Πανεπιστήμιο Δυτικής Αττικής
Τμήμα Μηχανικών Πληροφορικής και Υπολογιστών

Αντικειμενοστρεφής Προγραμματισμός (C++)

Ενότητα 7: Αντικειμενοστραφής σχεδίαση

Καθηγήτρια Κλειώ Σγουροπούλου

Ανάλυση, Σχεδίαση, Προγραμματισμός

- Ο **Αντικειμενοστρεφής Προγραμματισμός** (ΑΠ) επικεντρώνεται στις φυσικές οντότητες που εμπλέκονται σε μια εφαρμογή.
- Η **αντικειμενοστρεφής σχεδίαση** είναι μια μέθοδος που χρησιμοποιεί την αποσύνθεση (decomposition) και την αφαίρεση (abstraction) για την υλοποίηση ενός μοντέλου του συστήματος που θέλουμε να αναπτύξουμε.
- **Αντικειμενοστρεφής αποσύνθεση.**
  - Χρήση κλάσεων και αντικειμένων για τη λογική δόμηση ενός συστήματος.
- **Αντικειμενοστρεφής ανάλυση.**
  - Μελέτη των απαιτήσεων της εφαρμογής όσον αφορά ορισμό κλάσεων και αντικειμένων.

Το Μοντέλο Αντικειμένων

- Ορισμός,
  - Προγραμματιστική προσέγγιση που χρησιμοποιεί της αρχές της **αφαίρεσης,** της **ενθυλάκωσης,** της **ιεραρχίας** (κληρονομικότητας), της **δόμησης μονάδων** (modularity).
- Πολύ αποδοτικό στο σχεδιασμό πολύπλοκων εφαρμογών.
  - Μεγαλύτερες απαιτήσεις από το λογισμικό και απλούστερη προσέγγιση για τους προγραμματιστές,
  - Μεγαλύτερη παραγωγικότητα,
  - Μεγαλύτερος βαθμός συντηρισιμότητας.

Αφαίρεση (Abstraction)

- Απλοποιημένη όψη του αντικειμένου.
  - Μοντελοποίηση μόνο των απαραίτητων χαρακτηριστικών.
- Οι επιλογές εξαρτώνται από την εφαρμογή και το περιβάλλον.
- Παραδείγματα:
  - Γλώσσα προγραμματισμού,
    - Προστατεύει τον προγραμματιστή από πολυπλοκότητες του υλικού.
  - Μεταβλητές, διαπροσωπεία κλάσης.

Τύποι Αφαίρεσης

- Δύο βασικοί τύποι αφαίρεσης:
  - Επίπεδο ελέγχου (ενέργεια).
    - Λειτουργίες (**insert, move, resize, find,** …)
- Επίπεδο οντότητας.
    - Εύκολα αναγνωρίσιμα – με αντιστοιχία στον πραγματικό κόσμο αντικείμενα (**Window, Display, Cursor, Screen,** …

Employee, Student, Course, …)

Ενθυλάκωση

- Ενθυλάκωση μεταβλητών και συναρτήσεων στο εσωτερικό μιας κλάσης με μια καλά-ορισμένη διαπροσωπεία.
- Επιτυγχάνεται με την απόκρυψη πληροφορίας από το εξωτερικό περιβάλλον της κλάσης.
  - Απόκρυψη λεπτομερειών υλοποίησης,
  - Δημόσια και καλά-ορισμένη διαπροσωπεία με το περιβάλλον.
- Οδηγεί στην τυποποίηση διαπροσωπειών.
  - Μεγάλη βελτίωση στην ανάπτυξη λογισμικού.

Δόμηση σε μονάδες

- Μια από τις παλαιότερες και καλύτερες προγραμματιστικές έννοιες
  - Μονάδες προγραμμάτων που αποκρύπτουν και διαμοιράζονται πληροφορίες,
  - Υπορουτίνες – κοινές διαδικασίες που χρησιμοποιούνται σε διάφορα μέρη της εφαρμογής.

