
}
```

επιστρέφει την τιμή του πεδίου Theory ( Βαθμός Θεωρίας ), ενώ η μέθοδος

```
void setTheory(double th){
 Theory = th;
}
```

αναθέτει την τιμή που έχει η μεταβλητή **th** στο πεδίο **Theory**.

**Σημείωση** : Στη μέθοδο **setTheory(th)** θα μπορούσε να γίνει πριν την ανάθεση της τιμής της μεταβλητής **th** στο πεδίο **Theory** και έλεγχος εγκυρότητας της τιμής του Βαθμού Θεωρίας.

### 7.7.1 Παράδειγμα Δημιουργίας Κλάσης η οποία περιέχει private Δεδομένα και τις αντίστοιχες Μεθόδους get και set, Μεθόδους σαν μέλη, Μεθόδους Κατασκευής – Δομητές και τη Μέθοδο toString()

- Να γραφεί Αλγόριθμος/Πρόγραμμα το οποίο δημιουργεί 3 αντικείμενα τύπου `Foititis`. Ο πρώτος φοιτητής δημιουργείται με το δεύτερο δομητή περνώντας τις τιμές, τις οποίες δίνουμε σε μεταβλητές στη `main()`, ο δεύτερος φοιτητής δημιουργείται με το δεύτερο δομητή περνώντας απ' ευθείας τις τιμές και ο τρίτος φοιτητής δημιουργείται με τον πρώτο δομητή και οι υπόλοιπες τιμές δίνονται **στα αντίστοιχα private πεδία** του αντικειμένου **με τη χρήση των μεθόδων `setAM()`, `setTheory()`, `setErg()`.** Και για τους 3 φοιτητές το πρόγραμμα εμφανίζει τον Αριθμό Μητρώου, τον Κωδικό Μαθήματος, το Βαθμό Θεωρίας και το Βαθμό Εργαστηρίου με την κλήση της μεθόδου `toString()` και Υπολογίζει και Εμφανίζει τον Τελικό Βαθμό του κάθε φοιτητή. **Για το φοιτητή 3 θα εμφανίζει τον Τελικό του Βαθμό με πρόσβαση στα πεδία `Theory` και `Erg` με τις μεθόδους `getTheory()` και `getErg()`.**

## Κλάση `Foititis`

Η Κλάση **`Foititis`** περιέχει :

- Τα πεδία Αριθμός Μητρώου Φοιτητή, Κωδικός Μαθήματος, Βαθμός Θεωρίας και Βαθμός Εργαστηρίου**. Όλα τα πεδία δηλώνονται `private`, εκτός του πεδίου Κωδικός Μαθήματος.**
- Τη Μέθοδο Κατασκευής/Δομητή `Foititis()`, η οποία δίνει μια συγκεκριμμένη τιμή στον Κωδικό Μαθήματος.
- Τη Μέθοδο Κατασκευής `Foititis(int am, int km, double th, double erg)`, η οποία δίνει παραμετρικά τις τιμές σε όλα τα πεδία του αντικειμένου.
- **Τις μεθόδους `getAM()`, `getTheory()`, `getErg()`** για τη Χρήση των τιμών των `private` πεδίων.
- **Τις μεθόδους `setAM()`, `setTheory()`, `setErg()`** για την Ανάθεση τιμών στα `private` πεδία.
- Τη Μέθοδο Υπολογισμού - Εμφάνισης Τελικού Βαθμού `findTelikos1()`.
- Τη Μέθοδο Υπολογισμού - Επιστροφής Τελικού Βαθμού `findTelikos2()`.
- Τη Μέθοδο Εμφάνισης των Στοιχείων του κάθε Φοιτητή `toString()`.

```
public class Foititis{
    private int AM;          // Αριθμός Μητρώου Φοιτητή
    int KM;                  // Κωδικός Μαθήματος
    private double Theory;   // Βαθμός Θεωρίας
    private double Erg;      // Βαθμός Εργαστηρίου

    // Μέθοδος Κατασκευής - Δομητής 1
    Foititis(){
        KM = 6000232;
    }
```

```
    // Μέθοδος Κατασκευής - Δομητής 2
    Foititis (int am, int km, double th, double erg){
        AM = am;
        KM = km;
        Theory = th;
        Erg = erg;
    }

    // Μέθοδοι get - Χρήση τιμών των private πεδίων
    int getAM(){
     return AM;
    }

    double getTheory(){
     return Theory;
    }

    double getErg(){
     return Erg;
    }

    // Μέθοδοι set - Ανάθεση τιμών στα private πεδία
    void setAM(int am ){
     AM = am;
    }

    void setTheory(double th){
     Theory = th;
    }

    void setErg(double erg){
     Erg = erg;
    }

    // Μέθοδος Υπολογισμού - Εμφάνισης Τελικού Βαθμού
    void findTelikos1(){
        System.out.println( "Τελικός Βαθμός Φοιτητή = "
                            + (Theory * 0.6 + Erg * 0.4));
    }

    // Μέθοδος Υπολογισμού - Επιστροφής Τελικού Βαθμού
    double findTelikos2(){
        return Theory * 0.6 + Erg * 0.4;
    }

    // Μέθοδος Εμφάνισης Στοιχείων Φοιτητή
    public String toString() {
      String s = "\nΑριθμός Μητρώου = " + AM + "\nΚωδικός Μαθήματος = " + KM;
      s += "\nΒαθμός Θεωρίας = " + Theory + "\nΒαθμός Εργαστηρίου = " + Erg;
      return s;
    }
}
```

## Αλγόριθμος κλάσης Objects6Private – main()

1. Εισαγωγή Στοιχείων 1ου Φοιτητή σε Μεταβλητές
2. Δημιουργία Αντικειμένου 1ου Φοιτητή με το Δομητή 2
3. Εμφάνιση στοιχείων 1ου Φοιτητή με τη μέθοδο toString()
4. Υπολογισμός - Εμφάνιση Τελικού Βαθμού 1ου φοιτητή με την κλήση της μεθόδου findTelikos1()

5. Δημιουργία Αντικειμένου 2ου Φοιτητή με ταυτόχρονη Εισαγωγή Στοιχείων με το Δομητή 2
6. Εμφάνιση στοιχείων 2ου Φοιτητή με τη μέθοδο `toString()`
7. Υπολογισμός - Εμφάνιση Τελικού Βαθμού 2ου φοιτητή με την κλήση της μεθόδου `findTelikos2()`
8. Δημιουργία Αντικειμένου 3ου Φοιτητή με τον πρώτο δομητή
9. Εισαγωγή Υπολοίπων Στοιχείων 3ου Φοιτητή με Πρόσβαση στα δεδομένα του αντικειμένου με τις **μεθόδους `setAM()`, `setTheory()`, `setErg()`.**
10. Εμφάνιση στοιχείων 3ου Φοιτητή με τη μέθοδο `toString()`
11. Υπολογισμός - Εμφάνιση Τελικού Βαθμού 3ου φοιτητή με την κλήση της μεθόδου `findTelikos2()`
12. Υπολογισμός - Εμφάνιση Τελικού Βαθμού 3ου φοιτητή με με πρόσβαση στα πεδία `Theory` και `Erg` με τις μεθόδους **`getTheory()`** και **`getErg()`**

**Πρόγραμμα**

```
public class Objects6Private {
/*  Πρόγραμμμα το οποίο δημιουργεί 3 αντικείμενα τύπου foititis. Ο πρώτος φοιτητής δημιουργείται με το δεύτερο δομητή περνώντας τις τιμές, τις οποίες δίνουμε σε μεταβλητές στη main(), ο δεύτερος φοιτητής δημιουργείται με το δεύτερο δομητή περνώντας απ' ευθείας τις τιμές και ο τρίτος φοιτητής δημιουργείται με τον πρώτο δομητή και οι υπόλοιπες τιμές δίνονται στα αντίστοιχα πεδία του αντικειμένου με τη χρήση των μεθόδων setAM(), setTheory(), setErg(). Και για τους 3 φοιτητές το πρόγραμμα εμφανίζει τον Αριθμό Μητρώου, τον Κωδικό Μαθήματος, το Βαθμό Θεωρίας και το Βαθμό Εργαστηρίου με την κλήση της μεθόδου toString() και Υπολογίζει και Εμφανίζει τον Τελικό Βαθμό του κάθε φοιτητή. Για το φοιτητή 3 θα εμφανίζει τον Τελικό του Βαθμό με πρόσβαση στα πεδία Theory και Erg με τις μεθόδους getTheory() και getErg().
*/
    public static void main(String[] args) {

    // Εισαγωγή Στοιχείων Φοιτητή 1 -> Μεταβλητές -> Δομητής 2
    int am = 14013;
    int km = 6000232;
    double theory = 6.5;
    double erg = 8.2;

    // Δημιουργία Αντικειμένου Φοιτητή 1 με το δεύτερο δομητή
    Foititis f1 = new Foititis(am, km, theory, erg);

    // Εμφάνιση Στοιχείων Φοιτητή 1 με κλήση μεθόδου toString
    System.out.println ( "\nΣτοιχεία Φοιτητή 1");
    System.out.println ( f1 );

    // Υπολογισμός-Εμφάνιση Τελικού Βαθμού Φοιτητή 1
    // Κλήση μεθόδου findTelikos1()
    f1.findTelikos1();

    // Δημιουργία Αντικειμένου Φοιτητή 2 με το δεύτερο δομητή
    // + Εισαγωγή Στοιχείων -> Δομητής
    Foititis f2 = new Foititis(14014, 6000232, 8.6, 8.0);


    // Εμφάνιση Στοιχείων Φοιτητή 2 με κλήση μεθόδου toString
    System.out.println ( "\nΣτοιχεία Φοιτητή 2");
    System.out.println ( f2 );
```

```
        // Υπολογισμός-Εμφάνιση Τελικού Βαθμού Φοιτητή 2
        // Κλήση μεθόδου findTelikos2
        System.out.println ("Τελικός Βαθμός Φοιτητή f2 = " + f2.findTelikos2());

        // Δημιουργία Αντικειμένου Φοιτητή 3 με τον πρώτο δομητή
        Foititis f3 = new Foititis();

        // Ανάθεση τιμών στα υπόλοιπα private πεδία του φοιτητή 1 MONO με τις
        // μεθόδους set
        f3.setAM( 14015 );
        f3.setTheory( 4.6 );
        f3.setErg( 4.0 );

        // Εμφάνιση Στοιχείων Φοιτητή 3 με κλήση μεθόδου toString
        System.out.println ( "\nΣτοιχεία Φοιτητή 3");
        System.out.println ( f3 );

       // Υπολογισμός-Εμφάνιση Τελικού Βαθμού Φοιτητή 3
       // Κλήση μεθόδου findTelikos2
       System.out.println ( "Τελικός Βαθμός Φοιτητή f3 = " + f3.findTelikos2() );

         // Υπολογισμός - Εμφάνιση Τελικού Βαθμού Φοιτητή χωρίς μέθοδο
        // Πρόσβαση στα δεδομένα του φοιτητή MONO με getTheory(), getErg()
        System.out.print ( "Τελικός Βαθμός Φοιτητή 3 με τις μεθόδους ");
        System.out.println("getTheory(),getErg() = " +
                           (f3.getTheory()*0.6 + f3.getErg()*0.4) );
    }
  }
```

**Έξοδος Προγράμματος :**

```
run:
Στοιχεία Φοιτητή 1

Αριθμός Μητρώου = 14013
Κωδικός Μαθήματος = 6000232
Βαθμός Θεωρίας = 6.5
Βαθμός Εργαστηρίου = 8.2
Τελικός Βαθμός Φοιτητή = 7.18

Στοιχεία Φοιτητή 2

Αριθμός Μητρώου = 14014
Κωδικός Μαθήματος = 6000232
Βαθμός Θεωρίας = 8.5
Βαθμός Εργαστηρίου = 8.0
Τελικός Βαθμός Φοιτητή f2 με τη Μέθοδο Telikos2 = 8.3

Στοιχεία Φοιτητή 3

Αριθμός Μητρώου = 14015
Κωδικός Μαθήματος = 6000232
Βαθμός Θεωρίας = 4.6
Βαθμός Εργαστηρίου = 4.0
Τελικός Βαθμός Φοιτητή f3 = 4.359999999999999
Τελικός Βαθμός Φοιτητή f3 με τις μεθόδους getTheory(),getErg() =
4.359999999999999
BUILD SUCCESSFUL (total time: 0 seconds)
```

## 7.8 Δεδομένα final

Όταν μια κλάση περιέχει στα δεδομένα της ένα πεδίο, το οποίο δεν πρόκειται να αλλάξει τιμή, είναι δηλαδή **σταθερό**, μπορεί αυτό το πεδίο να δηλωθεί σαν **`final`**. Π.χ. το πεδίο `KM`, το οποίο αφορά τον Κωδικό Μαθήματος, ο οποίος είναι ίδιος για όλους τους φοιτητές και έχει συγκεκριμένη τιμή, μπορεί να δηλωθεί σαν `final` με την εντολή :

