Ελληνική Δημοκρατία
Τεχνολογικό Εκπαιδευτικό
Ίδρυμα Ηπείρου

Αντικειμενοστραφής Προγραμματισμός

Ενότητα 6 : **Προχωρημένα θέματα (1/2)**

Ιωάννης Τσούλος

Ευρωπαϊκή Ένωση
Ευρωπαϊκό Κοινωνικό Ταμείο

ΕΠΙΧΕΙΡΗΣΙΑΚΟ ΠΡΟΓΡΑΜΜΑ
ΕΚΠΑΙΔΕΥΣΗ ΚΑΙ ΔΙΑ ΒΙΟΥ ΜΑΘΗΣΗ
επένδυση στην κοινωνία της γνώσης
ΥΠΟΥΡΓΕΙΟ ΠΑΙΔΕΙΑΣ & ΘΡΗΣΚΕΥΜΑΤΩΝ, ΠΟΛΙΤΙΣΜΟΥ & ΑΘΛΗΤΙΣΜΟΥ
ΕΙΔΙΚΗ ΥΠΗΡΕΣΙΑ ΔΙΑΧΕΙΡΙΣΗΣ
Με τη συγχρηματοδότηση της Ελλάδας και της Ευρωπαϊκής Ένωσης

ΕΣΠΑ 2007-2013
πρόγραμμα για την ανάπτυξη
ΕΥΡΩΠΑΪΚΟ ΚΟΙΝΩΝΙΚΟ ΤΑΜΕΙΟ

Τμήμα Μηχανικών Πληροφορικής Τ.Ε

Αντικειμενοστραφής Προγραμματισμός

Ενότητα 6 : Προχωρημένα θέματα (1/2)

Ιωάννης Τσούλος
Επίκουρος Καθηγητής

Άρτα, 2015

Άδειες Χρήσης

- Το παρόν εκπαιδευτικό υλικό υπόκειται σε άδειες χρήσης Creative Commons.
- Για εκπαιδευτικό υλικό, όπως εικόνες, που υπόκειται σε άλλου τύπου άδειας χρήσης, η άδεια χρήσης αναφέρεται ρητώς.

Χρηματοδότηση

- Το έργο υλοποιείται στο πλαίσιο του Επιχειρησιακού Προγράμματος «**Εκπαίδευση και Δια Βίου Μάθηση**» και συγχρηματοδοτείται από την Ευρωπαϊκή Ένωση (Ευρωπαϊκό Κοινωνικό Ταμείο) και από εθνικούς πόρους.
- Το έργο «**Ανοικτά Ακαδημαϊκά Μαθήματα στο ΤΕΙ Ηπείρου**» έχει χρηματοδοτήσει μόνο τη αναδιαμόρφωση του εκπαιδευτικού υλικού.
- Το παρόν εκπαιδευτικό υλικό έχει αναπτυχθεί στα πλαίσια του εκπαιδευτικού έργου του διδάσκοντα.

Ευρωπαϊκή Ένωση
Ευρωπαϊκό Κοινωνικό Ταμείο
ΕΠΙΧΕΙΡΗΣΙΑΚΟ ΠΡΟΓΡΑΜΜΑ
ΕΚΠΑΙΔΕΥΣΗ ΚΑΙ ΔΙΑ ΒΙΟΥ ΜΑΘΗΣΗ
επένδυση στην κοινωνία της γνώσης
ΥΠΟΥΡΓΕΙΟ ΠΑΙΔΕΙΑΣ & ΘΡΗΣΚΕΥΜΑΤΩΝ, ΠΟΛΙΤΙΣΜΟΥ & ΑΘΛΗΤΙΣΜΟΥ
ΕΙΔΙΚΗ ΥΠΗΡΕΣΙΑ ΔΙΑΧΕΙΡΙΣΗΣ
ΕΣΠΑ 2007-2013
πρόγραμμα για την ανάπτυξη
ΕΥΡΩΠΑΪΚΟ ΚΟΙΝΩΝΙΚΟ ΤΑΜΕΙΟ
Με τη συγχρηματοδότηση της Ελλάδας και της Ευρωπαϊκής Ένωσης

Encapsulation

- Ας ορίσουμε μια κλάση `Person` η οποία θα περιέχει ορισμένες πληροφορίες για ένα άτομο, όπως π.χ. το όνομά του, την ηλικία του, το τηλέφωνό του και τη διεύθυνση email του.
- Μια τέτοια κλάση μπορεί να χρησιμοποιηθεί π.χ. σε ένα πρόγραμμα ατζέντας ή ακόμη και ως βάση σε πρόγραμμα πελατολογίου, ασθενών, κλπ.