Ιεραρχία (κληρονομικότητα)

- Σχέση γονέα-παιδιού μεταξύ κλάσεων,
- Παράγωγες κλάσεις.
  - Κληρονομούν δεδομένα και λειτουργίες από τις βασικές κλάσεις,
  - Ορίζουν μόνο τα επιπρόσθετα χαρακτηριστικά.

Βασικές αρχές ΑΠ

- Δύσκολη η αποδοτική σχεδίαση πολύπλοκων εφαρμογών. Δεν είναι δυνατή η απευθείας κωδικοποίηση
- Απαραίτητη η οργάνωση της διαδικασίας σχεδίασης για μεγαλύτερη παραγωγικότητα και συντηρισιμότητα
- Η αντικειμενοστρεφής προσέγγιση έχει αναγνωριστεί ως ένας από τους αποδοτικότερους τρόπους για τη σχεδίαση πολύπλοκων εφαρμογών

Στόχοι της διαδικασίας σχεδίασης 1/2

- Απλότητα,
  - Καλά σχεδιασμένα, εύστοχα αντικείμενα,
  - Διαπροσωπείες φιλικές προς το χρήστη και το εξωτερικό περιβάλλον.
- Ευελιξία,
  - Εύκολη επέμβαση για αλλαγές.
- Επεκτασιμότητα,
  - Εύκολη υλοποίηση επεκτάσεων χωρίς καταστροφή του αρχικού σχεδιασμού.
- Μεταφερσιμότητα.
  - Ανεξαρτησία από το υλικό.

Στόχοι της διαδικασίας σχεδίασης 2/2

- Επαναχρησιμοποίηση
  - Καλά σχεδιασμένο λογισμικό ώστε να χρησιμοποιείται σε μεγάλο αριθμό εφαρμογών

Σχεδιάζοντας ΑΠ προγράμματα

- Ένα ΑΠ πρόγραμμα οφείλει να είναι ένα πιστό μοντέλο της πραγματικότητας,
  - Είναι σημαντικός ο εντοπισμός των βασικών εννοιών σε μια εφαρμογή,
  - Δημιουργία των σχετικών κλάσεων,
  - Επαναληπτική διαδικασία,
  - Παραδοσιακή προσέγγιση.
- «Από πάνω προς τα κάτω» διαδικασιακή σχεδίαση.
  - Ιεραρχικές εξαρτήσεις,
  - Αλλαγές δύσκολο να υλοποιηθούν.

Συστατικά (Components)

- Component = ομάδα σχετικών κλάσεων που συνεργάζονται,
  - Καλά ορισμένες διαπροσωπείες.
- Βήματα σχεδίασης.
  - Εντόπισε τις κλάσεις,
  - Καθόρισε τις λειτουργίες των κλάσεων,
  - Καθόρισε τις εξαρτήσεις των κλάσεων,
  - Καθόρισε τις διαπροσωπείες των κλάσεων.

Εντοπισμός κλάσεων

- Το πιο προφανές βήμα στη σχεδίαση,
  - Αν η κλάση δεν είναι προφανής, δεν κάνει για την εφαρμογή!
- Αποκάλυψη της εγγενούς δομής της εφαρμογής,
- Πολύ σημαντική η καλή κατανόηση της εφαρμογής.

Καθορισμός λειτουργιών κλάσεων

- Σύνολο βασικών λειτουργιών που ορίζει η γλώσσα για τη δημιουργία μιας κλάσης
  - Κατασκευαστές, καταστροφείς, κ.λπ.
- Ένα πιο δύσκολο έργο
  - Πρόβλεψη λειτουργιών αλληλεπίδρασης μεταξύ κλάσεων
- Πλεονέκτημα C++:
  - Αν το αρχικό σχέδιο δεν είναι επαρκές μπορεί εύκολα να αναθεωρηθεί

Εξαρτήσεις κλάσεων 1/2

- Εξάρτηση = σχέση μεταξύ διαφορετικών κλάσεων
- Τρεις τύποι εξαρτήσεων
  - Κληρονομικότητα (Inheritance)
    - Παράγωγες κλάσεις
    - Παράδειγμα: **GraduateStudent** ειδικός τύπος του αντικειμένου **Student**
  - Σύνθεση (Composition)
    - Ένα αντικείμενο περιέχει αντίγραφο ενός άλλου αντικειμένου
    - Παράδειγμα: Το αντικείμενο **Student** περιέχει το αντικείμενο **Address**

