Προχωρημένος Προγραμματισμός

Σύνθεση και Συνάθροιση

ΕΛΕΥΘΕΡΙΟΣ ΚΟΣΜΑΣ

ΧΕΙΜΕΡΙΝΟ ΕΞΑΜΗΝΟ 2022-2023 | ΤΜΗΜΑ ΗΛΕΚΤΡΟΛΟΓΩΝ ΜΗΧΑΝΙΚΩΝ ΚΑΙ ΜΗΧΑΝΙΚΩΝ ΥΠΟΛΟΓΙΣΤΩΝ

Περίληψη

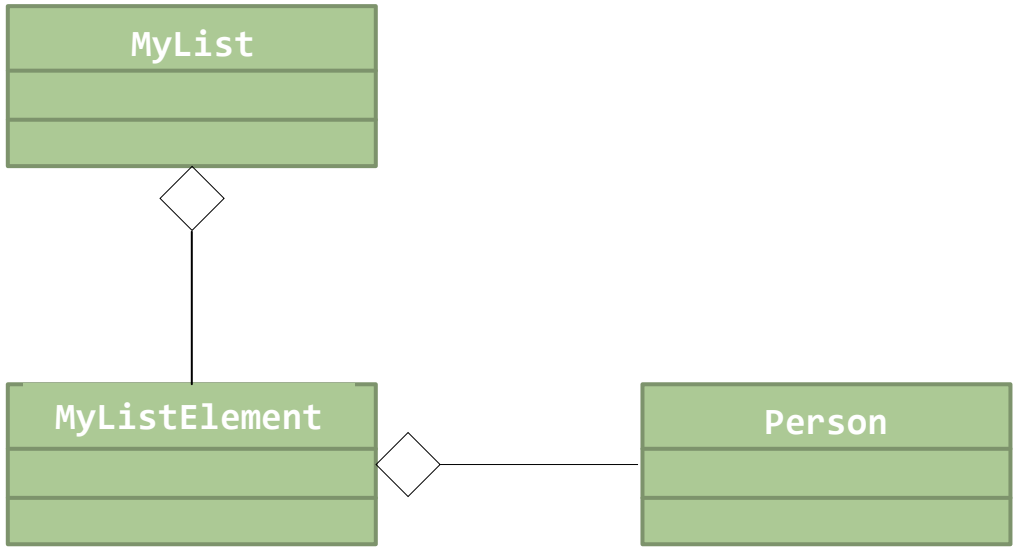
Σήμερα ...

- θα αναφερθούμε στα UML διαγράμματα
- θα συζητήσουμε τις σχέσεις σύνθεσης και συνάθροισης μεταξύ των κλάσεων

- θα μελετήσουμε ένα ολοκληρωμένο παράδειγμα

Η γλώσσα UML

- η UML (Unified Modeling Language) είναι μια γλώσσα για να περιγράφουμε και να καταλαβαίνουμε τον κώδικα μας
  - τα UML διαγράμματα παρέχουν μια οπτικοποίηση των σχέσεων μεταξύ των κλάσεων
    - π.χ.



Σχέσεις κλάσεων

- θέμα που προκύπτει όταν έχουμε κλάσεις που περιέχουν αντικείμενα άλλων κλάσεων:

  ? πότε και πού θα γίνεται η δημιουργία των αντικειμένων και πότε η καταστροφή τους;

  π.χ.,

  1. τα αντικείμενα τύπου `MyListElement` στο προηγούμενο παράδειγμα
     - δημιουργούνται μέσα στην κλάση `MyList`
     - καταστρέφονται είτε μέσα στην `MyList` ή αν το αντίστοιχο αντικείμενο `MyList` καταστραφεί
     - αλλαγές σε αντικείμενα `MyListElement` γίνονται μόνο μέσα στη `MyList`
  2. τα αντικείμενα τύπου `Person` που χρησιμοποιούνται στην `MyListElement`
     - δημιουργούνται εκτός της κλάσης
     - μπορεί να υπάρχουν αφού καταστραφεί η κλάση
     - αλλαγές στα αντικείμενα `Person` επηρεάζουν και τα περιεχόμενα της `MyList` (και το αντίστροφο)

- συχνά οι σχέσεις του πρώτου τύπου λέγονται σχέσεις σύνθεσης, ενώ του δεύτερου σχέσεις συνάθροισης

Σχέση σύνθεσης (Composition)

- η κλάση `A` έχει σχέση σύνθεσης με την κλάση `B`, αν το αντικείμενο της κλάσης `A` αποτελείται από αντικείμενα της κλάσης `B`
- τα αντικείμενα της κλάσης `B`
  - δημιουργούνται από την κλάση `A`
  - δεν υπάρχουν εκτός της κλάσης `A`
  - καταστρέφονται όταν καταστρέφεται το (αντίστοιχο) αντικείμενο της κλάσης `A`

- παραδείγματα:
  - ένας άνθρωπος αποτελείται από μέρη του σώματος: κεφάλι, πόδια, χέρια κλπ.
  - ένα κτήριο αποτελείται από τοίχους, δωμάτια, πόρτες, κλπ.