Encapsulation

```
class Person
{
public:
    // οι μεταβλητές της κλάσης
    string Firstname_, Lastname_;
    int Age_;
    string Telephone_;
    string Email_;
    // ο constructor
    Person(string fname, string lname, int age, string tel, string email)
    {
        Firstname_ = fname;
        Lastname_ = lname;
        Age_ = age;
        Telephone_ = tel;
        Email_ = email;
    }
}
```

*<u>Παρατήρηση:</u> Τα ονόματα των μεταβλητών της κλάσης λήγουν σε "_". Κάτι τέτοιο **δεν** είναι αναγκαίο, είναι όμως μια συνηθισμένη πρακτική και βοηθάει στην αναγνώριση και διαχωρισμό των απλών μεταβλητών από τις μεταβλητές μέλη μιας κλάσης.*

Encapsulation

- Αφού ορίσαμε την κλάση, μπορούμε να προχωρήσουμε στην δημιουργία κάποιων αντικειμένων της κλάσης:

```
Person bilbo("Bilbo", "Baggins", 111, "+306970123456",
        "bilbobaggins@theshire.net");
```

- Με αυτόν τον τρόπο, δημιουργήσαμε το αντικείμενο bilbo που αντιστοιχεί στο άτομο Bilbo Baggins, 111 ετών με τηλ. 306970123456 και email bilbobaggins@theshire.net.
- Όπως είναι οι πληροφορίες που περιγράφουν το άτομο, είναι προσβάσιμες **σε όλους**.
- Αυτό σημαίνει ότι αν με κάποιον τρόπο αποκτήσουμε πρόσβαση στο αντικείμενο bilbo, **θα μπορούμε** να αλλάξουμε οποιαδήποτε πληροφορία θελήσουμε και με οποιονδήποτε τρόπο.

Encapsulation

- Και μάλιστα με πολύ απλό τρόπο:

```
bilbo.Firstname_ = “μπίλμπο”;
bilbo.Lastname_ = “μπαγκινσόπουλος”;
bilbo.Age_ = 3;
bilbo.Email_ = “this is definitely not a valid email address”;
bilbo.Telephone_ = “yeah, try to call this”;
```

Πρόκειται για τρανή παραβίαση των προσωπικών δεδομένων!!!

- **Πώς μπορούμε να αποφύγουμε τέτοιου είδους μη προβλεπόμενη μετατροπή των δεδομένων ενός αντικειμένου;**

Encapsulation

- Η C++ έχει σχεδιαστεί γύρω από το μοντέλο του αντικειμενοστραφούς προγραμματισμού, προβλέπει τον περιορισμό της πρόσβασης των δεδομένων σε επίπεδα.
- Ένα από τα επίπεδα πρόσβασης είναι η πρόσβαση χωρίς περιορισμό σε όλους, που ορίζεται με τη λέξη **`public`** → ορίζει μια περιοχή δηλώσεως μεθόδων ή μεταβλητών (γενικότερα μέλη της κλάσης).
- Κάθε μέλος που βρίσκεται στην περιοχή `public` θα είναι προσβάσιμο από οπουδήποτε μέσα στην ίδια κλάση ή εκτός της κλάσης.
- Το αντίστροφο, δηλαδή ο περιορισμός της πρόσβασης γίνεται με τη χρήση της λέξης **`private`**. Η `private` περιορίζει την πρόσβαση των μεταβλητών ή των μεθόδων που βρίσκονται στην αντίστοιχη περιοχή, μόνο στην συγκεκριμένη κλάση (και φυσικά σε αντικείμενα αυτής).

Encapsulation

- Η κλάση Person, με τη χρήση της private, θα μετασχηματιστεί ως εξής:

```
class Person
{
private:
// οι μεταβλητές της κλάσης
     private string Firstname_, Lastname_;
     private int Age_;
     private string Telephone_;
     private string Email_;
public:
...
```

- Αυτό όμως σημαίνει ότι δεν θα είναι πλέον δυνατή η πρόσβαση σε οποιαδήποτε πληροφορία του ατόμου ακόμη και για απλή ανάγνωση!
- Κάτι τέτοιο **δεν** είναι επιθυμητό και πρέπει να βρεθεί τρόπος να επιτραπεί έστω και ελεγχόμενη πρόσβαση στα δεδομένα.