```
final int KM = 6000232;
```

Η τιμή του πεδίου στην παραπάνω δήλωση δίνεται με δυναμική αρχικοποίηση στην κλάση `Foititis` και **ΔΕΝ** μπορεί να αλλάξει.

Σ' αυτή την περίπτωση η πλήρης μέθοδος κατασκευής-δομητής δέχεται σαν παραμέτρους τιμές για **όλα τα υπόλοιπα πεδία, εκτός αυτού που έχει δηλωθεί σαν `final`, την τιμή του οποίου παίρνουν όλα τα αντικείμενα που θα δημιουργηθούν**.

### 7.8.1 Παράδειγμα Δημιουργίας Κλάσης η οποία περιέχει private και final Δεδομένα και τις αντίστοιχες Μεθόδους get και set, Μεθόδους σαν μέλη, Μεθόδους Κατασκευής – Δομητές, και τη Μέθοδο toString()

- Να γραφεί Αλγόριθμος/Πρόγραμμα το οποίο δημιουργεί `3` αντικείμενα τύπου `Foititis`. Ο πρώτος φοιτητής δημιουργείται με το δεύτερο δομητή περνώντας όλες τις τιμές, **εκτός από τον Κωδικό Μαθήματος, ο οποίος δηλώνεται σαν `final`,** τις οποίες δίνουμε σε μεταβλητές στη `main()`, ο δεύτερος φοιτητής δημιουργείται με το δεύτερο δομητή περνώντας απ' ευθείας τις τιμές **εκτός από τον Κωδικό Μαθήματος** και ο τρίτος φοιτητής δημιουργείται με τον πρώτο **κενό δομητή**, ενώ οι υπόλοιπες τιμές δίνονται στα αντίστοιχα `private` πεδία του αντικειμένου με τη χρήση των μεθόδων `setAM()`, `setTheory()`, `setErg()`. Και για τους `3` φοιτητές το πρόγραμμα εμφανίζει τον Αριθμό Μητρώου, τον Κωδικό Μαθήματος, το Βαθμό Θεωρίας και το Βαθμό Εργαστηρίου με την κλήση της μεθόδου `toString()` και Υπολογίζει και Εμφανίζει τον Τελικό Βαθμό του κάθε φοιτητή.

#### Κλάση `Foititis`

Η Κλάση **`Foititis`** περιέχει :

- Τα πεδία Αριθμός Μητρώου Φοιτητή, Κωδικός Μαθήματος, Βαθμός Θεωρίας και Βαθμός Εργαστηρίου. Όλα τα πεδία δηλώνονται `private`, εκτός του πεδίου Κωδικός Μαθήματος, **ο οποίος δηλώνεται σαν `final`.**
- **Τη Μέθοδο Κατασκευής/Δομητή `Foititis()`, κενός δομητής για τη δημιουργία αντικειμένων με τιμή ΜΟΝΟ στο πεδίο τύπου `final`.**
- **Τη Μέθοδο Κατασκευής `Foititis(int am, double th, double erg)`, η οποία δίνει παραμετρικά τις τιμές σε όλα τα πεδία του αντικειμένου, εκτός από τον Κωδικό Μαθήματος που δίνεται αυτόματα.**

- Τις μεθόδους `getAM()`, `getTheory()`, `getErg()` για τη Χρήση των τιμών των `private` πεδίων.
- Τις μεθόδους `setAM()`, `setTheory()`, `setErg()` για την Ανάθεση τιμών στα `private` πεδία.
- Τη Μέθοδο Υπολογισμού - Εμφάνισης Τελικού Βαθμού `findTelikos1()`.
- Τη Μέθοδο Υπολογισμού - Επιστροφής Τελικού Βαθμού `findTelikos2()`.
- Τη Μέθοδο Εμφάνισης των Στοιχείων του κάθε Φοιτητή `toString()`.

```
public class Foititis {
    private int AM;
    final int KM = 6000232;
    private double Theory, Erg;

    Foititis(){
    }

    Foititis(int am, double th, double erg){
        AM = am;
        Theory = th;
        Erg = erg;
    }

    int getAM(){
     return AM;
    }

    double getTheory(){
     return Theory;
    }

    double getErg(){
     return Erg;
    }

    void setAM(int am ){
     AM = am;
    }

    void setTheory(double th){
     Theory = th;
    }

    void setErg(double erg){
     Erg = erg;
    }

    void findTelikos1(){
     System.out.println( "Τελικός Βαθμός Φοιτητή = " + (Theory * 0.6 + Erg * 0.4));
    }

    double findTelikos2(){
        return Theory * 0.6 + Erg * 0.4;
    }

    public String toString() {
      String s = "\nΑριθμός Μητρώου = " + AM + "\nΚωδικός Μαθήματος = " + KM;
      s += "\nΒαθμός Θεωρίας = " + Theory + "\nΒαθμός Εργαστηρίου = " + Erg;
      return s;
    }
}
```

## Αλγόριθμος κλάσης Objects7Final**- main()**

1. Εισαγωγή Στοιχείων 1ου Φοιτητή ( **εκτός του Κωδικού Μαθήματος** ) σε Μεταβλητές
2. Δημιουργία Αντικειμένου 1ου Φοιτητή με το Δομητή 2
3. Εμφάνιση στοιχείων 1ου Φοιτητή με τη μέθοδο `toString()`
4. Υπολογισμός - Εμφάνιση Τελικού Βαθμού 1ου φοιτητή με την κλήση της μεθόδου `findTelikos1()`
5. Δημιουργία Αντικειμένου 2ου Φοιτητή με ταυτόχρονη Εισαγωγή Στοιχείων με το Δομητή 2
6. Εμφάνιση στοιχείων 2ου Φοιτητή με τη μέθοδο `toString()`
7. Υπολογισμός - Εμφάνιση Τελικού Βαθμού 2ου φοιτητή με την κλήση της μεθόδου `findTelikos2()`
8. **Δημιουργία Αντικειμένου 3ου Φοιτητή με τον πρώτο κενό δομητή**
9. Εισαγωγή Υπολοίπων Στοιχείων 3ου Φοιτητή με Πρόσβαση στα δεδομένα του αντικειμένου με τις μεθόδους `setAM(), setTheory(), setErg()`.
10. Εμφάνιση στοιχείων 3ου Φοιτητή με τη μέθοδο `toString()`
11. Υπολογισμός - Εμφάνιση Τελικού Βαθμού 3ου φοιτητή με την κλήση της μεθόδου `findTelikos2()`

## Πρόγραμμα

```
public class Objects7Final {
/*  Πρόγραμμμα το οποίο δημιουργεί 3 αντικείμενα τύπου Foititis. Ο πρώτος
φοιτητής δημιουργείται με το δεύτερο δομητή περνώντας τις τιμές, τις οποίες
δίνουμε σε μεταβλητές στη main(), ο δεύτερος φοιτητής δημιουργείται με το
δεύτερο δομητή περνώντας απ' ευθείας τις τιμές και ο τρίτος φοιτητής
δημιουργείται με τον πρώτο κενό δομητή και οι υπόλοιπες τιμές δίνονται στα
αντίστοιχα πεδία του αντικειμένου με τη χρήση των μεθόδων setAM(),
setTheory(), setErg(). Και για τους 3 φοιτητές το πρόγραμμα εμφανίζει τον
Αριθμό Μητρώου, τον Κωδικό Μαθήματος, το Βαθμό Θεωρίας και το Βαθμό
Εργαστηρίου με την κλήση της μεθόδου toString() και Υπολογίζει και Εμφανίζει
τον Τελικό Βαθμό του κάθε φοιτητή.
*/
    public static void main(String[] args) {

    // Εισαγωγή Στοιχείων Φοιτητή 1 -> Μεταβλητές -> Δομητής
    int am = 14013;
    double theory = 6.5;
    double erg = 8.2;

    // Δημιουργία Αντικειμένου Φοιτητή 1 με το δεύτερο δομητή
    Foititis f1 = new Foititis(am, theory, erg);

    // Εμφάνιση Στοιχείων Φοιτητή 1 με κλήση μεθόδου toString
    System.out.println ( "\nΣτοιχεία Φοιτητή 1");
    System.out.println ( f1 );

    // Υπολογισμός-Εμφάνιση Τελικού Βαθμού Φοιτητή 1
    // Κλήση μεθόδου findTelikos1()
    f1.findTelikos1();

    // Δημιουργία Αντικειμένου Φοιτητή 2 με το δεύτερο δομητή
    // + Εισαγωγή Στοιχείων -> Δομητής
    Foititis f2 = new Foititis(14014, 8.6, 8.0);
```

```
        // Εμφάνιση Στοιχείων Φοιτητή 2 με κλήση μεθόδου toString
        System.out.println ( "\nΣτοιχεία Φοιτητή 2");
        System.out.println ( f2 );

        // Υπολογισμός-Εμφάνιση Τελικού Βαθμού Φοιτητή 2
        // Κλήση μεθόδου findTelikos2()
       System.out.println ( "Τελικός Βαθμός Φοιτητή f2 = " + f2.findTelikos2() );

        // Δημιουργία Αντικειμένου Φοιτητή 3 με τον πρώτο κενό δομητή
        Foititis f3 = new Foititis();

        // Ανάθεση τιμών στα υπόλοιπα private πεδία του φοιτητή 1 MONO με τις
        // μεθόδους set
        f3.setAM( 14015 );
        f3.setTheory( 4.6 );
        f3.setErg( 4.0 );

        // Εμφάνιση Στοιχείων Φοιτητή 3 με κλήση μεθόδου toString
        System.out.println ( "\nΣτοιχεία Φοιτητή 3");
        System.out.println ( f3 );

       // Υπολογισμός-Εμφάνιση Τελικού Βαθμού Φοιτητή 3
       // Κλήση μεθόδου findTelikos2()
       System.out.println ( "Τελικός Βαθμός Φοιτητή f3 = " + f3.findTelikos2() );
     }
  }
```

## Έξοδος Προγράμματος :

```
run:
Στοιχεία Φοιτητή 1

Αριθμός Μητρώου = 14013
Κωδικός Μαθήματος = 6000232
Βαθμός Θεωρίας = 6.5
Βαθμός Εργαστηρίου = 8.2
Τελικός Βαθμός Φοιτητή = 7.18

Στοιχεία Φοιτητή 2

Αριθμός Μητρώου = 14014
Κωδικός Μαθήματος = 6000232
Βαθμός Θεωρίας = 8.5
Βαθμός Εργαστηρίου = 8.0
Τελικός Βαθμός Φοιτητή f2 με τη Μέθοδο Telikos2 = 8.3

Στοιχεία Φοιτητή 3

Αριθμός Μητρώου = 14015
Κωδικός Μαθήματος = 6000232
Βαθμός Θεωρίας = 4.6
Βαθμός Εργαστηρίου = 4.0
Τελικός Βαθμός Φοιτητή f3 = 4.359999999999999

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

## 7.9 Δεδομένα – Μέθοδοι static

Όταν ένα πεδίο ή μια μέθοδος σε μια κλάση αντικειμένων ( όχι στην κλάση που περιέχει τη `main()` ) έχει δηλωθεί σαν **`static`**, η πρόσβαση σ' αυτό το μέλος μπορεί να γίνει είτε με τον κλασσικό τρόπο **<όνομα_αντικειμένου).<μέλος>** είτε με το **<όνομα_κλάσης>.<μέλος>**, στην οποία ανήκει η μεταβλητή ή η μέθοδος `static`, **χωρίς να είναι απαραίτητη η δημιουργία αντικειμένου**.

Συνήθως οι εφαρμογές περιέχουν μια **ξεχωριστή κλάση** με μεθόδους ή και μεταβλητές `static`, οι οποίες απαιτούνται για την επεξεργασία των δεδομένων των αντικειμένων που θα δημιουργηθούν και οι οποίες δεν απαιτούν τη δημιουργία αντικειμένου, γιατί δεν αφορούν κάποια οντότητα, αλλά συνήθως την επεξεργασία δεδομένων πολλών αντικειμένων.

Οι τιμές στα πεδία `static` που δίνονται με δυναμική αρχικοποίηση είτε με ανάθεση τιμής ανατίθενται στο αντίστοιχο πεδίο **ΟΛΩΝ** των αντικειμένων της κλάσης, ακόμη κι αν αλλάξει η τιμή τους χρησιμοποιώντας το όνομα ενός από τα αντικείμενα ( **καθολικές** μεταβλητές ).

Οι μέθοδοι `static` μπορούν να προσπελάσουν **ΜΟΝΟ** `static` μεταβλητές.

Μπορεί σε μια κλάση να υπάρχει ένα `static block`, μια ομάδα εντολών, οι οποίες εκτελούνται πριν τη δημιουργία κάθε αντικειμένου.