☞ στην περίπτωση μας η κλάση `MyList` έχει σχέση σύνθεσης με την κλάση `MyListElement`

Σχέση συνάθροισης (Aggregation)

- η κλάση `A` έχει σχέση συνάθροισης με την κλάση `B`, αν το αντικείμενο (ή αντικείμενα) της κλάσης `B` ανήκουν στο αντικείμενο της κλάσης `A`
- τα αντικείμενα της κλάσης `B`
  - έχουν υπόσταση και εκτός της κλάσης `A`
  - όταν καταστρέφεται το (αντίστοιχο) αντικείμενο της κλάσης `A` δεν καταστρέφονται απαραίτητα και τα αντικείμενα της κλάσης `B`

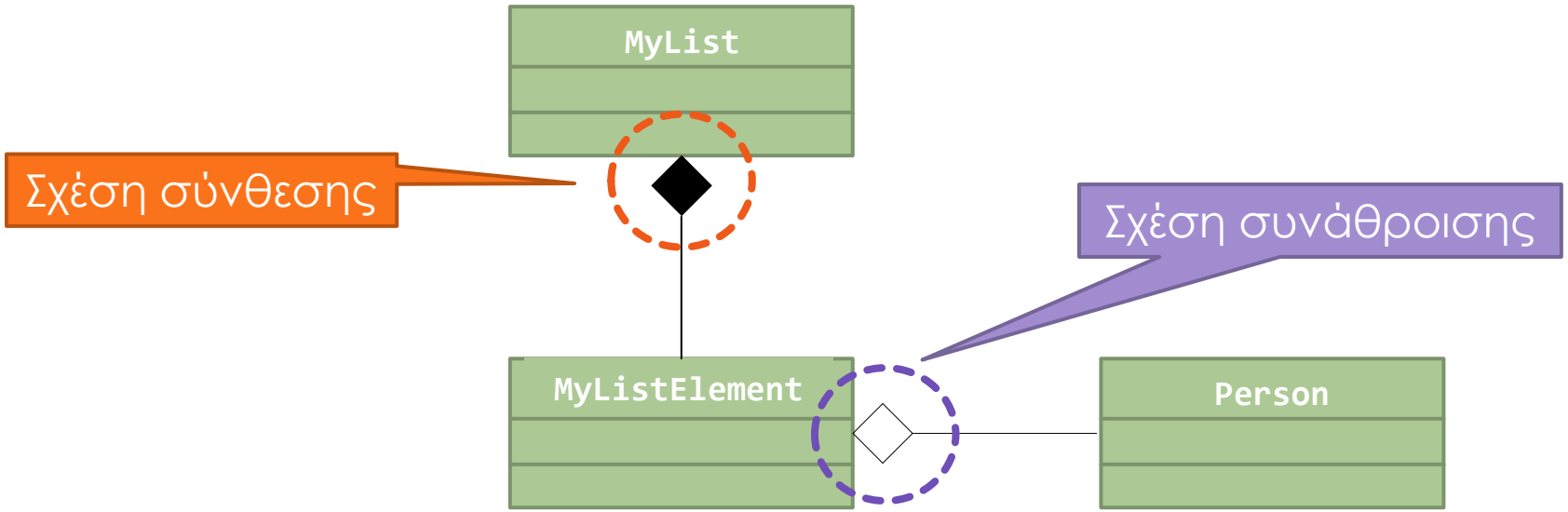
- παραδείγματα:
  - σε έναν άνθρωπο μπορεί να ανήκει ένα αυτοκίνητο, ρούχα, κλπ.
  - ένα κτήριο μπορεί να έχει μέσα ανθρώπους, έπιπλα, κλπ.