Encapsulation

- Αυτό επιτυγχάνουμε με την χρήση των μεθόδων της κλάσης.
- Ελεγχόμενη πρόσβαση για ανάγνωση αλλά και μετατροπή των δεδομένων.
- Συνήθως και για τις περισσότερες περιπτώσεις κλάσεων, αρκεί ο ορισμός ενός ζεύγους μεθόδων για κάθε μεταβλητή της κλάσης, μία για ανάγνωση και μια για μετατροπή της τιμής της μεταβλητής (ένα ζεύγος getter/setter όπως λέγονται συχνά).

Encapsulation

Για την κλάση Person παραθέτουμε πιθανές μεθόδους get/set για ορισμένες από τις μεταβλητές (Age_ και Email_)

```
// Return the age
int Person::getAge()
{
return Age_;
}
// return the Email address
string Person::getEmail()
{
return Email_;
}
// method to set the age of the person
bool Person::setAge(int Age)
{
// check if given Age is non-negative (> 0)
if (Age > 0)
{
Age_ = Age;
return true;
} else
return false;
}
// method to set the email address
bool Person::setEmail(string Email)
{
// call a helper method to check the validity of the
email
// address (if it's in the form x@y.z).
// Ideally, the email address should be a class on its
own.
if (isValid(Email) == true)
{
Email_ = Email;
return true;
} else
return false;
}
```

Encapsulation

- Βλέπουμε πώς **ελέγχεται πλέον** η πρόσβαση στις μεταβλητές.
- Η μεταβλητή που κρατά τη διεύθυνση email του ατόμου, για παράδειγμα, αλλάζει **μόνο** αν έχουμε δώσει μια έγκυρη διεύθυνση `email` (της μορφής `x@y.z`).
- Με τον ίδιο τρόπο που περιορίζουμε την πρόσβαση σε μεταβλητές μπορούμε να **περιορίσουμε** την πρόσβαση και σε μεθόδους.
- Θα μπορούσαμε π.χ. να έχουμε μια μέθοδο που να ελέγχει αν ο αριθμός τηλεφώνου του ατόμου είναι έγκυρος, πραγματοποιώντας αναζήτηση σε κάποια βάση δεδομένων.
- Μια τέτοια μέθοδος **δεν θα θέλαμε** να είναι προσβάσιμη από οποιονδήποτε εκτός της κλάσης, παρά **μόνο** σε άλλες μεθόδους της ίδιας της κλάσης (π.χ. στην μέθοδο `setTelephone()`).

Inheritance

- Στην κληρονομικότητα μια κλάση κληρονομεί τα χαρακτηριστικά μιας υπάρχουσας κλάσης και προσθέτει καινούρια ή τροποποιεί τα ήδη υπάρχοντα.
- Η κλάση `Person` ορίζει χαρακτηριστικά που περιγράφουν ένα άτομο αλλά δεν προβλέπει επιπλέον πληροφορίες (φύλο, δουλειά, διεύθυνση εργασίας κλπ).
- **Πρόβλημα:** Δεν μπορούμε να προβλέψουμε όλες τις πιθανές πληροφορίες και να τις εισάγουμε στην κλάση Person γιατί οι περισσότερες θα έμεναν αναπάντητες και κάτι τέτοιο θα οδηγούσε σε σπατάλη χώρου (αφού θα έπρεπε να καταχωρήσουμε όλες τις πληροφορίες που θα ήταν κενές).

Inheritance

- Χρήσιμο να έχουμε μια κοινή βάση και να κρατάμε επιπλέον πληροφορίες μόνο όταν τις χρειαζόμαστε.
- Έστω ότι η κοινή βάση είναι η κλάση Person και θέλουμε να μελετήσουμε τις περιπτώσεις να είναι κάποιος υπάλληλος (Clerk) ή δάσκαλος (Teacher).
- Και οι δύο κατηγορίες ατόμων μοιράζονται κοινά χαρακτηριστικά που θεωρούμε ότι περιέχονται στην κλάση Person.
- Μπορούμε δηλαδή να ορίσουμε δύο νέες κλάσεις που θα κληρονομούν την κλάση Person.