Εξαρτήσεις κλάσεων 2/2

- Σύνδεση (Link)
  - Ανεξάρτητα αντικείμενα που επικοινωνούν μεταξύ τους
  - Παράδειγμα: Το αντικείμενο **Student** περνάει μηνύματα στο αντικείμενο **CourseCatalog**

Διαπροσωπείες κλάσεων

- Τα πρωτότυπα συναρτήσεων για τις συναρτήσεις-μέλη μιας κλάσης καθορίζουν τη διαπροσωπεία της,
- Οι διαπροσωπείες διαμοιράζονται με άλλους προγραμματιστές,
- Αποδοτική σχεδίαση.

ΠΑΡΑΔΕΙΓΜΑ

Διαχείριση Ιατρικών Επισκέψεων

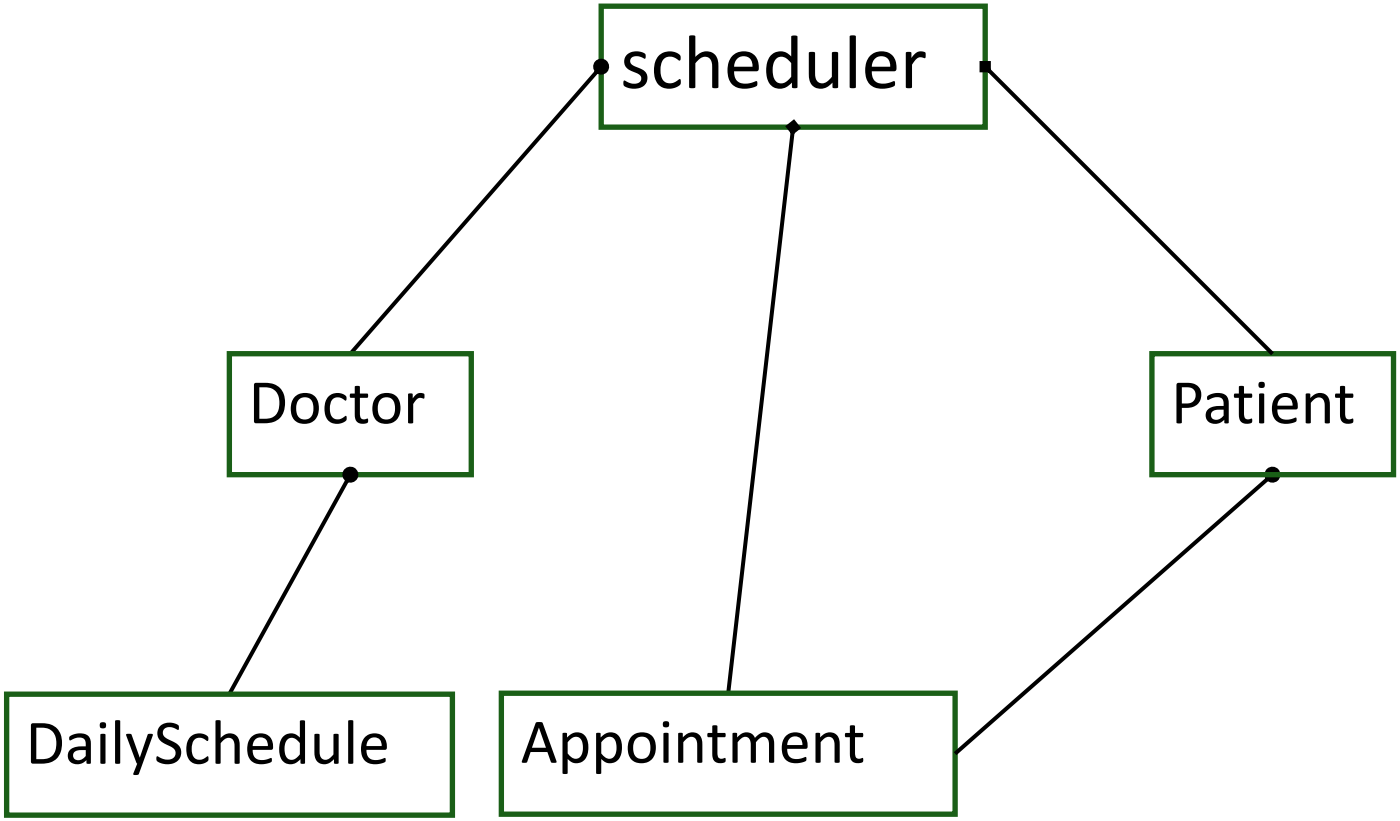
1ο βήμα: Προδιαγραφές

- Αυτόματος προγραμματισμός επισκέψεων των ασθενών,
- Διαχείριση πολλών ασθενών και γιατρών,
- 15λεπτες επισκέψεις μεταξύ 8:00 π.μ. και 6:00 μ.μ.,
- Εκτύπωση προσωπικού προγράμματος για κάθε γιατρό,
- Διαδραστικό πρόγραμμα με έξοδο σε οθόνη και αρχεία.

2ο βήμα: Ανάλυση

- Εντοπισμός κλάσεων. Αντικείμενα της εφαρμογής:
  - **Γιατρός (Doctor),**
  - **Ασθενής (Patient),**
  - **Ημερήσιο Πρόγραμμα (DailySchedule),**
  - **Επίσκεψη (Appointment),**
  - **Χρονοπρογραμματιστής (Scheduler).**
- Τυπικό σενάριο για τη διαδικασία χρονοπρογραμματισμού επισκέψεων:
- Ένας **Ασθενής** δίνει το όνομά του στο **Χρονοπρογραμματιστή,**
- Ο **Ασθενής** επιλέγει **Γιατρό,**
- Ο **Χρονοπρογραμματιστής** προσθέτει την **Επίσκεψη** στο **Πρόγραμμα** του **Γιατρού** και του **Ασθενούς,**
- Ο **Χρονοπρογραμματιστής** επιβεβαιώνει την **Επίσκεψη.**

Εξαρτήσεις κλάσεων



Notation:

Composite:

Link:

Λειτουργίες 1/2

- **Doctor** πρέπει να υποστηρίζει τις λειτουργίες:
  - **AddToSchedule**
  - **ShowAppointments**
- **Patient** πρέπει να υποστηρίζει τις λειτουργίες :
  - **InputName**
  - **ChooseDoctor**
  - **ChooseTimeSlot**
  - **SetAppointment**
- **DailySchedule** πρέπει να υποστηρίζει τις λειτουργίες :
  - **SetAppointment**
  - **IsTimeSlotFree**
  - **ShowAppointments**

Λειτουργίες 2/2

- **Appointment** πρέπει να υποστηρίζει τις λειτουργίες:
  - **Constructor:** Κατασκεύασε ένα αντικείμενο επίσκεψης από χρονικό διάστημα, κωδικό γιατρού, όνομα ασθενούς
  - **IsScheduled**
- **Scheduler** πρέπει να υποστηρίζει τις λειτουργίες :
  - **PrintAllAppointments:** Τύπωσε μορφοποιημένη έξοδο όλων των υπαρχουσών επισκέψεων
  - **ScheduleOneAppointment:** Πρόσθεσε μια επίσκεψη στο πρόγραμμα. Αλληλεπίδραση με το χρήστη
  - **ScheduleAllAppointments:** Επαναληπτική διαδικασία ερώτησης χρήστη για πρόσθεση επισκέψεων

Βοηθητικές κλάσεις

- **Time slot**

```
class TimeSlot {
public:
TimeSlot( const unsigned n = 0 );
unsigned AsInteger() const; // Returns the integer value of the
time.
friend istream & operator >>(istream & inp, TimeSlot & T);
friend ostream & operator <<(ostream & os, const TimeSlot & T);
// ------ Static member functions -------

static unsigned GetStartHour();
static unsigned GetApptLen();
static void SetStartHour( unsigned n );
static void SetApptLen( unsigned n );
private:
static unsigned StartHour;
static unsigned ApptLen;
unsigned intValue;
void SetIntValue( unsigned n );
};
```