☞ στην περίπτωση μας η κλάση `MyListElement` έχει σχέση συνάθροισης με την κλάση `Person`

Σύνθεση και Συνάθροιση

UML διαγράμματα

- για να ξεχωρίζουν μεταξύ τους (κάποιες φορές) αναπαριστώνται διαφορετικά στα UML διαγράμματα



Σύνθεση και Συνάθροιση
Ορισμός σχέσης

- το αν θα είναι μια σχέση, σχέση σύνθεσης ή συνάθροισης εξαρτάται κατά πολύ:
  1. από την υλοποίηση μας και
  2. το σχεδιασμό

  - π.χ., σε ένα διαφορετικό πρόγραμμα μπορεί να επαναχρησιμοποιούμε το `MyListElement`

  - π.χ., σε μία διαφορετική εφαρμογή, τα ανθρώπινα όργανα υπάρχουν και χωρίς τον άνθρωπο

Σύνθεση και Συνάθροιση

Ορισμός σχέσης - Προσοχή!

- ο διαχωρισμός σε σχέσεις συνάθροισης και σύνθεσης είναι ως ένα βαθμό ένας φορμαλισμός
  - μην «κολλήσετε» προσπαθώντας να ορίσετε τη σχέση
  - ☞ το σημαντικό είναι όταν δημιουργείτε το πρόγραμμα σας να σκεφτείτε:
    - ποιες κλάσεις χρειάζονται τα αντικείμενα που δημιουργούνται
    - πότε πρέπει να δημιουργηθούν μέσα στον κώδικα
    - ποιες κλάσεις επηρεάζονται όταν αλλάζουν
  - δεν υπάρχει **χρυσός κανόνας**
    - συνήθως, το κάθε πρόγραμμα μπορεί να σχεδιαστεί με πολλούς τρόπους
    - διαλέξτε αυτόν που θα κάνει το πρόγραμμα:
      - πιο απλό, ευανάγνωστο,
      - εύκολο να επεκταθεί, να ξαναχρησιμοποιηθεί και να διατηρηθεί.

Ολοκληρωμένο παράδειγμα

Λογισμικό για τμήμα ΤΕΙ
Περιγραφή

Θέλουμε να δημιουργήσουμε ένα λογισμικό για ένα τμήμα του ΤΕΙ

- το τμήμα έχει
  - 4 φοιτητές οπού ο καθένας έχει ένα όνομα και ένα αριθμό μητρώου (ΑΜ)
  - 2 καθηγητές που ο καθένας έχει ένα όνομα και ένα ΑΦΜ
- το τμήμα δίνει 2 μαθήματα
- το κάθε μάθημα
  - έχει κωδικό και όνομα και κάποιες διδακτικές μονάδες.
  - ανατίθεται σε ένα καθηγητή
- οι φοιτητές γράφονται σε κάποιο μάθημα
  - αν περάσουν το μάθημα παίρνουν τις μονάδες
- θέλουμε να μπορούμε να τυπώσουμε τις πληροφορίες για το μάθημα
  - το όνομα, τον καθηγητή και τη λίστα των φοιτητών που παίρνουν το μάθημα

Λογισμικό για τμήμα ΤΕΙ

Περιγραφή - Υποψήφιες κλάσεις, μέθοδοι και μηνύματα

Θέλουμε να δημιουργήσουμε ένα λογισμικό για ένα **τμήμα** του ΤΕΙ

- το **τμήμα** έχει
  - 4 **φοιτητές** οπού ο καθένας έχει ένα **όνομα** και ένα **αριθμό μητρώου** (**ΑΜ**)
  - 2 **καθηγητές** που ο καθένας έχει ένα **όνομα** και ένα **ΑΦΜ**
- το τμήμα δίνει 2 **μαθήματα**
- το κάθε **μάθημα**
  - έχει **κωδικό** και **όνομα** και κάποιες **διδακτικές μονάδες**.
  - **ανατίθεται** σε ένα καθηγητή
- οι φοιτητές **γράφονται** σε κάποιο μάθημα
  - αν **περάσουν** το μάθημα **παίρνουν** τις μονάδες
- θέλουμε να μπορούμε να **τυπώσουμε** τις πληροφορίες για το μάθημα
  - το **όνομα**, τον **καθηγητή** και τη **λίστα** των **φοιτητών** που παίρνουν το μάθημα

Λογισμικό για τμήμα ΤΕΙ

Περιγραφή - Υποψήφιες κλάσεις, μέθοδοι και μηνύματα

- ουσιαστικά:
  - τμήμα
  - φοιτητές
  - καθηγητές
  - μαθήματα
  - όνομα
  - ΑΜ, ΑΦΜ, κωδικός
  - βαθμός
  - λίστα φοιτητών
- ☞ τα ουσιαστικά είναι υποψήφια για κλάσεις ή πεδία