### 7.9.1 Παράδειγμα Δημιουργίας Κλάσης η οποία περιέχει private και static Δεδομένα και τις αντίστοιχες Μεθόδους get και set, Μεθόδους σαν μέλη, Μεθόδους Κατασκευής – Δομητές και τη Μέθοδο toString()

- Να γραφεί Αλγόριθμος/Πρόγραμμα το οποίο δημιουργεί `2` αντικείμενα τύπου `Foititis`. Ο πρώτος φοιτητής δημιουργείται με το δεύτερο δομητή περνώντας απ' ευθείας τις τιμές και ο δεύτερος φοιτητής δημιουργείται με τον πρώτο κενό δομητή, ενώ οι υπόλοιπες τιμές του δίνονται στα αντίστοιχα `private` πεδία του αντικειμένου με τη χρήση των μεθόδων `setAM()`, `setTheory()`, `setErg()`. Και για τους `2` φοιτητές το πρόγραμμα εμφανίζει τον Αριθμό Μητρώου, **τον Κωδικό Μαθήματος ( ο οποίος δηλώνεται σαν `static` ),** το Βαθμό Θεωρίας και το Βαθμό Εργαστηρίου με την κλήση της μεθόδου `toString()` και Υπολογίζει και Εμφανίζει τον Τελικό Βαθμό των δύο φοιτητών με την κλήση των μεθόδων `findTelikos1()` και `findTelikos2()` αντίστοιχα. Αλλάζει την τιμή του Κωδικού Μαθήματος για το φοιτητή `1` και εμφανίζει τη νέα τιμή του Κωδικού Μαθήματος και για τους 2 φοιτητές. Αλλάζει την τιμή του Κωδικού Μαθήματος χρησιμοποιώντας το όνομα της κλάσης `Foititis` και εμφανίζει τη νέα τιμή του Κωδικού Μαθήματος και για τους 2 φοιτητές. Εμφανίζει την αρχική τιμή της `static` μεταβλητής `betterGrade`, η οποία δηλώνεται σε μια κλάση με μεθόδους `static` ( `ClassOfStaticMethods` ), καλεί τη `static` μέθοδο `setBetterGrade()`, η οποία βρίσκει το φοιτητή με το μεγαλύτερο Τελικό Βαθμό και εμφανίζει τη νέα τιμή της μεταβλητής `betterGrade`.

## Κλάση `Foititis`

Η Κλάση **`Foititis`** περιέχει :

- Τα πεδία Αριθμός Μητρώου Φοιτητή, Κωδικός Μαθήματος, Βαθμός Θεωρίας και Βαθμός Εργαστηρίου. Όλα τα πεδία δηλώνονται `private`, **εκτός από τον Κωδικό Μαθήματος, ο οποίος δηλώνεται σαν `static`.**
- Τη Μέθοδο Κατασκευής/Δομητή `Foititis()`, κενός δομητής.
- Τη Μέθοδο Κατασκευής `Foititis(int am, double th, double erg)`, η οποία δίνει παραμετρικά τις τιμές σε όλα τα πεδία του αντικειμένου **εκτός από τον Κωδικό Μαθήματος**.
- Τις μεθόδους `getAM()`, `getTheory()`, `getErg()` για τη Χρήση των τιμών των `private` πεδίων.
- Τις μεθόδους `setAM()`, `setTheory()`, `setErg()` για την Ανάθεση τιμών στα `private` πεδία.
- Τη Μέθοδο Υπολογισμού - Εμφάνισης Τελικού Βαθμού `findTelikos1()`.
- Τη Μέθοδο Υπολογισμού - Επιστροφής Τελικού Βαθμού `findTelikos2()`.
- Τη Μέθοδο Εμφάνισης των Στοιχείων του κάθε Φοιτητή `toString()`.

```
public class Foititis {

    private int AM;          // Αριθμός Μητρώου Φοιτητή
    static int KM = 6000232; // Κωδικός Μαθήματος
    private double Theory;   // Βαθμός Θεωρίας
    private double Erg;      // Βαθμός Εργαστηρίου

    // Μέθοδος Κατασκευής - Δομητής 1
    Foititis (){
    }

    // Μέθοδος Κατασκευής - Δομητής 2
    Foititis(int am, double th, double erg){
        AM = am;
        Theory = th;
        Erg = erg;
    }

    // Μέθοδοι get - Χρήση τιμών των private πεδίων
    int getAM(){
     return AM;
    }

    double getTheory(){
     return Theory;
    }

    double getErg(){
     return Erg;
    }

    // Μέθοδοι set - Ανάθεση τιμών στα private πεδία
    void setAM(int am ){
     AM = am;
    }
```

```
    void setTheory(double th){
     Theory = th;
    }

    void setErg(double erg){
     Erg = erg;
    }

    // Μέθοδος Υπολογισμού - Εμφάνισης Τελικού Βαθμού
    void findTelikos1(){
        System.out.println( "Τελικός Βαθμός Φοιτητή = "
                + (Theory * 0.6 + Erg * 0.4));
    }

    // Μέθοδος Υπολογισμού - Επιστροφής Τελικού Βαθμού
    double findTelikos2(){
        return Theory * 0.6 + Erg * 0.4;
    }

    // Μέθοδος Εμφάνισης Στοιχείων Φοιτητή
    public String toString() {
      String s = "\nΑριθμός Μητρώου = " + AM + "\nΚωδικός Μαθήματος = " + KM;
      s += "\nΒαθμός Θεωρίας = " + Theory + "\nΒαθμός Εργαστηρίου = " + Erg;
      return s;
    }
}
```

**Παρατήρηση**

- Η κλάση `Foititis` θα μπορούσε να περιέχει την πλήρη Μέθοδο Κατασκευής `Foititis(int am, int km, double th, double erg)`, η οποία δίνει παραμετρικά τις τιμές σε όλα τα πεδία του αντικειμένου, αλλά με τη δημιουργία ενός αντικειμένου με τον πλήρη δομητή θα άλλαζε η τιμή του `static` πεδίου `KM` για όλα τα αντικείμενα.

## Κλάση `ClassOfStaticMethods`

- Η κλάση `ClassOfStaticMethods` περιέχει τη `static` καθολική μεταβλητή `betterGrade` και τη `static` μέθοδο `setBetterGrade()`, η οποία βρίσκει και επιστρέφει το φοιτητή με το μεγαλύτερο Τελικό Βαθμό.

```
public class ClassOfStaticMethods {

    static int betterGrade = 0;   // Μεταβλητή static

    /* Μέθοδος static, η οποία βρίσκει και επιστρέφει το φοιτητή με το
    μεγαλύτερο Τελικό Βαθμό
    */

    static int setBetterGrade(Foititis f1, Foititis f2){
        if ( f1.findTelikos2() > f2.findTelikos2())
          betterGrade = 1;
        else
          betterGrade = 2;
        return betterGrade;
    }
}
```

## **Αλγόριθμος κλάσης** `Objects8Static` **– main()**

1. Δημιουργία Αντικειμένου 1ου Φοιτητή με ταυτόχρονη Εισαγωγή Στοιχείων ( **εκτός του Κωδικού Μαθήματος** ) με το Δομητή 2
2. Εμφάνιση στοιχείων 1ου Φοιτητή με τη μέθοδο `toString()`
3. Υπολογισμός - Εμφάνιση Τελικού Βαθμού 1ου φοιτητή με την κλήση της μεθόδου `findTelikos1()`
4. Εισαγωγή Στοιχείων 2ου Φοιτητή ( **εκτός του Κωδικού Μαθήματος** ) σε Μεταβλητές
5. Δημιουργία Αντικειμένου 2ου Φοιτητή με το Δομητή 2
6. Εμφάνιση στοιχείων 2ου Φοιτητή με τη μέθοδο `toString()`
7. Υπολογισμός - Εμφάνιση Τελικού Βαθμού 2ου φοιτητή με την κλήση της μεθόδου `findTelikos2()`
8. **Αλλαγή της τιμής του Κωδικού Μαθήματος για το φοιτητή 1**
9. Εμφάνιση της νέας τιμής του Κωδικού Μαθήματος και για τους 2 φοιτητές
10. **Αλλαγή της τιμής του Κωδικού Μαθήματος χρησιμοποιώντας το όνομα της κλάσης `Foititis`**
11. Εμφάνιση της νέας τιμής του Κωδικού Μαθήματος και για τους 2 φοιτητές
12. **Εμφάνιση της αρχικής τιμή της static μεταβλητής `betterGrade`, η οποία δηλώνεται σε μια κλάση με μεθόδους `static` (`ClassOfStaticMethods`)**
13. **Κλήση της `static` μεθόδου `setBetterGgrade()`, η οποία βρίσκει το φοιτητή με το μεγαλύτερο Τελικό Βαθμό.**
14. **Εμφάνιση της νέας τιμής της `static` μεταβλητής `betterGrade`.**

### **Πρόγραμμα**

```
public class Objects8Static {
/*  Πρόγραμμμα το οποίο δημιουργεί 2 αντικείμενα τύπου Foititis. Ο πρώτος
φοιτητής δημιουργείται με το δεύτερο δομητή περνώντας απ' ευθείας τις
τιμές,ενώ ο δεύτερος φοιτητής δημιουργείται με τον πρώτο κενό δομητή και οι
υπόλοιπες τιμές – εκτός του Κωδικού Μαθήματος που δηλώνεται static – δίνονται
στα αντίστοιχα πεδία του αντικειμένου με τη χρήση των μεθόδων setAM(),
setTheory(), setErg(). Και για τους 2 φοιτητές το πρόγραμμα εμφανίζει τον
Αριθμό Μητρώου, τον Κωδικό Μαθήματος, το Βαθμό Θεωρίας και το Βαθμό
Εργαστηρίου με την κλήση της μεθόδου toString() και Υπολογίζει και Εμφανίζει
τον Τελικό Βαθμό του κάθε φοιτητή.
Αλλάζει την τιμή του Κωδικού Μαθήματος για το φοιτητή 1 και εμφανίζει τη νέα
τιμή του Κωδικού Μαθήματος και για τους 2 φοιτητές. Αλλάζει την τιμή του
Κωδικού Μαθήματος χρησιμοποιώντας το όνομα της κλάσης Foititis και εμφανίζει
τη νέα τιμή του Κωδικού Μαθήματος και για τους 2 φοιτητές. Εμφανίζει την
αρχική τιμή της static μεταβλητής betterGrade, η οποία δηλώνεται σε μια κλάση
με μεθόδους static ( Classofstaticmethods ), καλεί τη static μέθοδο
setBetterGrade(), η οποία βρίσκει το φοιτητή με το μεγαλύτερο Τελικό Βαθμό
και εμφανίζει τη νέα τιμή της static μεταβλητής betterGrade.
*/
    public static void main(String[] args) {
    double Telikos;

    // Δημιουργία Αντικειμένου Φοιτητή 1 με το δομητή/Εισαγωγή Στοιχείων
    Foititis f1 = new Foititis(14013, 6.5, 8.2);

    // Εμφάνιση Στοιχείων Φοιτητή 1 με κλήση μεθόδου toString
    System.out.println ( "\nΣτοιχεία Φοιτητή 1");
    System.out.println ( f1 );

  // Υπολογισμός-Εμφάνιση Τελικού Βαθμού Φοιτητή 1-Κλήση μεθόδου findTelikos1
    f1.findTelikos1();

    // Εισαγωγή Στοιχείων Φοιτητή 2
```

```
        int am = 14014;
        double theory = 8.6;
        double erg = 8.0;

        // Δημιουργία Αντικειμένου Φοιτητή 2 με το δομητή
        Foititis f2 = new Foititis();

        // Ανάθεση τιμών στα private πεδία του φοιτητή 2 ΜΟΝΟ με μεθόδους set
        f2.setAM( am );
        f2.setTheory( theory );
        f2.setErg( erg );

        // Εμφάνιση Στοιχείων Φοιτητή 2 με κλήση μεθόδου toString
        System.out.println ( "\nΣτοιχεία Φοιτητή 2");
        System.out.println ( f2 );

      // Υπολογισμός-Εμφάνιση Τελικού Βαθμού Φοιτητή 2-Κλήση μεθόδου findTelikos2
       System.out.println ( "Τελικός Βαθμός Φοιτητή f2 = " + f2.findTelikos2() );

        // Αλλαγή Τιμής static μεταβλητής ΚΜ φοιτητή f1 με όνομα αντικειμένου
        f1.KM = 8000123;

         // Εμφάνιση Κωδικού Μαθήματος Φοιτητή 1
        System.out.println ( "\nΚωδικός Μαθήματος φοιτητή f1 = " + f1.KM);

        // Εμφάνιση Κωδικού Μαθήματος Φοιτητή 2
        System.out.println ( "\nΚωδικός Μαθήματος φοιτητή f2 = " + f2.KM);

        // Αλλαγή Τιμής static μεταβλητής ΚΜ  φοιτητή f1-f2 με όνομα κλάσης
        Foititis.KM = 8000456;

         // Εμφάνιση Κωδικού Μαθήματος Φοιτητή 1
        System.out.println ( "\nΚωδικός Μαθήματος φοιτητή f1 = " + f1.KM);

        // Εμφάνιση Κωδικού Μαθήματος Φοιτητή 2
        System.out.println ( "\nΚωδικός Μαθήματος φοιτητή f2 = " + f2.KM);

        // Εμφάνιση Αρχικής Τιμής static μεταβλητής betterGrade
        System.out.println ( "\nΑρχική Τιμή betterGrade = "
                                          + ClassOfStaticMethods.betterGrade);

        // Κλήση Μεθόδου setBetterGrade()- Εμφάνιση νέας Τιμής betterGrade
        System.out.println ( "\nΚαλύτερο Τελικό Βαθμό έχει ο Φοιτητής #"
                               + ClassOfStaticMethods.setBetterGrade( f1, f2));
    }
}
```