3ο βήμα: Σχεδίαση

- Μετά την αναγνώριση όλων των κλάσεων, καθορισμός των διαπροσωπειών τους,
- Αποτέλεσμα της φάσης σχεδίασης είναι το header file με όλες τις κλάσεις και τις διαπροσωπείες τους.
- Κλάσεις:
  - **TimeSlot,**
  - **Doctor,**
  - **Patient,**
  - **DailySchedule,**
  - **Appointment,**
  - **Scheduler.**

4ο βήμα: Κυρίως πρόγραμμα

- Το κυρίως πρόγραμμα δημιουργεί ένα στιγμιότυπο του Scheduler

```
#include "doctors.h"
#include "schedlr.h"

static Doctor doctorArray[NumDoctors];
int main()
{
cout << "Doctors Office Scheduling
Program\n\n";
Scheduler officeSchedule( doctorArray );
officeSchedule.ScheduleAllAppointments();
officeSchedule.PrintAllAppointments("appts.
txt");
return 0;
}
```

5ο βήμα: Κώδικας

Σειρά σας ☺…

ΤΕΛΟΣ ΕΝΟΤΗΤΑΣ

Σημειώματα

Σημείωμα Αναφοράς

Σημείωμα Αδειοδότησης

Επεξήγηση όρων χρήσης έργων τρίτων

©	Δεν επιτρέπεται η επαναχρησιμοποίηση του έργου, παρά μόνο εάν ζητηθεί εκ νέου άδεια από το δημιουργό.
διαθέσιμο με άδεια CC-BY	Επιτρέπεται η επαναχρησιμοποίηση του έργου και η δημιουργία παραγώγων αυτού με απλή αναφορά του δημιουργού.
διαθέσιμο με άδεια CC-BY-SA	Επιτρέπεται η επαναχρησιμοποίηση του έργου με αναφορά του δημιουργού, και διάθεση του έργου ή του παράγωγου αυτού με την ίδια άδεια.
διαθέσιμο με άδεια CC-BY-ND	Επιτρέπεται η επαναχρησιμοποίηση του έργου με αναφορά του δημιουργού. Δεν επιτρέπεται η δημιουργία παραγώγων του έργου.
διαθέσιμο με άδεια CC-BY-NC	Επιτρέπεται η επαναχρησιμοποίηση του έργου με αναφορά του δημιουργού. Δεν επιτρέπεται η εμπορική χρήση του έργου.
διαθέσιμο με άδεια CC-BY-NC-SA	Επιτρέπεται η επαναχρησιμοποίηση του έργου με αναφορά του δημιουργού και διάθεση του έργου ή του παράγωγου αυτού με την ίδια άδεια. Δεν επιτρέπεται η εμπορική χρήση του έργου.
διαθέσιμο με άδεια CC-BY-NC-ND	Επιτρέπεται η επαναχρησιμοποίηση του έργου με αναφορά του δημιουργού. Δεν επιτρέπεται η εμπορική χρήση του έργου και η δημιουργία παραγώγων του.
διαθέσιμο με άδεια CC0 Public Domain	Επιτρέπεται η επαναχρησιμοποίηση του έργου, η δημιουργία παραγώγων αυτού και η εμπορική του χρήση, χωρίς αναφορά του δημιουργού.
διαθέσιμο ως κοινό κτήμα	Επιτρέπεται η επαναχρησιμοποίηση του έργου, η δημιουργία παραγώγων αυτού και η εμπορική του χρήση, χωρίς αναφορά του δημιουργού.
χωρίς σήμανση	Συνήθως δεν επιτρέπεται η επαναχρησιμοποίηση του έργου.

Διατήρηση Σημειωμάτων

Οποιαδήποτε αναπαραγωγή ή διασκευή του υλικού θα πρέπει να συμπεριλαμβάνει:

- το Σημείωμα Αναφοράς
- το Σημείωμα Αδειοδότησης
- τη δήλωση Διατήρησης Σημειωμάτων
- το Σημείωμα Χρήσης Έργων Τρίτων (εφόσον υπάρχει)

μαζί με τους συνοδευόμενους υπερσυνδέσμους.