- ρήματα:
  - ανατίθεται
  - εγγράφεται
  - τυπώνει
  - περνάω μάθημα
  - παίρνω μονάδες
- ☞ τα ρήματα είναι υποψήφια για να γίνουν μέθοδοι και μηνύματα μεταξύ αντικειμένων

Λογισμικό για τμήμα ΤΕΙ

Περιγραφή - Υποψήφιες κλάσεις, μέθοδοι και μηνύματα

- ουσιαστικά:
  - τμήμα
  - φοιτητές
  - καθηγητές
  - μαθήματα
  - όνομα
  - ΑΜ, ΑΦΜ, κωδικός
  - βαθμός
  - λίστα φοιτητών

☞ τα ουσιαστικά είναι υποψήφια για κλάσεις ή πεδία

- ρήματα:
  - ανατίθεται
  - εγγράφεται
  - τυπώνει
  - περνάω μάθημα
  - παίρνω μονάδες

☞ τα ρήματα είναι υποψήφια για να γίνουν μέθοδοι και μηνύματα μεταξύ αντικειμένων

Όλα τα ουσιαστικά **μπορούν** να γίνουν κλάσεις αλλά συνήθως **επιλέγουμε** αυτά για τα οποία υπάρχει **αρκετή πολυπλοκότητα**

Λογισμικό για τμήμα ΤΕΙ
Υλοποίηση - Κλάση Professor

- κρατάει το όνομα και το ΑΦΜ του καθηγητή
- ενδεχομένως να κρατάει και τα μαθήματα που έχει αναλάβει

? η μέθοδος για να αναλάβει ο καθηγητής ένα μάθημα θα πρέπει να είναι εδώ ή στην κλάση του μαθήματος;

Λογισμικό για τμήμα ΤΕΙ
Υλοποίηση - Κλάση Student

- κρατάει το όνομα του φοιτητή και τις μονάδες που έχει πάρει μέχρι τώρα
- ενδεχομένως να κρατάει και τα μαθήματα που παίρνει
- ενδεχομένως να κρατάει και τη λίστα με τα μαθήματα που έχει περάσει

☞ χρειαζόμαστε μέθοδο:
- για να γραφτεί ο φοιτητής στο μάθημα, ή
- να το περάσει

? ή καλύτερα να τις βάλουμε στην κλάση του μαθήματος;

Λογισμικό για τμήμα ΤΕΙ

Υλοποίηση - Κλάση Course

- κρατάει:
  - το όνομα του μαθήματος
  - τις μονάδες του μαθήματος
  - τον καθηγητή που κάνει το μάθημα
  - τους φοιτητές που παίρνουν το μάθημα
- ? τίποτα άλλο;
  - ? τι θα κάνουμε με τους βαθμούς και το ποιος πέρασε το μάθημα;
- μέθοδοι:
  - ανάθεση καθηγητή
  - εγγραφή φοιτητή στο μάθημα
  - καταχώρηση βαθμών στους φοιτητές

Λογισμικό για τμήμα ΤΕΙ

Υλοποίηση - Κλάση Department

τα βάζει όλα μαζί

- εδώ δημιουργούμε τους φοιτητές, τους καθηγητές και τα μαθήματα
  - οι φοιτητές και οι καθηγητές ως άτομα θα μπορούσαν να υπάρχουν και εκτός του τμήματος
- εδώ δημιουργούμε τη `main`