Inheritance

- Η δήλωση της κληρονομικότητας μιας κλάσης γίνεται ως εξής (*ορίζουμε ταυτόχρονα και την πρόσβαση στις μεταβλητές και τον δημιουργό της κλάσης*)

```
class Clerk : public Person
{
private:
	string JobTitle_;
	string CompanyName_;
	string JobAddress_;
	string JobEmail_;
	string JobTel_;
	string JobFax_;
	string JobDescription_;
public:
	Clerk(string fname, string lname, int age, string tel,
		string email, string jobtitle, string companyname,
		string jobaddress, string jobemail,
		string jobtel, string jobfax,
		string jobdescription)
{
		Firstname_ = fname;
		Lastname_ = lname;
		Age_ = age;
		Telephone_ = tel;
		Email_ = email;
		JobTitle_ = jobtitle;
		CompanyName_ = companyname;
		JobAddress_ = jobaddress;
		JobEmail_ = jobemail;
		JobTel_ = jobtel;
		JobFax = jobfax;
		JobDescription_ = jobdescription;
}
// ακολουθούν οι μέθοδοι get/set για κάθε μεταβλητή με τους απαραίτητους ελέγχους...
// η ακόλουθη μέθοδος δίνει συνοπτικές πληροφορίες για τον υπάλληλο.
	string getInfo() {
	return (getFirstname()+" "+getLastname()
		+" works at "+CompanyName_
		+", at "+JobAddress_
		+".\n Email: "+getEmail()+"\n"
		+"Tel: "+JobTel_);
}
}
```

Inheritance

- Αντίστοιχα, ορίζουμε την κλάση Teacher:

```
class Teacher : public Person
{
private:
	string Title_;
	string School_;
	string SchoolAddress_;
	string SchoolTel_;
	string CourseName_;
	string CourseDescription_;
public:
	Teacher(string fname, string lname, int age, string tel,
		string email, string title, string school,
		string schooladdress, string schooltel,
	{
	Firstname_ = fname;
	Lastname_ = lname;
	Age_ = age;
	Telephone_ = tel;
	Email_ = email;
	Title_ = title;
	School_ = school;
	SchoolAddress_ = schooladdress;
	SchoolTel_ = jobtel;
	CourseName_ = coursename;
	CourseDescription_ = coursedescription;
}
// ακολουθούν οι μέθοδοι get/set για κάθε μεταβλητή με τους απαραίτητους ελέγχους...
...
// Η ακόλουθη μέθοδος δίνει συνοπτικές πληροφορίες για τον
// καθηγητή.
string getInfo() {
	return (getFirstname()+" "+getLastname()
		+" teaches "+CourseName_+" at "+School_
		+", "+SchoolAddress_+".\n"
		+"Email: "+getEmail()+"\n"
		+"Tel: "+SchoolTel_);
}
}
```

Inheritance

- **Παρατήρηση**
  - Χρησιμοποιήσαμε τις μεθόδους get() της κλάσης Person για την πρόσβαση στις μεταβλητές της (εφόσον είναι δηλωμένες private).
- **Χρησιμότητα**

```
Person bilbo("Bilbo", "Baggins", 111, "+306970123456", "bilbobaggins@theshire.net");

Clerk sam( "Samwise", "Gamgee", 33, "+30697654321", "samgamgee@theshire.net",
"Gardener", "Baggins Inc.", "Bag End, Hobbiton, The Shire", "gardener@baggins.com",
"+302103456789", "+302103456780", "Garden Dept. Director");

Teacher pippin( "Peregrin", "Took", 27, "+30690090090", "pippin@theshire.net", "Dr.",
"King's College", "Hobbiton", "+30210000001", "Philosophy", "Deal with the important
matters of life, eg. what do we eat?");
```

Inheritance

- Μπορούμε να χρησιμοποιήσουμε για κάθε ένα από αυτά τα αντικείμενα, πέρα από τις μεθόδους της κλάσης στην οποία ανήκει, και τις μεθόδους της γονικής κλάσης:

```
cout << “bilbo has email address: ” << bilbo.getEmail()) << endl;
```

αυτή η εντολή θα τυπώσει:

```
bilbo has email address: bilbobaggins@shire.net
```