## Έξοδος Προγράμματος :

```
run:
Στοιχεία Φοιτητή 1

Αριθμός Μητρώου = 14013
Κωδικός Μαθήματος = 6000232
Βαθμός Θεωρίας = 6.5
Βαθμός Εργαστηρίου = 8.2
Τελικός Βαθμός Φοιτητή = 7.18

Στοιχεία Φοιτητή 2

Αριθμός Μητρώου = 14014
Κωδικός Μαθήματος = 6000232
Βαθμός Θεωρίας = 8.6
Βαθμός Εργαστηρίου = 8.0
Τελικός Βαθμός Φοιτητή f2 = 8.36
```

```
Κωδικός Μαθήματος φοιτητή f1 = 8000123

Κωδικός Μαθήματος φοιτητή f2 = 8000123

Κωδικός Μαθήματος φοιτητή f1 = 8000456

Κωδικός Μαθήματος φοιτητή f2 = 8000456

Αρχική Τιμή betterGgrade = 0

Καλύτερο Τελικό Βαθμό έχει ο Φοιτητής #2
BUILD SUCCESSFUL (total time: 0 seconds)
```

## Παρατήρηση

- Σε προβλήματα που πρέπει να δημιουργηθούν αντικείμενα, στα οποία κάποιο πεδίο τους πρέπει να παίρνει σαν τιμές κάποιους διαδοχικούς αριθμούς, ( π.χ. τα υποκαταστήματα μιας τράπεζας σ' όλη την Ελλάδα πρέπει να δίνουν τον επόμενο αριθμό σε κάθε καινούριο λογαριασμό ή η Γραμματεία να δίνει διαδοχικούς αριθμούς στον Αριθμό Μητρώου κάθε φοιτητή ), συνήθως δηλώνουμε μια `static` μεταβλητή σε κάποια κλάση με `static` μεταβλητές και μεθόδους, την οποία μεταβλητή ενημερώνουμε κάθε φορά που πρέπει να δημιουργήσουμε ένα νέο αντικείμενο, όπως φαίνεται στο Παράδειγμα που ακολουθεί :

### 7.9.2 Παράδειγμα Δημιουργίας Κλάσης η οποία περιέχει private Δεδομένα και τις αντίστοιχες Μεθόδους get και set, Μεθόδους σαν μέλη, Μεθόδους Κατασκευής – Δομητές, τη Μέθοδο toString() και κλάση με static μεταβλητές και μεθόδους

- Να γραφεί Αλγόριθμος/Πρόγραμμα δημιουργεί 3 αντικείμενα τύπου `Foititis`. Ο πρώτος φοιτητής δημιουργείται με τον πρώτο κενό δομητή και οι υπόλοιπες τιμές δίνονται στα αντίστοιχα `private` πεδία του αντικειμένου με τη χρήση των μεθόδων `setAM()`, `setKM()`, `setTheory()`, `setErg()`. Ο Αριθμός Μητρώου είναι η αρχική τιμή της `static` μεταβλητής `currentAM` στην κλάση `ClassStaticVariableMethod`. Ο δεύτερος φοιτητής δημιουργείται με το δεύτερο πλήρη δομητή περνώντας απ' ευθείας τις υπόλοιπες τιμές, όπου σαν Αριθμός Μητρώου περνάει η νέα τιμή της `static` μεταβλητής `currentAM`, η οποία αυξάνεται κατά `1`, ενώ ο τρίτος φοιτητής δημιουργείται με το δεύτερο πλήρη δομητή περνώντας απ' ευθείας τις υπόλοιπες τιμές, οι οποίες δίνονται σε μεταβλητές στη `main()`, όπου σαν Αριθμός Μητρώου περνάει η νέα τιμή της `static` μεταβλητής `currentAM`, η οποία αυξάνεται κατά `1`. Και για τους `3` φοιτητές το πρόγραμμα εμφανίζει τον Αριθμό Μητρώου, τον Κωδικό Μαθήματος, το Βαθμό Θεωρίας και το Βαθμό Εργαστηρίου με την κλήση της μεθόδου `toString()` και Υπολογίζει και Εμφανίζει τον Τελικό Βαθμό του κάθε φοιτητή με την κλήση της μεθόδου επιστροφής τιμής `findTelikos2()`. Με την Κλήση της Μεθόδου `findBetterGrade()` βρίσκει το φοιτητή με το μεγαλύτερο βαθμό και εμφανίζει την τελική τιμή της `static` μεταβλητής `currentAM` στην κλάση `ClassStaticVariableMethod`.

## Κλάση `Foititis`

Η Κλάση **`Foititis`** περιέχει :

- Τα πεδία Αριθμός Μητρώου Φοιτητή, Κωδικός Μαθήματος, Βαθμός Θεωρίας και Βαθμός Εργαστηρίου. Όλα τα πεδία δηλώνονται `private`.
- Τη Μέθοδο Κατασκευής/Δομητή `Foititis()`, κενός δομητής.
- Τη Μέθοδο Κατασκευής `Foititis(int am, int km, double th, double erg)`, η οποία δίνει παραμετρικά τις τιμές σε όλα τα πεδία του αντικειμένου.
- Τις μεθόδους `getAM()`,`getKM()`,`getTheory()`, `getErg()` για τη Χρήση των τιμών των `private` πεδίων.
- Τις μεθόδους `setAM()`, `setKM()`, `setTheory()`, `setErg()` για την Ανάθεση τιμών στα `private` πεδία.
- Τη Μέθοδο Υπολογισμού - Εμφάνισης Τελικού Βαθμού `findTelikos1()`.
- Τη Μέθοδο Υπολογισμού - Επιστροφής Τελικού Βαθμού `findTelikos2()`.
- Τη Μέθοδο Εμφάνισης των Στοιχείων του κάθε Φοιτητή `toString()`.

```
public class Foititis {

    private int AM;          // Αριθμός Μητρώου Φοιτητή
    private  int KM ;        // Κωδικός Μαθήματος
    private double Theory;   // Βαθμός Θεωρίας
    private double Erg;      // Βαθμός Εργαστηρίου

    // Μέθοδος Κατασκευής - Δομητής 1
    Foititis(){
    }

    // Μέθοδος Κατασκευής - Δομητής 2
    Foititis(int am, int km, double th, double erg){
        AM = am;
        KM = km;
        Theory = th;
        Erg = erg;
    }

    // Μέθοδοι get - Χρήση τιμών των private πεδίων
    int getAM(){
     return AM;
    }

    int getKM(){
     return KM;
    }

    double getTheory(){
     return Theory;
    }

    double getErg(){
     return Erg;
    }
```

```
    // Μέθοδοι set - Ανάθεση τιμών στα private πεδία
    void setAM(int am ){
     AM = am;
    }

    void setKM(int km ){
     KM = km;
    }

    void setTheory(double th){
     Theory = th;
    }

    void setErg(double erg){
     Erg = erg;
    }

    // Μέθοδος Υπολογισμού - Εμφάνισης Τελικού Βαθμού
    void findTelikos1(){
        System.out.println( "Τελικός Βαθμός Φοιτητή = "
                + (Theory * 0.6 + Erg * 0.4));
    }

    // Μέθοδος Υπολογισμού - Επιστροφής Τελικού Βαθμού
    double findTelikos2(){
        return Theory * 0.6 + Erg * 0.4;
    }

    // Μέθοδος Εμφάνισης Στοιχείων Φοιτητή
    public String toString() {
      String s = "\nΑριθμός Μητρώου = " + AM + "\nΚωδικός Μαθήματος = " + KM;
      s += "\nΒαθμός Θεωρίας = " + Theory + "\nΒαθμός Εργαστηρίου = " + Erg;
      return s;
    }
}
```

## Κλάση ClassStaticVariableMethod

- Η κλάση **ClassStaticVariableMethod** περιέχει τη `static` καθολική μεταβλητή **currentAM** και τη `static` μέθοδο `findBetterGrade()`, η οποία βρίσκει και επιστρέφει το φοιτητή με το μεγαλύτερο Τελικό Βαθμό.

```
public class ClassStaticVariableMethod {
  static int currentAM = 145001;   // Καθολική Μεταβλητή static

    /* Μέθοδος static, η οποία βρίσκει και επιστρέφει το φοιτητή με το
    μεγαλύτερο Τελικό Βαθμό
    */
    static int findBetterGrade(Foititis f1, Foititis f2){
        int betterGrade;
        if ( f1.findTelikos2() > f2.findTelikos2())
          betterGrade = 1;
        else
          betterGrade = 2;
        return betterGrade;
    }
}
```

## **Αλγόριθμος κλάσης** ObjectsClassStatic **– main()**

1. Δημιουργία Αντικειμένου 1ου Φοιτητή με τον κενό δομητή
2. Εισαγωγή Στοιχείων 1ου Φοιτητή ( **εκτός του Αριθμού Μητρώου** ) σε μεταβλητές της main()
3. Ανάθεση των τιμών στα private πεδία του αντικειμένου ( **Αριθμός Μητρώου = currentAM )**
4. Εμφάνιση στοιχείων 1ου Φοιτητή με τη μέθοδο toString()
5. Υπολογισμός - Εμφάνιση Τελικού Βαθμού 1ου φοιτητή με την κλήση της μεθόδου findTelikos2()
6. Ενημέρωση Τιμής static μεταβλητής currentAM
7. Δημιουργία Αντικειμένου 2ου Φοιτητή με το Δομητή 2, πέρασμα σταθερών τιμών ( **Αριθμός Μητρώου = νέα τιμή currentAM )**
8. Εμφάνιση στοιχείων 2ου Φοιτητή με τη μέθοδο toString()
9. Υπολογισμός - Εμφάνιση Τελικού Βαθμού 2ου φοιτητή με την κλήση της μεθόδου findTelikos2()
10. Ενημέρωση Τιμής static μεταβλητής currentAM
11. Εισαγωγή Στοιχείων 3ου Φοιτητή ( **εκτός του Αριθμού Μητρώου** ) σε μεταβλητές της main()
12. Δημιουργία Αντικειμένου 3ου Φοιτητή με το Δομητή 2, πέρασμα των τιμών παραμετρικά ( **Αριθμός Μητρώου = νέα τιμή currentAM )**
13. Εμφάνιση στοιχείων 3ου Φοιτητή με τη μέθοδο toString()
14. Υπολογισμός - Εμφάνιση Τελικού Βαθμού 2ου φοιτητή με την κλήση της μεθόδου findTelikos2()
15. **Κλήση της static μεθόδου findBetterGgrade(), η οποία βρίσκει το φοιτητή με το μεγαλύτερο Τελικό Βαθμό.**
16. **Εμφάνιση της τελικής τιμής της static μεταβλητής currentAM.**