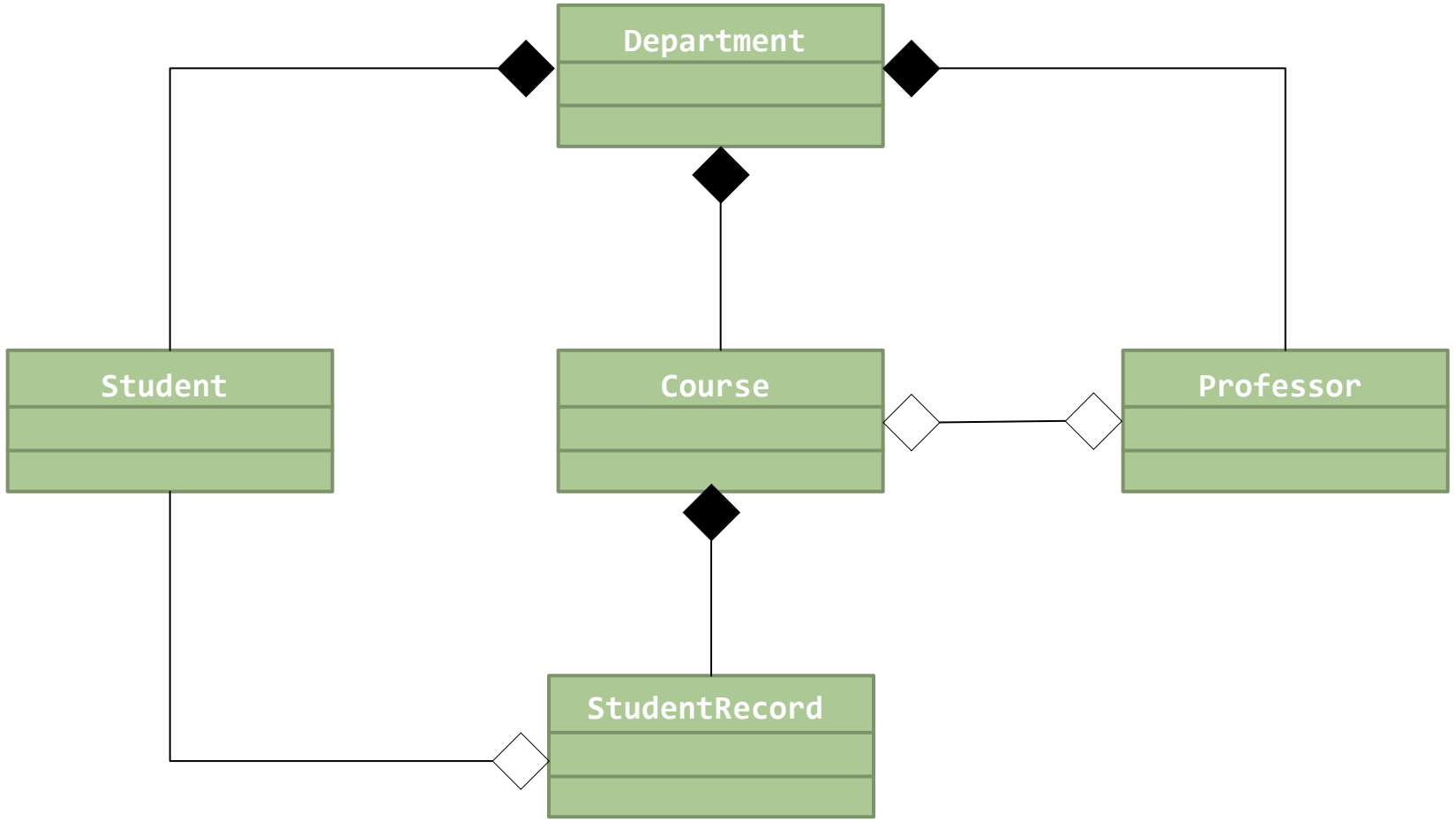
? χρειαζόμαστε άλλη κλάση;

Λογισμικό για τμήμα ΤΕΙ
Υλοποίηση - Κλάση StudentRecord

- χρειαζόμαστε να κρατάμε για κάθε φοιτητή
  - τις πληροφορίες του (π.χ. αυτά που έχουμε στο eclass)
  - το βαθμό του
- μας βολεύει να δημιουργήσουμε μια καινούρια κλάση που να βάζει μαζί αυτές τις πληροφορίες

Λογισμικό για τμήμα ΤΕΙ

Υλοποίηση - UML διάγραμμα



Λογισμικό για τμήμα ΤΕΙ
Υλοποίηση - Αποθήκευση φοιτητών

- η κλάση `Course` πρέπει να αποθηκεύει τους φοιτητές που παίρνουν το μάθημα
  - δε ξέρουμε εκ των προτέρων πόσοι φοιτητές θα πάρουν το μάθημα

☞ θα χρησιμοποιήσουμε την κλάση `ArrayList` για να τους αποθηκεύσουμε

Λογισμικό για τμήμα ΤΕΙ

Υλοποίηση - Κλάση Professor

```
1.  public class Professor {
2.     private String name;
3.     private int AFM;
4.     private Course lesson;
5.
6.     public Professor(String name, int AFM) {
7.        this.name = name;
8.        this.AFM = AFM;
9.     }
10.
11.    public void setLesson(Course c) {
12.       lesson = c;
13.    }
14.
15.    public String toString() {
16.       return name + " " + AFM + " " + lesson;
17.    }
18. }
```