- Ενώ η εντολή:

```
cout << “sam works as a “ << sam.getJobTitle() << “ at “
<< sam.getCompanyName()) << endl;
```

θα τυπώσει:

```
sam works as a Gardener at Baggins Inc.
```

Inheritance

- Παράλληλα, η εντολή:

```
cout << “pippin teaches “ << pippin.getCourseName() << “ at “
<< pippin.getSchool()) << endl;
```

θα τυπώσει:

```
pippin teaches Philosophy at King's College
```

- Τέλος, οι εντολές:

```
cout << “sam's private telephone is “ << sam.getTel() << endl;
cout << “pippin is “ << pippin.getAge() << “ years old” << endl;
```

- θα τυπώσουν:

```
sam's private telephone is +30697654321
pippin is 27 years old
```

Inheritance

- Καλέσαμε μεθόδους της κλάσης Person και από τα τρία αντικείμενα→ Κληρονομικότητα κλάσεων!
- Επαναχρησιμοποίηση κώδικα με απλή επέκταση (*code reusability*).
- Έχοντας μερικές κλάσεις με ορισμένα μόνο τα βασικά χαρακτηριστικά, μπορούμε αναλόγως το πρόγραμμα που πρέπει να υλοποιήσουμε να προσθέσουμε ή να τροποποιήσουμε χαρακτηριστικά κατά βούληση, ώστε να επιτύχουμε το επιθυμητό αποτέλεσμα.

Βιβλιογραφία

1. Εγχειρίδιο της C++, 2η Ελληνική έκδοση, Jesse Liberty, Γκιούρδας.
2. Μάθετε τη C++, 2η Ελληνική έκδοση, Jesse Liberty , Γκιούρδας.
3. Προγραμματισμός με τη γλώσσα C++ Μέρος Α, Αλεβίζος Θ., Έκδοση ΤΕΙ Καβάλας
4. C++ Αντικειμενοστραφής Προγραμματισμός Υπολογιστών Τομαράς Α., , Εκδόσεις Νέων Τεχνολογιών.
5. Ανακαλύψτε τη γλώσσα C, J. Purdum, Εκδόσεις Δίαυλος.
6. Εισαγωγή στο Συστηματικό Προγραμματισμό και στη γλώσσα C++, Σ. Μπαλτζής, εκδόσεις πανεπιστημίου Ιωαννίνων.
7. C++ From the beginning, Jan Skansholm, Addison Wesley.
8. The design and analysis of computer algorithms, A.V. AHO, J.E. HOPCROFT, J.D. ULLMANN, Addison Wesley 1974.
9. Structure and Interpretation of Computer Programs, H. ABELSON, G.J. SUSSMAN, J. SUSSMAN, MIT Press, Mc Graw Hill Book Company, 1985
10. The art of computer programming, D.E. KNUTH, Addison-Wesley.

Σημείωμα Αναφοράς

Σημείωμα Αδειοδότησης

[1] http://creativecommons.org/licenses/by-nc-nd/4.0/deed.el

Τέλος Ενότητας

Επεξεργασία: Ευάγγελος Καρβούνης

Άρτα, 2015

Ευρωπαϊκή Ένωση
Ευρωπαϊκό Κοινωνικό Ταμείο
ΕΠΙΧΕΙΡΗΣΙΑΚΟ ΠΡΟΓΡΑΜΜΑ
ΕΚΠΑΙΔΕΥΣΗ ΚΑΙ ΔΙΑ ΒΙΟΥ ΜΑΘΗΣΗ
επένδυση στην κοινωνία της γνώσης
ΥΠΟΥΡΓΕΙΟ ΠΑΙΔΕΙΑΣ & ΘΡΗΣΚΕΥΜΑΤΩΝ, ΠΟΛΙΤΙΣΜΟΥ & ΑΘΛΗΤΙΣΜΟΥ
ΕΙΔΙΚΗ ΥΠΗΡΕΣΙΑ ΔΙΑΧΕΙΡΙΣΗΣ
ΕΣΠΑ 2007-2013
ΕΥΡΩΠΑΪΚΟ ΚΟΙΝΩΝΙΚΟ ΤΑΜΕΙΟ
Με τη συγχρηματοδότηση της Ελλάδας και της Ευρωπαϊκής Ένωσης

Τέλος Ενότητας

Προχωρημένα θέματα (1/2)