### **Πρόγραμμα**

```
public class ObjectsClassStatic {
/*  Πρόγραμμμα το οποίο δημιουργεί 3 αντικείμενα τύπου Foititis. Ο πρώτος
φοιτητής δημιουργείται με τον πρώτο κενό δομητή και οι υπόλοιπες τιμές
δίνονται στα αντίστοιχα private πεδία του αντικειμένου με τη χρήση των
μεθόδων setAM(), setKM(), setTheory(), setErg(). Ο Αριθμός Μητρώου είναι η
αρχική τιμή της static μεταβλητής currentAM στην κλάση
ClassStaticVariableMethod. Ο δεύτερος φοιτητής δημιουργείται με το δεύτερο
πλήρη δομητή περνώντας απ' ευθείας τις υπόλοιπες τιμές, όπου σαν Αριθμός
Μητρώου περνάει η νέα τιμή της static μεταβλητής currentAM, η οποία αυξάνεται
κατά 1, ενώ ο τρίτος φοιτητής δημιουργείται με το δεύτερο πλήρη δομητή
περνώντας απ' ευθείας τις υπόλοιπες τιμές, οι οποίες δίνονται σε μεταβλητές
στη main(), όπου σαν Αριθμός Μητρώου περνάει η νέα τιμή της static μεταβλητής
currentAM, η οποία αυξάνεται κατά 1. Και για τους 3 φοιτητές το πρόγραμμα
εμφανίζει τον Αριθμό Μητρώου, τον Κωδικό Μαθήματος, το Βαθμό Θεωρίας και το
Βαθμό Εργαστηρίου με την κλήση της μεθόδου toString() και Υπολογίζει και
Εμφανίζει τον Τελικό Βαθμό του κάθε φοιτητή με την κλήση της μεθόδου
επιστροφής τιμής Telikos2(). Με την Κλήση της Μεθόδου findBetterGrade()
βρίσκει το φοιτητή με το μεγαλύτερο βαθμό και εμφανίζει την τελική τιμή
της static μεταβλητής currentAM στην κλάση ClassStaticVariableMethod.
*/
    public static void main(String[] args) {

    // Δημιουργία Αντικειμένου Φοιτητή 1 με το δομητή/Εισαγωγή Στοιχείων
    Foititis f1 = new Foititis();
```

```
    // Εισαγωγή Στοιχείων Φοιτητή 1
    int km = 14014;
    double theory = 8.6;
    double erg = 8.0;

     // Ανάθεση τιμών στα private πεδία του φοιτητή 1 MONO με μεθόδους set
    f1.setAM( ClassStaticVariableMethod.currentAM );
    f1.setKM( km );
    f1.setTheory( theory );
    f1.setErg( erg );

    // Εμφάνιση Στοιχείων Φοιτητή 1 με κλήση μεθόδου toString
    System.out.println ( "\nΣτοιχεία Φοιτητή 1");
    System.out.println ( f1 );

    // Υπολογισμός-Εμφάνιση Τελικού Βαθμού Φοιτητή 1
    // Κλήση μεθόδου findTelikos2()
    System.out.println ( "Τελικός Βαθμός Φοιτητή f1 = "
                          + f1.findTelikos2());

    // Ενημέρωση Τιμής static μεταβλητής currentAM
    ClassStaticVariableMethod.currentAM++;

    // Δημιουργία Αντικειμένου Φοιτητή 2 με το δομητή
    Foititis f2 =
          new Foititis(ClassStaticVariableMethod.currentAM, 14014, 6.5, 8.2);

    // Εμφάνιση Στοιχείων Φοιτητή 2 με κλήση μεθόδου toString
    System.out.println ( "\nΣτοιχεία Φοιτητή 2");
    System.out.println ( f2 );

   // Υπολογισμός-Εμφάνιση Τελικού Βαθμού Φοιτητή 2
    // Κλήση μεθόδου findTelikos2()
    System.out.println ( "Τελικός Βαθμός Φοιτητή f2 = "
                          + f2.findTelikos2() );

    // Εισαγωγή Στοιχείων Φοιτητή 3
    km = 14014;
    theory = 5.5;
    erg = 5.2;

    // Δημιουργία Αντικειμένου Φοιτητή 3 με το δομητή
    // Ενημέρωση Τιμής static μεταβλητής currentAM
    Foititis f3 =
        new Foititis(++ClassStaticVariableMethod.currentAM, km, theory, erg);

    // Εμφάνιση Στοιχείων Φοιτητή 3 με κλήση μεθόδου toString
    System.out.println ( "\nΣτοιχεία Φοιτητή 2");
    System.out.println ( f3 );

   // Υπολογισμός-Εμφάνιση Τελικού Βαθμού Φοιτητή 3
    // Κλήση μεθόδου findTelikos2()
    System.out.println ( "Τελικός Βαθμός Φοιτητή f3 = "
                          + f3.findTelikos2() );

   // Κλήση Μεθόδου findBetterGrade()
    System.out.println ( "\nΚαλύτερο Τελικό Βαθμό έχει ο Φοιτητής #"
                     + ClassStaticVariableMethod.findBetterGrade( f1, f2));

    // Εμφάνιση νέας Τιμής currentAM
    System.out.println ( "\nΤελική Τιμή static μεταβλητής currentAM = "
                               + ClassStaticVariableMethod.currentAM);
   }
}
```

**Έξοδος Προγράμματος :**

```
run:

Στοιχεία Φοιτητή 1

Αριθμός Μητρώου = 145001
Κωδικός Μαθήματος = 14014
Βαθμός Θεωρίας = 8.6
Βαθμός Εργαστηρίου = 8.0
Τελικός Βαθμός Φοιτητή f1 = 8.36

Στοιχεία Φοιτητή 2

Αριθμός Μητρώου = 145002
Κωδικός Μαθήματος = 14014
Βαθμός Θεωρίας = 6.5
Βαθμός Εργαστηρίου = 8.2
Τελικός Βαθμός Φοιτητή f2 = 7.18

Στοιχεία Φοιτητή 2

Αριθμός Μητρώου = 145003
Κωδικός Μαθήματος = 14014
Βαθμός Θεωρίας = 5.5
Βαθμός Εργαστηρίου = 5.2
Τελικός Βαθμός Φοιτητή f3 = 5.38

Καλύτερο Τελικό Βαθμό έχει ο Φοιτητής #1

Τελική Τιμή static μεταβλητής currentAM = 145003
BUILD SUCCESSFUL (total time: 0 seconds)
```

## 7.10 Αναφορές Αντικειμένων

Όταν δημιουργείται ένα αντικείμενο **f** της κλάσης **Foititis** με τον τελεστή **new**, στην πράξη το `f` περιέχει τη διεύθυνση – αναφορά στο αντικείμενο. Σε αντίθεση με τους κλασικούς τύπους δεδομένων όπου χρησιμοποιώντας το όνομα της αντίστοιχης μεταβλητής απλού τύπου έχουμε πρόσβαση στο περιεχόμενά της, δε συμβαίνει το ίδιο με τα αντικείμενα. Για παράδειγμα οι εντολές

```
int a = 5;
int b = a;
```

έχουν σαν αποτέλεσμα να αντιγραφεί το περιεχόμενο της μεταβλητής `a` στη μεταβλητή `b`, οπότε και οι 2 μεταβλητές `a` και `b` να έχουν το ίδιο περιεχόμενο, αλλά περιέχουν διαφορετικές διευθύνσεις στη μνήμη. Αν π.χ. χρησιμοποιήσω την εντολή

```
b = 7;
```

το περιεχόμενο της μεταβλητής `b` είναι πλέον διαφορετικό απ' αυτό της `a`.

Με τα αντικείμενα τα πράγματα είναι διαφορετικά. Με τις εντολές

```
Foititis f1 = new Foititis (14013, 6.5, 8.2);
Foititis f2 = new Foititis (14014, 8.5, 3.2);
f2 = f1;
```

δε γίνεται αντιγραφή των περιεχομένων του αντικειμένου **f1** στο **f2**. Απλώς οι μεταβλητές αναφορές αντικειμένων **f1** και **f2** δείχνουν στο ίδιο αντικείμενο ( περιέχουν την ίδια διεύθυνση ), το **f1**. Αν αλλάξω την τιμή σε ένα πεδίο του αντικειμένου που δείχνει το `f2`, π.χ. με την εντολή :

```
f2.Theory = 4.0;
```

αυτό θα επηρεάσει το αντικείμενο, στο οποίο δείχνει και το **f1**, δηλαδή το `f1.Theory` θα περιέχει πλέον την τιμή `4.0`.

Για να ανταλλάξω τα περιεχόμενα 2 αντικειμένων **f1** και **f2** μπορώ να χρησιμοποιήσω τις εντολές :

```
Foititis temp = f1;
f1 = f2;
f2 = temp;
```

με τις οποίες ανταλλάσω τις τιμές των μεταβλητών αναφοράς στα αντικείμενα **f1** και **f2.**

Οι **αναφορές αντικειμένων**, όταν περνάνε σαν παράμετροι σε μεθόδους, περνάνε **με τιμή**. Δηλαδή, οποιαδήποτε αλλαγή κι αν γίνει μέσα στη μέθοδο στη μεταβλητή αναφοράς δε γίνεται στην πραγματική, αλλά στην τυπική μεταβλητή αναφοράς.

Παρ' όλα αυτά **το αντικείμενο** στο οποίο δείχνει η μεταβλητή αναφοράς περνάει με **αναφορά**. Μπορούμε δηλαδή να αλλάξουμε κάποια ή όλα τα πεδία του αντικειμένου μέσα στη μέθοδο και αυτό να επηρεάσει το αντικείμενο.

### 7.10.1 Παράδειγμα Αλλαγών σε Τιμές Μεταβλητών Αναφοράς Αντικειμένων και σε Παραμέτρους - Μεταβλητές Αναφοράς Αντικειμένων σε Μεθόδους

- Να γραφεί Αλγόριθμος/Πρόγραμμα το οποίο δημιουργεί 2 αντικείμενα τύπου `Foititis`. **Η κλάση `Foititis` περιέχει και το πεδίο `telikosBathmos` για τον Τελικό Βαθμό**. Και οι δύο φοιτητές δημιουργούνται με το δομητή περνώντας απ' ευθείας τις τιμές. Και για τους 2 φοιτητές το πρόγραμμα **υπολογίζει τον Τελικό Βαθμό του κάθε φοιτητή** και εμφανίζει τον Αριθμό Μητρώου, τον Κωδικό Μαθήματος, το Βαθμό Θεωρίας το Βαθμό Εργαστηρίου **και τον Τελικό Βαθμό με την κλήση της μεθόδου `toString()`. Ανταλλάσει** τις τιμές των αναφορών αντικειμένων `f1` και `f2` και εμφανίζει τις νέες τιμές των πεδίων τους. **Ανταλλάσει** τις τιμές των αναφορών αντικειμένων `f1` και `f2` με την κλήση της μεθόδου `swap(f1, f2)` και εμφανίζει τις τιμές των πεδίων τους που **ΔΕΝ έχουν αλλάξει**. **Ανταλλάσει** τις τιμές του πεδίου Αριθμός Μητρώου των αντικειμένων `f1` και `f2` με την κλήση της μεθόδου `swapAM(f1, f2)` και εμφανίζει τις τιμές των πεδίων τους που **έχουν αλλάξει**.

#### Κλάση `Foititis`

Η Κλάση **`Foititis`** περιέχει :

- Τα πεδία Αριθμός Μητρώου Φοιτητή, Κωδικός Μαθήματος, Βαθμός Θεωρίας Βαθμός Εργαστηρίου και **Τελικός Βαθμός.** Όλα τα πεδία δηλώνονται `private`.
- Τη Μέθοδο Κατασκευής `Foititis(int am, int km, double th, double erg)`, η οποία δίνει παραμετρικά τις τιμές σε όλα τα πεδία του αντικειμένου, εκτός του **Τελικού Βαθμού**.
- Τις μεθόδους `getAM()`, `getKM()`, `getTheory()`, `getErg()`, `getTelikos()` για τη Χρήση των τιμών των `private` πεδίων.
- Τις μεθόδους `setAM()`, `setKM()`, `setTheory()`, `setErg()`, `setTelikos()` για την Ανάθεση τιμών στα `private` πεδία.
- Τη Μέθοδο Εμφάνισης των Στοιχείων του κάθε Φοιτητή `toString()`, όπου **έχει προστεθεί** και η εμφάνιση του **Τελικού Βαθμού**.

```
public class Foititis {
    private int AM;                     // Αριθμός Μητρώου Φοιτητή
    private int KM;                     // Κωδικός Μαθήματος
    private double Theory;              // Βαθμός Θεωρίας
    private double Erg;                 // Βαθμός Εργαστηρίου
    private double telikosBathmos;      // Τελικός Βαθμός

    // Μέθοδος Κατασκευής - Δομητής
        Foititis(int am, int km, double th, double erg){
        AM = am;
        KM = km;
        Theory = th;
        Erg = erg;
    }

    // Μέθοδοι get - Χρήση τιμών των private πεδίων
    int getAM(){
     return AM;
    }
```

```
    int getKM(){
     return KM;
    }

    double getTheory(){
     return Theory;
    }

    double getErg(){
     return Erg;
    }

    double getTelikosBathmos(){
     return telikosBathmos;
    }

    // Μέθοδοι set - Ανάθεση τιμών στα private πεδία
    void setAM(int am ){
     AM = am;
    }

    void setKM(int km ){
     AM = km;
    }

    void setTheory(double th){
     Theory = th;
    }

    void setErg(double erg){
     Erg = erg;
    }

    void setTelikosBathmos(){
     telikosBathmos = Theory * 0.6 + Erg * 0.4;
    }

    // Μέθοδος Εμφάνισης Στοιχείων Φοιτητή + Τελικός Βαθμός
    public String toString() {
      String s = "\nΑριθμός Μητρώου = " + AM + "\nΚωδικός Μαθήματος = " + KM;
      s += "\nΒαθμός Θεωρίας = " + Theory + "\nΒαθμός Εργαστηρίου = " + Erg
              + "\nΤελικός Βαθμός = " + telikosBathmos;
      return s;
    }
}
```