Λογισμικό για τμήμα ΤΕΙ

Υλοποίηση - Κλάση Student

```java
public class Student {
   private String name;
   private int AM;
   private int units = 0;

   public Student(String name, int am) {
      this.name = name;
      AM = am;
   }

   public int getAM() { return AM; }

   public void addUnits(int units) { this.units += units; }

   public String toString() {
      return name + " AM:" + AM + " units:" + units;
   }
}
```

Λογισμικό για τμήμα ΤΕΙ

Υλοποίηση - Κλάση StudentRecord

```
1.  public class StudentRecord {
2.    private Student student;
3.    private double grade;
4.
5.    public StudentRecord(Student s) { student = s; }
6.
7.    public int setGrade(double grade) { this.grade = grade; }
8.
9.    public Student getStudent() { return student; }
10.
11.   public String toString() { return student + " :" + grade; }
12.
13.   public boolean passed() {
14.     if (grade >= 5) { return true; }
15.     return false;
16.   }
17. }
```

Λογισμικό για τμήμα ΤΕΙ

Υλοποίηση - Κλάση Course

```
1.  public class Course {
2.     private String name;
3.     private int code;
4.     private int units;
5.     private Professor professor;
6.     private ArrayList<StudentRecord> studentList = new ArrayList<StudentRecord>();
7.
8.     public Course(String name, int code, int units) {
9.        this.name = name; this.code = code; this.units = units;
10.    }
11.
12.    public void setProfessor(Professor p) {
13.       professor = p;
14.       p.setLesson(this);
15.    }
16.
17.    public void enroll(Student s) {
18.       studentList.add(new StudentRecord(s));
19.    }
```

χρησιμοποιούμε το this ως αναφορά στο παρόν αντικείμενο, ώστε να το προσθέσουμε στο αντικείμενο Professor

δημιουργία του αντικειμένου StudentRecord και ταυτόχρονη προσθήκη στη λίστα - λέγεται και «ανώνυμο αντικείμενο»

Λογισμικό για τμήμα ΤΕΙ

Υλοποίηση - Κλάση Course II

```
20.    public void assignGrades() {
21.       System.out.println("Give grades for course " + this.toString());
22.       Scanner input = new Scanner(System.in);
23.       for (StudentRecord record: studentList) {
24.          System.out.println("Give grade for student “ + record.getStudent().getAM() +":");
25.          double grade = input.nextDouble();
26.          record.setGrade(grade);
27.          if (record.passed()){
28.             record.getStudent().addUnits(units);
29.          }
30.       }
31.    }
32.
33.    public String toString() { return name + " " + code + " " + "(" + units + ")"); }
34.
35.    public void printInfo() {
36.       System.out.println("Course " + name + " " + code + " " + "(" + units + ")");
37.       for (StudentRecord record: studentList) { System.out.println(record); }
38.    }
```

διασχίζουμε τη λίστα των φοιτητών

αλυσιδωτές κλήσεις μεθόδων: είναι εφικτές εφόσον μια μέθοδος επιστρέφει αντικείμενο

Λογισμικό για τμήμα ΤΕΙ

Υλοποίηση - Κλάση Department


```
1.  public class Department {
2.     public static void main(String args[]) {
3.        int numOfStudents = Integer.parseInt(args[0]);
4.        Professor ekosmas = new Professor("Eleftherios Kosmas", 2012);
5.        Professor kvasilak = new Professor("Kostas Vasilakis", 2013);
6.        Course oop = new Course("oop", 212, 10);
7.        Course intro = new Course("intro", 101, 5);
8.        Student[] students = new Student[numOfStudents];
9.        Scanner input = new Scanner(System.in);
10.       for (int i = 0; i < numOfStudents; i ++){
11.          System.out.print("Give student name: ");
12.          String name = input.next();
13.          students[i] = new Student(name, i);
14.       }
15.       oop.setProfessor(ekosmas);
16.       oop.enroll(students[0]);oop.enroll(students[1]);oop.enroll(students[3]);
17.       intro.setProfessor(kvasilak);
18.       intro.enroll(students[2]); intro.enroll(students[3]);
19.       oop.assignGrades(); intro.assignGrades();
20.       System.out.println(ekosmas); System.out.println(kvasilak);
21.       oop.printInfo(); intro.printInfo();
22.    }
23. }
```

Σύνοψη

- UML διαγράμματα
- Σχέσεις κλάσεων
  - σύνθεση
  - συνάθροιση
- Ολοκληρωμένο παράδειγμα