## Κλάση ClassOfStaticMethods

- Η κλάση ClassOfStaticMethods περιέχει τη static μέθοδο swapF(), η οποία ανταλλάσσει τις τιμές των μεταβλητών αναφοράς 2 αντικειμένων και τη static μέθοδο swapAM(), η οποία ανταλλάσει τις τιμές των Αριθμών Μητρώου 2 αντικειμένων.

```
public class ClassOfStaticMethods {

    // Μέθοδος ανταλλαγής των τιμών αναφοράς 2 αντικειμένων
    static void swapF(Foititis f1, Foititis f2){
    Foititis temp = f1;
    f1 = f2;
    f2 = temp;
    }
```

```
    // Μέθοδος ανταλλαγής των τιμών του Αριθμού Μητρώου 2 αντικειμένων
    static void swapAM(Foititis f1, Foititis f2){
    int temp = f1.getAM();
    f1.setAM(f2.getAM());
    f2.setAM(temp);
    }
}
```

## Αλγόριθμος κλάσης Objects9Swap – main()

1. Δημιουργία Αντικειμένου 1ου Φοιτητή με ταυτόχρονη Εισαγωγή Στοιχείων ( **εκτός του Τελικού Βαθμού** ) με το Δομητή
2. Υπολογισμός **Τελικού Βαθμού** 1ου φοιτητή με την κλήση της μεθόδου **setTelikosBathmos()**
3. Εμφάνιση ΟΛΩΝ των στοιχείων του 1ου Φοιτητή με τη μέθοδο toString()
4. Δημιουργία Αντικειμένου 2ου Φοιτητή με ταυτόχρονη Εισαγωγή Στοιχείων ( **εκτός του Τελικού Βαθμού** ) με το Δομητή
5. Υπολογισμός **Τελικού Βαθμού** 2ου φοιτητή με την κλήση της μεθόδου **setTelikosBathmos()**
6. Εμφάνιση ΟΛΩΝ των στοιχείων του 2ου Φοιτητή με τη μέθοδο toString()
7. **Ανταλλαγή** των τιμών των **μεταβλητών αναφοράς** 2 αντικειμένων
8. Εμφάνιση ΟΛΩΝ των στοιχείων του 1ου Φοιτητή με τη μέθοδο toString()
9. Εμφάνιση ΟΛΩΝ των στοιχείων του 2ου Φοιτητή με τη μέθοδο toString()
10. **Ανταλλαγή** των τιμών των **μεταβλητών αναφοράς** 2 αντικειμένων **με κλήση της static μεθόδου swapF()**
11. Εμφάνιση ΟΛΩΝ των στοιχείων του 1ου Φοιτητή με τη μέθοδο toString()
12. Εμφάνιση ΟΛΩΝ των στοιχείων του 2ου Φοιτητή με τη μέθοδο toString()
13. Αλλαγή της τιμής του Αριθμού Μητρώου **Ανταλλαγή** των τιμών των **μεταβλητών αναφοράς** 2 αντικειμένων **με κλήση της static μεθόδου swapAM()**
14. Εμφάνιση ΟΛΩΝ των στοιχείων του 1ου Φοιτητή με τη μέθοδο toString()
15. Εμφάνιση ΟΛΩΝ των στοιχείων του 2ου Φοιτητή με τη μέθοδο toString()

### Πρόγραμμα

```
public class Objects9Swap {
/*  Πρόγραμμμα το οποίο δημιουργεί 2 αντικείμενα τύπου Foititis. Και οι δύο
φοιτητές δημιουργούνται με το δεύτερο δομητή περνώντας απ' ευθείας τις τιμές.
Και για τους 2 φοιτητές το πρόγραμμα Υπολογίζει τον Τελικό Βαθμό του κάθε
φοιτητή και εμφανίζει τον Αριθμό Μητρώου, τον Κωδικό Μαθήματος, το Βαθμό
Θεωρίας το Βαθμό Εργαστηρίου και τον Τελικό Βαθμό με την κλήση της μεθόδου
toString(). Ανταλάσσει τις τιμές των αναφορών αντικειμένων f1 και f2 και
εμφανίζει τις νέες τιμές των πεδίων τους. Ανταλάσσει τις τιμές των αναφορών
αντικειμένων f1 και f2 με την κλήση της μεθόδου swapF(f1, f2) και εμφανίζει
τις τιμές των πεδίων τους που ΔΕΝ έχουν αλλάξει. Ανταλάσσει τις τιμές του
πεδίου Αριθμός Μητρώου των αντικειμένων f1 και f2 με την κλήση της μεθόδου
swapAM(f1, f2) και εμφανίζει τις τιμές των πεδίων τους που έχουν αλλάξει.
*/
   public static void main(String[] args) {

    // Δημιουργία Αντικειμένου Φοιτητή 1 με το δομητή/ Εισαγωγή Στοιχείων
    Foititis f1 = new Foititis(14013, 6000232, 6.5, 8.2);

     // Υπολογισμός Τελικού Βαθμού Φοιτητή 1 με μεθόδους get-set
    f1.setTelikosBathmos();

    // Εμφάνιση Στοιχείων Φοιτητή 1 με κλήση μεθόδου toString()
    System.out.println ( "\nΣτοιχεία Φοιτητή 1");
```

```
        System.out.println ( f1 );

        // Δημιουργία Αντικειμένου Φοιτητή 2 με το δομητή/ Εισαγωγή Στοιχείων
        Foititis f2 = new Foititis(14014, 6000232, 8.5, 8.3);

        // Υπολογισμός Τελικού Βαθμού Φοιτητή 2 με μεθόδους get-set
        F2.setTelikosBathmos();

        // Εμφάνιση Στοιχείων Φοιτητή 2 με κλήση μεθόδου toString()
        System.out.println ( "\nΣτοιχεία Φοιτητή 2");
        System.out.println ( f2 );

        //Ανταλλαγή αναφορών αντικειμένων
        Foititis temp = f1;
        f1 = f2;
        f2 = temp;

        // Εμφάνιση Στοιχείων Φοιτητών μετά την ανταλλαγή των f1, f2
        System.out.println ( "\nΣτοιχεία Φοιτητών μετά την ανταλλαγή f1, f2");

        // Εμφάνιση Στοιχείων Φοιτητή 1 με κλήση μεθόδου toString()
        System.out.println ( "\nΣτοιχεία Φοιτητή 1");
        System.out.println ( f1 );

        // Εμφάνιση Στοιχείων Φοιτητή 2 με κλήση μεθόδου toString()
        System.out.println ( "\nΣτοιχεία Φοιτητή 2");
        System.out.println ( f2 );

        // Κλήση μεθόδου swapF() - πέρασμα αναφορών αντικειμένων ΜΕ ΤΙΜΗ,
        // καμιά αλλαγή
        ClassOfStaticMethods.swapF(f1, f2);

        // Εμφάνιση Στοιχείων Φοιτητών μετά την κλήση της μεθόδου swapF(f1, f2)
        System.out.println ( "\nΣτοιχεία Φοιτητών μετά την κλήση της μεθόδου "
                + "swapF(f1, f2 )");

        // Εμφάνιση Στοιχείων Φοιτητή 1 με κλήση μεθόδου toString()
        System.out.println ( "\nΣτοιχεία Φοιτητή 1");
        System.out.println ( f1 );

        // Εμφάνιση Στοιχείων Φοιτητή 2 με κλήση μεθόδου toString()
        System.out.println ( "\nΣτοιχεία Φοιτητή 2");
        System.out.println ( f2 );

        // Κλήση μεθόδου swapAM() - πέρασμα αναφορών αντικειμένων, αλλαγή ΑΜ
        ClassOfStaticMethods.swapAM(f1, f2);

        // Εμφάνιση Στοιχείων Φοιτητών μετά την κλήση  μεθόδου swapAM(f1,f2)
        System.out.println ( "\nΣτοιχεία Φοιτητών μετά την κλήση της μεθόδου "
                + "swapAM(f1, f2)");

         // Εμφάνιση Στοιχείων Φοιτητή 1 με κλήση μεθόδου toString()
        System.out.println ( "\nΣτοιχεία Φοιτητή 1");
        System.out.println ( f1 );

        // Εμφάνιση Στοιχείων Φοιτητή 2 με κλήση μεθόδου toString()
        System.out.println ( "\nΣτοιχεία Φοιτητή 2");
        System.out.println ( f2 );
    }
}
```

**Έξοδος Προγράμματος** :

```
Στοιχεία Φοιτητή 1
```

```
Αριθμός Μητρώου = 14013
Κωδικός Μαθήματος = 6000232
Βαθμός Θεωρίας = 6.5
Βαθμός Εργαστηρίου = 8.2
Τελικός Βαθμός = 7.18

Στοιχεία Φοιτητή 2

Αριθμός Μητρώου = 14014
Κωδικός Μαθήματος = 6000232
Βαθμός Θεωρίας = 8.5
Βαθμός Εργαστηρίου = 8.3
Τελικός Βαθμός = 8.42

Στοιχεία Φοιτητών μετά την ανταλλαγή f1, f2
Στοιχεία Φοιτητή 1

Αριθμός Μητρώου = 14014
Κωδικός Μαθήματος = 6000232
Βαθμός Θεωρίας = 8.5
Βαθμός Εργαστηρίου = 8.3
Τελικός Βαθμός = 8.42

Στοιχεία Φοιτητή 2

Αριθμός Μητρώου = 14013
Κωδικός Μαθήματος = 6000232
Βαθμός Θεωρίας = 6.5
Βαθμός Εργαστηρίου = 8.2
Τελικός Βαθμός = 7.18

Στοιχεία Φοιτητών μετά την κλήση της μεθόδου swapF(f1, f2)
Στοιχεία Φοιτητή 1

Αριθμός Μητρώου = 14014
Κωδικός Μαθήματος = 6000232
Βαθμός Θεωρίας = 8.5
Βαθμός Εργαστηρίου = 8.3
Τελικός Βαθμός = 8.42

Στοιχεία Φοιτητή 2

Αριθμός Μητρώου = 14013
Κωδικός Μαθήματος = 6000232
Βαθμός Θεωρίας = 6.5
Βαθμός Εργαστηρίου = 8.2
Τελικός Βαθμός = 7.18

Στοιχεία Φοιτητών μετά την κλήση της μεθόδου swapAM(f1, f2)

Στοιχεία Φοιτητή 1

Αριθμός Μητρώου = 14013
Κωδικός Μαθήματος = 6000232
Βαθμός Θεωρίας = 8.5
Βαθμός Εργαστηρίου = 8.3
Τελικός Βαθμός = 8.42

Στοιχεία Φοιτητή 2

Αριθμός Μητρώου = 14014
Κωδικός Μαθήματος = 6000232
Βαθμός Θεωρίας = 6.5
Βαθμός Εργαστηρίου = 8.2
Τελικός Βαθμός = 7.1
BUILD SUCCESSFUL (total time: 0 seconds)
```

## 7.11 Αναφορές Πινάκων

Όταν δημιουργείται ένα αντικείμενο τύπου πίνακας με τον τελεστή **new**, στην πράξη το `pin` περιέχει τη διεύθυνση – αναφορά στο αντικείμενο τύπου πίνακας `pin`. Με τις εντολές

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

δε γίνεται αντιγραφή των περιεχομένων του πίνακα **pin1** στο **pin2**. Απλώς οι μεταβλητές αναφορές πινάκων **pin1** και **pin1** δείχνουν στο ίδιο αντικείμενο - πίνακας ( περιέχουν την ίδια διεύθυνση ), το **pin1**. Αν αλλάξω την τιμή σε ένα στοιχείο του πίνακα που δείχνει το **pin2**, π.χ. με την εντολή :

```
pin2[0] = 1000;
```

αυτό θα επηρεάσει τον πίνακα, στον οποίο δείχνει και το **pin1**, δηλαδή το **pin1[0]** θα περιέχει πλέον την τιμή **1000.**

Για να ανταλλάξω τα περιεχόμενα 2 πινάκων **pin1** και **pin2,** οι οποίοι έχουν δηλωθεί να είναι του ιδίου τύπου( π.χ. **int[5]**) μπορώ να χρησιμοποιήσω τις εντολές :

```
int temp[] = new int[5];
temp = pin1;
pin1 = pin2;
pin2 = temp;
```

με τις οποίες ανταλάσσω τις τιμές των μεταβλητών αναφοράς στους πίνακες **pin1** και **pin2.**

Οι **αναφορές πινάκων**, όταν περνάνε σαν παράμετροι σε μεθόδους, περνάνε **με τιμή**. Δηλαδή, οποιαδήποτε αλλαγή κι αν γίνει μέσα στη μέθοδο στη μεταβλητή αναφοράς δε γίνεται στην πραγματική, αλλά στην τυπική μεταβλητή αναφοράς.

Παρ' όλα αυτά **τα στοιχεία των πινάκων** στα οποίο δείχνουν οι μεταβλητές αναφοράς περνάνε με **αναφορά**. Μπορούμε δηλαδή να αλλάξουμε κάποια ή όλα τα στοιχεία του πίνακα μέσα στη μέθοδο και αυτό να επηρεάσει τον πίνακα.

### 7.11.1 Αντιγραφή – Ανταλλαγή Αναφορών Πινάκων και Στοιχείων ενός Πίνακα

- Να γραφεί Πρόγραμμα που να δημιουργεί `2` πίνακες ακεραίων `5` θέσεων και τους γεμίζει με τους ακέραιους `1-5` και `6-10`. Ανταλάσσει τις τιμές των αναφορών των πινάκων `pin1` και `pin2` και εμφανίζει τις νέες τιμές των πινάκων που έχουν αλλάξει. Ανταλάσσει τις τιμές των αναφορών των πινάκων `pin1` και `pin2` με την κλήση της μεθόδου `swapPin(pin1, pin2)` και εμφανίζει τις νέες τιμές των πινάκων που **ΔΕΝ έχουν αλλάξει**. Δημιουργεί ένα τυχαίο δείκτη μεταξύ `0` και `3` και καλεί τη μέθοδο `swapElement(pin1, index, index+1)` με την οποία ανταλάσσει τις τιμές των στοιχείων `pin1[index]` και `pin1[index+1]` και εμφανίζει τις νέες τιμές του πίνακα `pin1` που **έχουν αλλάξει**. Αποθηκεύει την αναφορά `pin1` στο `pin2`. Καλεί τη μέθοδο `swapElement(pin2, index, index+1)` με την οποία ανταλάσσει τις τιμές των στοιχείων `pin2[index]` και `pin2[index+1]` και εμφανίζει τις νέες τιμές των πινάκων `pin1` και `pin2` που έχουν αλλάξει και **έχουν τις ΙΔΙΕΣ τιμές**.

**Αλγόριθμος**

1. Δήλωση πίνακα `pin1`
2. Γέμισμα πίνακα `pin1` με τους αριθμούς `1-5`
3. Δήλωση πίνακα `pin2`
4. Γέμισμα πίνακα `pin2` με τους αριθμούς `6-10`
5. Εμφάνιση στοιχείων πίνακα `pin1`
6. Εμφάνιση στοιχείων πίνακα `pin2`
7. Ανταλλαγή Αναφορών `pin1, pin2`
8. Εμφάνιση στοιχείων πίνακα `pin1` μετά την ανταλλαγή
9. Εμφάνιση στοιχείων πίνακα `pin2` μετά την ανταλλαγή
10. Ανταλλαγή Αναφορών `pin1, pin2` με την κλήση της μεθόδου `swapPin()`
11. Εμφάνιση στοιχείων πίνακα `pin1` μετά την ανταλλαγή
12. Εμφάνιση στοιχείων πίνακα `pin2` μετά την ανταλλαγή
13. Δημιουργία τυχαίου δείκτη `index` στο `0:3`
14. Εμφάνιση δείκτη `index`
15. Κλήση μεθόδου ανταλλαγής των στοιχείων `pin1[index], pin1[index+1]`
16. Εμφάνιση στοιχείων πίνακα `pin1`
17. Εκχώρηση της αναφοράς `pin1` στο `pin2`
18. Εμφάνιση δείκτη `index`
19. Κλήση μεθόδου ανταλλαγής των στοιχείων `pin2[index], pin2[index+1]`
20. Εμφάνιση στοιχείων πίνακα `pin2`
21. Εμφάνιση στοιχείων πίνακα `pin1`

## Πρόγραμμα

```
public class SwapPinakes {
/*   Πρόγραμμμα το οποίο δημιουργεί 2 πίνακες ακεραίων 5 θέσεων και τους γεμίζει με τους ακέραιους 1-5 και 6-10. Ανταλάσσει τις τιμές των αναφορών των πινάκων pin1 και pin2 και εμφανίζει τις νέες τιμές των πινάκων που έχουν αλλάξει. Ανταλάσσει τις τιμές των αναφορών των πινάκων pin1 και pin2 με την κλήση της μεθόδου swapPin(pin1, pin2) και εμφανίζει τις νέες τιμές των πινάκων που ΔΕΝ έχουν αλλάξει. Δημιουργεί ένα τυχαίο δείκτη μεταξύ 0 και 4 και καλεί τη μέθοδο swapElement(pin1, index, index+1) με την οποία ανταλάσσει τις τιμές των στοιχείων pin1[index] και pin1[index+1] και εμφανίζει τις νέες τιμές του πίνακα pin1 που έχουν αλλάξει. Αποθηκεύει την αναφορά pin1 στο pin2. Καλεί τη μέθοδο swapElement(pin2, index, index+1) με την οποία ανταλάσσει τις τιμές των στοιχείων pin2[index] και pin2[index+1] και εμφανίζει τις νέες τιμές των πινάκων pin1 και pin2 που έχουν αλλάξει και έχουν τις ΙΔΙΕΣ τιμές.
*/

    static void swapPin(int pin1[], int pin2[]){
    // Μέθοδος ανταλλαγής των τιμών αναφοράς 2 πινάκων ακέραιων
        int[] temp = new int[5];
        temp = pin1;
        pin1 = pin2;
        pin2 = temp;
    }

    static void swapElement(int a[], int i, int j){
    // Μέθοδος ανταλλαγής των τιμών 2 στοιχείων ενός πίνακα ακέραιων
    int temp = a[i];
    a[i] = a[j];
    a[j] = temp;
    }

    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();
    }

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

        // Δήλωση πίνακα pin1
        int pin1[] = new int[5];

        // Γέμισμα πίνακα pin1 με τους αριθμούς 1-5
         for (i = 0;i <= pin1.length-1;i++)
            pin1[i] = i + 1;

        // Δήλωση πίνακα pin2
        int pin2[] = new int[5];

        // Γέμισμα πίνακα pin2 με τους αριθμούς 6-10
         for (i = 0;i <= pin2.length-1;i++)
            pin2[i] = i + 6;

        // Εμφάνιση στοιχείων πίνακα pin1
        System.out.print("\nΠίνακας pin1  = ");
        showPin(pin1);


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

```
        // Ανταλλαγή Αναφορών pin1, pin2
        int[] temp = new int[5];
        temp = pin1;
        pin1 = pin2;
        pin2 = temp;

        // Εμφάνιση στοιχείων πίνακα pin1 μετά την ανταλλαγή Αναφορών
        System.out.print("\nΠίνακας pin1 μετά την ανταλλαγή Αναφορών = ");
        showPin(pin1);

        // Εμφάνιση στοιχείων πίνακα pin2 μετά την ανταλλαγή Αναφορών
        System.out.print("Πίνακας pin2 μετά την ανταλλαγή Αναφορών = ");
        showPin(pin2);

        // Ανταλλαγή Αναφορών pin1, pin2 με κλήση μεθόδου swapPin
        swapPin( pin1, pin2);

        // Εμφάνιση στοιχείων πίνακα pin1 μετά την ανταλλαγή
        System.out.print("\nΠίνακας pin1 μετά την ανταλλαγή Αναφορών με τη" +
                         " μέθοδο swapPin = ");
        showPin(pin1);

        // Εμφάνιση στοιχείων πίνακα pin2 μετά την ανταλλαγή
        System.out.print("Πίνακας pin2 μετά την ανταλλαγή Αναφορών με τη"
                       + " μέθοδο swapPin = ");
        showPin(pin2);

       // Δημιουργία τυχαίου δείκτη 0:4
       int index = (int) ( Math.random()*4);

        // Εμφάνιση δείκτη
        System.out.print("\nΑνταλλαγή των τιμών των στοιχείων ");
        System.out.println("pin1[" + index + "], pin1[" + (index+1) + "]" );

        // Κλήση μεθόδου ανταλλαγής των στοιχείων pin1[index], pin1[index+1]
        swapElement( pin1, index, index+1);

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

        //Εκχώρηση της αναφοράς pin1 στο pin2
        pin2 = pin1;

        System.out.println("\nΕκχώρηση της αναφοράς pin1 στο pin2");

        // Εμφάνιση δείκτη
        System.out.print("\nΑνταλλαγή των τιμών των στοιχείων ");
        System.out.println("pin2[" + index + "], pin2[" + (index+1) + "]" );

        // Κλήση μεθόδου ανταλλαγής των στοιχείων pin2[index], pin2[index+1]
        swapElement( pin2, index, index+1);

        // Εμφάνιση στοιχείων πίνακα pin2
        System.out.print("Πίνακας pin2 μετά την ανταλλαγή στοιχείων  = ");
        showPin(pin2);

         // Εμφάνιση στοιχείων πίνακα pin1
        System.out.print("Πίνακας pin1 μετά την ανταλλαγή στοιχείων  = ");
        showPin(pin1);
        }
}
```

**Έξοδος Προγράμματος :**

```
run:

Πίνακας pin1  = 1 2 3 4 5
Πίνακας pin2  = 6 7 8 9 10

Πίνακας pin1 μετά την ανταλλαγή Αναφορών = 6 7 8 9 10
Πίνακας pin2 μετά την ανταλλαγή Αναφορών = 1 2 3 4 5

Πίνακας pin1 μετά την ανταλλαγή Αναφορών με τη μέθοδο swapPin = 6 7 8 9 10
Πίνακας pin2 μετά την ανταλλαγή Αναφορών με τη μέθοδο swapPin = 1 2 3 4 5

Ανταλλαγή των τιμών των στοιχείων pin1[0], pin1[1]
Πίνακας pin1 μετά την ανταλλαγή των στοιχείων  = 7 6 8 9 10

Εκχώρηση της αναφοράς pin1 στο pin2

Ανταλλαγή των τιμών των στοιχείων pin2[0], pin2[1]
Πίνακας pin2 μετά την ανταλλαγή των στοιχείων  = 6 7 8 9 10
Πίνακας pin1 μετά την ανταλλαγή των στοιχείων  = 6 7 8 9 10
BUILD SUCCESSFUL (total time: 0 seconds)
```

## 7.12 Πίνακες Αντικειμένων

Στις περισσότερες εφαρμογές απαιτείται η δημιουργία ενός πίνακα αντικειμένων, όταν τα αντικείμενα που θα χρειαστούμε είναι πολλά. Για το σκοπό αυτό θα χρειαστεί να δημιουργήσουμε **πρώτα** το **αντικείμενο** τύπου πίνακας και **μετά** να δημιουργήσουμε τα **αντικείμενα** του πίνακα.

**Παράδειγμα**

```
Foititis f[] = new Foititis[3]; // Δημιουργία Πίνακα 3 φοιτητών

f[0] = new Foititis(14014, 8.5, 8.3); // Δημιουργία Αντικειμένου Φοιτητή 0
                                      με το δομητή/Εισαγωγή Στοιχείων
```

### 7.12.1 Παράδειγμα Δημιουργίας Πίνακα Αντικειμένων – Υπολογισμός Τελικού/Μεγαλύτερου Τελικού Βαθμού – Ανταλλαγή Περιεχομένων 2 Αντικειμένων στον Πίνακα με χρήση μεθόδων

- Να γραφεί Αλγόριθμος/Πρόγραμμα το οποίο δημιουργεί έναν **πίνακα 3 αντικειμένων** τύπου `Foititis`. Η κλάση `Foititis` περιέχει και το πεδίο `telikosBathmos` για τον Τελικό Βαθμό. Και οι τρεις φοιτητές δημιουργούνται με το δομητή περνώντας απ' ευθείας τις τιμές, εκτός του πεδίου `KM` ( το οποίο δηλώνεται σαν `static` και παίρνει τιμή με δυναμική αρχικοποίηση ) και του Τελικού Βαθμού. Το πρόγραμμα υπολογίζει τον Τελικό Βαθμό ΟΛΩΝ των φοιτητών με την κλήση της μεθόδου `findTelikosBathmosAll(f)`, εμφανίζει τον Αριθμό Μητρώου, τον Κωδικό Μαθήματος, το Βαθμό Θεωρίας, το Βαθμό Εργαστηρίου και τον Τελικό Βαθμό με την κλήση της μεθόδου `displayStudents(f)` και βρίσκει το φοιτητή με το μεγαλύτερο Τελικό Βαθμό με την κλήση της μεθόδου `findBetterGrade(f)` και τον εμφανίζει. Κατόπιν δημιουργεί έναν τυχαίο ακέραιο αριθμό `index` μεταξύ `0` και `f.length-2` και καλεί τη μέθοδο `swapF(f, index, index+1)` με την οποία ανταλάσσει τα περιεχόμενα των `f[index]`, `f[index+1]` και εμφανίζει τον Αριθμό Μητρώου, τον Κωδικό Μαθήματος, το Βαθμό Θεωρίας, το Βαθμό Εργαστηρίου και τον Τελικό Βαθμό με την κλήση της μεθόδου `displayStudents(f)`. Όλες οι παραπάνω **μέθοδοι** θα γραφούν σε μια **ξεχωριστή κλάση με `static` μεθόδους** και όχι στην κλάση που βρίσκεται η `main()`.

#### Κλάση `Foititis`

Η Κλάση **`Foititis`** περιέχει :

- Τα πεδία Αριθμός Μητρώου Φοιτητή, Κωδικός Μαθήματος, Βαθμός Θεωρίας Βαθμός Εργαστηρίου και **Τελικός Βαθμός.** Όλα τα πεδία δηλώνονται `private`, εκτός του πεδίου `KM` το οποίο δηλώνεται σαν `static` και παίρνει τιμή με δυναμική αρχικοποίηση.

- Τη Μέθοδο Κατασκευής `Foititis(int am, double th, double erg)`, η οποία δίνει παραμετρικά τις τιμές σε όλα τα πεδία του αντικειμένου, εκτός του **Κωδικού Μαθήματος** και του **Τελικού Βαθμού**.
- Τις μεθόδους `getAM()`, `getKM()`, `getTheory()`, `getErg()`, `getTelikosBathmos()` για τη Χρήση των τιμών των `private` πεδίων.
- Τις μεθόδους `setAM()`, `setKM()`, `setTheory()`, `setErg()`, `setTelikosBathmos()` για την Ανάθεση τιμών στα `private` πεδία.
- Τη Μέθοδο Εμφάνισης των Στοιχείων του κάθε Φοιτητή `toString()`, όπου **έχει προστεθεί** και η εμφάνιση του **Τελικού Βαθμού**.

```
public class Foititis {
    private int AM;
    static int KM = 6000232;
    private double Theory;
    private double Erg;
    private double telikosBathmos;

    Foititis() {
     }

    Foititis(int am, double th, double erg){
        AM = am;
        Theory = th;
        Erg = erg;
    }

    int getAM(){
     return AM;
    }

    int getKM(){
     return KM;
    }

    double getTheory(){
     return Theory;
    }

    double getErg(){
     return Erg;
    }

    double getTelikosBathmos(){
     return telikosBathmos;
    }

    void setAM(int am ){
     AM = am;
    }

    void setKM(int km ){
     KM = km;
    }

    void setTheory(double th){
     Theory = th;
    }

    void setErg(double erg){
     Erg = erg;
    }
```

```
    void setTelikosBathmos(){
        telikosBathmos = Theory * 0.6 + Erg * 0.4;
    }

    public String toString() {
      String s = "\nΑριθμός Μητρώου = " + AM + "\nΚωδικός Μαθήματος = " + KM;
      s += "\nΒαθμός Θεωρίας = " + Theory + "\nΒαθμός Εργαστηρίου = " + Erg;
      s += "\nΤελικός Βαθμός = " + telikosBathmos;
      return s;
    }
}
```

## Κλάση ClassOfMethods

Περιέχει τις παρακάτω `static` μεθόδους :

`displayStudents(Foititis f[])` - Μέθοδος Εμφάνισης των Στοιχείων όλων των Φοιτητών

`findTelikosBathmosAll(Foititis f[])` - Μέθοδος Υπολογισμού του Τελικού Βαθμού όλων των Φοιτητών

`findBetterGrade(Foititis f[])` - Μέθοδος Εύρεσης του Μεγίστου Τελικού Βαθμού

`swapF(Foititis f[], int i, int j)`- Μέθοδος Αλλαγής του Περιεχομένου των στοιχείων `i` και `j` ενός Πίνακα Αντικειμένων Φοιτητών

```
public class ClassOfMethods  {
/* Κλάση που περιέχει ΜΟΝΟ τις παρακάτω static μεθόδους :
   displayStudents(Foititis f[]) - Μέθοδος Εμφάνισης των Στοιχείων όλων
                                   των Φοιτητών
    findTelikosBathmosAll(Foititis f[]) - Μέθοδος Υπολογισμού του
                                   Τελικού Βαθμού όλων των Φοιτητών
    findBetterGrade(Foititis f[]) -    Μέθοδος Εύρεσης του Μεγίστου Τελικού
                                    Βαθμού
    swap(Foititis f[], int i, int j)-   Μέθοδος Αλλαγής του Περιεχομένου των
                                    στοιχείων i και j ενός Πίνακα
                                     Αντικειμένων Φοιτητών
    */
    static int betterGrade = 0;   // Μεταβλητή static

    // Μέθοδος Εμφάνισης των Στοιχείων όλων των Φοιτητών
    static void displayStudents(Foititis f[]){
     // Εμφάνιση Στοιχείων Φοιτητών με κλήση μεθόδου toString
     for (int i = 0;i<f.length;i++) {
        System.out.println ( "\nΣτοιχεία Φοιτητή " + i);
        System.out.println ( f[i] );
     }
    System.out.println ( "\n");
    }
```

```
    static void findTelikosBathmosAll(Foititis f[]){
    // Μέθοδος Υπολογισμού του Τελικού Βαθμού όλων των Φοιτητών
    for (int i = 0;i<f.length;i++)
     f[i].setTelikosBathmos();
    }

    // Μέθοδος Εύρεσης του Μεγίστου Τελικού Βαθμού
    static int findBetterGrade(Foititis f[]){

        double maxTelikos = f[0].getTelikosBathmos() ;
        for (int i = 1;i<=f.length-1;i++) {
        if ( f[i].getTelikos() > maxTelikos){
          maxTelikos = f[i].getTelikosBathmos();
          betterGrade = i;
        }
    }
        return betterGrade;
}

    static void swap(Foititis f[], int i, int j){
    // Μέθοδος Αλλαγής του Περιεχομένου των στοιχείων i και j ενός Πίνακα
    // Αντικειμένων Φοιτητών
     Foititis temp = new Foititis();
     temp = f[i];
     f[i] = f[j];
     f[j] = temp;
    }
}
```

## Αλγόριθμος κλάσης PinObjects – main()

1. Δημιουργία Πίνακα Αντικειμένων 3 Φοιτητών
2. Δημιουργία 3 Αντικειμένων τύπου Foititis με ταυτόχρονη Εισαγωγή Στοιχείων ( **εκτός του Κωδικού Μαθήματος και του Τελικού Βαθμού** ) με τον πλήρη Δομητή
3. Υπολογισμός **Τελικού Βαθμού** για τους 3 φοιτητές με την κλήση της μεθόδου findTelikosBathmosAll(Foititis f[])
4. Εμφάνιση ΟΛΩΝ των στοιχείων των 3 Φοιτητών με τη μέθοδο displayStudents(Foititis f[])
5. Εύρεση του φοιτητή με το μεγαλύτερο Τελικό Βαθμό με την κλήση της μεθόδου findBetterGrade(Foititis f[])
6. Εμφάνιση της θέσης του φοιτητή με το μεγαλύτερο Τελικό Βαθμό
7. Δημιουργία ενός τυχαίου ακέραιου αριθμού index μεταξύ 0 και f.length-2
8. Ανταλλαγή των περιεχομένων των f[index], f[index+1] με την κλήση της μεθόδου swapF(Foititis f[], int i, int j)
9. Εμφάνιση ΟΛΩΝ των στοιχείων των 3 Φοιτητών με τη μέθοδο displayStudents(Foititis f[])

**Πρόγραμμα**

```
public class PinObjects {
/*
Να γραφεί Αλγόριθμος/Πρόγραμμμα το οποίο δημιουργεί έναν πίνακα 3
αντικειμένων τύπου Foititis. Η κλάση Foititis περιέχει και το πεδίο
telikosBathmos για τον Τελικό Βαθμό. Και οι τρεις φοιτητές δημιουργούνται με
το δομητή περνώντας απ' ευθείας τις τιμές, εκτός του πεδίου ΚΜ το οποίο
δηλώνεται σαν static και παίρνει τιμή με δυναμική αρχικοποίηση και του
Τελικού Βαθμού. Το πρόγραμμα υπολογίζει τον Τελικό Βαθμό του κάθε φοιτητή με
την κλήση της μεθόδου findTelikosBathmosAll(f), εμφανίζει τον Αριθμό Μητρώου,
τον Κωδικό Μαθήματος, το Βαθμό Θεωρίας, το Βαθμό Εργαστηρίου και τον Τελικό
Βαθμό με την κλήση της μεθόδου displayStudents(f) και βρίσκει το φοιτητή με
το μεγαλύτερο Τελικό Βαθμό με την κλήση της μεθόδου findBetterGrade(f) και
τον εμφανίζει. Κατόπιν δημιουργεί έναν τυχαίο ακέραιο αριθμό index μεταξύ 0
και f.length-2 και καλεί τη μέθοδο swap(f, index, index+1) με την οποία
ανταλάσσει τα περιεχόμενα των f[index], f[index+1] και εμφανίζει τον Αριθμό
Μητρώου, τον Κωδικό Μαθήματος, το Βαθμό Θεωρίας, το Βαθμό Εργαστηρίου και τον
Τελικό Βαθμό με την κλήση της μεθόδου displayStudents(f).
*/
    public static void main(String[] args) {

    // Ανάθεση Τιμής στο Μέγεθος του Πίνακα
    int n = 3;

    // Δημιουργία Πίνακα 3 φοιτητών
  Foititis f[] = new Foititis[n];

  // Δημιουργία Αντικειμένων Φοιτητών με το δομητή
  f[0] = new Foititis(14013, 6.0, 6.4);
  f[1] = new Foititis(14014, 8.6, 8.0);
  f[2] = new Foititis(14015, 4.6, 5.0);

// Υπολογισμός Τελικού Βαθμού Φοιτητών με κλήση μεθόδου findTelikosBathmosAll
  ClassOfMethods.findTelikosBathmosAll(f);

  // Εμφάνιση Στοιχείων Φοιτητών με κλήση μεθόδου displayStudents
  ClassOfMethods.displayStudents(f);

  // Εύρεση Μεγίστου Τελικού Βαθμού με κλήση μεθόδου findBetterGrade
  int thesiMax = ClassOfMethods.findBetterGrade(f);
  System.out.println ( "Μέγιστο Τελικό Βαθμό = "
    + f[thesiMax].getTelikosBathmos() + " έχει ο Φοιτητής " + thesiMax);

  // Δημιουργία τυχαίου ακεραίου αριθμού μεταξύ 0 και f.length-2
  int index = (int) (Math.random()*2);

  // Εμφάνιση τυχαίου ακεραίου αριθμού μεταξύ 0 και f.length-2
 System.out.println( "\n\nΑνταλλαγή f[" + index + "] and f[" + (index+1) + "]");

  // Αλλαγή του Περιεχομένου των στοιχείων index και index + 1 του Πίνακα
  // Αντικειμένων Φοιτητών
  ClassOfMethods.swapF(f, index, (index + 1) );

  // Εμφάνιση Στοιχείων Φοιτητών με κλήση μεθόδου displayStudents()
  ClassOfMethods.displayStudents(f);

   }
}
```

**Έξοδος Προγράμματος :**

```
run:

Στοιχεία Φοιτητή 0

Αριθμός Μητρώου = 14013
Κωδικός Μαθήματος = 6000232
Βαθμός Θεωρίας = 6.0
Βαθμός Εργαστηρίου = 6.4
Τελικός Βαθμός = 6.16

Στοιχεία Φοιτητή 1

Αριθμός Μητρώου = 14014
Κωδικός Μαθήματος = 6000232
Βαθμός Θεωρίας = 8.6
Βαθμός Εργαστηρίου = 8.0
Τελικός Βαθμός = 8.36

Στοιχεία Φοιτητή 2

Αριθμός Μητρώου = 14015
Κωδικός Μαθήματος = 6000232
Βαθμός Θεωρίας = 4.6
Βαθμός Εργαστηρίου = 5.0
Τελικός Βαθμός = 4.76


Μέγιστο Τελικό Βαθμό = 8.36 έχει ο Φοιτητής 1


Ανταλλαγή f[1] and f[2]

Στοιχεία Φοιτητή 0

Αριθμός Μητρώου = 14013
Κωδικός Μαθήματος = 6000232
Βαθμός Θεωρίας = 6.0
Βαθμός Εργαστηρίου = 6.4
Τελικός Βαθμός = 6.16

Στοιχεία Φοιτητή 1

Αριθμός Μητρώου = 14015
Κωδικός Μαθήματος = 6000232
Βαθμός Θεωρίας = 4.6
Βαθμός Εργαστηρίου = 5.0
Τελικός Βαθμός = 4.76

Στοιχεία Φοιτητή 2

Αριθμός Μητρώου = 14014
Κωδικός Μαθήματος = 6000232
Βαθμός Θεωρίας = 8.6
Βαθμός Εργαστηρίου = 8.0
Τελικός Βαθμός = 8.36
BUILD SUCCESSFUL (total time: 0 seconds)
```