Αντικειμενοστραφής Προγραμματισμος Εργαστήριο 5

Γεώργιος Τσουμάνης, Άγγελος Δήμητσας, Βασίλειος Νάστος

January 11, 2023

1 STL(Standard Template Library)

- Η STL είναι μια βιβλιοθήκη επαναχρησιμοποιήσιμων στοιχείων που βασίζεται στο γενερικό προγραμματισμό (προγραμματισμό με templates)
- Η STL αποτελείται από
  - Containers (περιέκτες): Επιτρέπουν την οργάνωση μιας συλλογής αντικειμένων στη μνήμη του Η/Υ. Πρόκειται για templated κλάσεις (π.χ. vector¡int¿, list¡double¿, ...).
  - Algorithms (αλγόριθμοι): Αλγόριθμοι που εφαρμόζονται σε containers (π.χ. sort, find, ...). Είναι γενικοί και μπορούν να χρησιμοποιηθούν σε διάφορους τύπους containers.
  - Iterators (επαναλήπτες): ≪Δείκτες≫ που επιτρέπουν τη διάσχιση και σε ορισμένες περιπτώσεις την αλλαγή ενός container. Ένας iterator δείχνει σε κάποιο στοιχείο ενός container.
- Containers: Σε ένα container μπορούν να αποθηκευτούν τιμές βασικών τύπων καθώς και αντικείμενα. Containers χωρίζονται σε δύο κατηγορίες.
  - Ακολουθιακά containers (sequence containers) – κάθε αντικείμενο ακολουθείται από κάποιο άλλο αντικείμενο, έχουν δηλαδή γραμμική διευθέτηση:
    - vector
    - array
    - list
    - forward list
    - deque
    - stack
    - queue
  - Containers αντιστοίχισης (associative containers) – γίνεται χρήση κλειδιών για την πρόσβαση στα στοιχεία του container, επιτρέπουν τη γρήγορη πρόσβαση βάσει κλειδιών.
    - set
    - map

2 Vector

Διάνυσμα με δυνατότητα δυναμικής αύξησης κατά το runtime του μεγέθους του έτσι ώστε να δεχθεί νέα στοιχεία. Χρησιμοποιούν συνεχόμενες τοποθεσίες αποθήκευσης για τα στοιχεία τους εξίσου αποτελεσματικά με τις συστοιχίες, που σημαίνει ότι τα στοιχεία τους μπορούν επίσης να έχουν πρόσβαση χρησιμοποιώντας αντισταθμίσεις(offset) σε τακτικούς δείκτες στα στοιχεία της. Οι βασικές λειτουργίες ενός vector παρουσιάζονται στην εικόνα 1.

C++ Vector Functions

In C++, the vector header file provides various functions that can be used to perform different operations on a vector.

Function	Description
`size()`	returns the number of elements present in the vector
`clear()`	removes all the elements of the vector
`front()`	returns the first element of the vector
`back()`	returns the last element of the vector
`empty()`	returns **1** (true) if the vector is empty
`capacity()`	check the overall size of a vector

Figure 1: Βασικές λειτουργίες vectors

2.1 Βασικές λειτουργίες

- Αρχικοποιήση ενός vector

```cpp
#include <iostream>
#include <vector>

using namespace std;

int main()
{

    vector <int> v; // Άδειο vector
    vector <int> v1(10); //vector με 10 αντίγραφα της προκαθορισμένης τιμής.
    vector <int> v2(10,5); //vector με 10 αντίγραφα της τιμής 5.

    cout<<"Vector 1:"<<v.size()<<endl;
    cout<<"Vector 2:"<<v1.size()<<endl;
    cout<<"Vector 3:"<<v2.size()<<endl;

    cout<<"Vector 1 Capacity:"<<v.capacity()<<endl;
    cout<<"Vector 2 Capacity:"<<v1.capacity()<<endl;
    cout<<"Vector 3 Capacity:"<<v2.capacity()<<endl;

    return 0;
    // v.size() // μεγεθος vector
    // v.capacity // χωρητικότητα vector


    // Output
    // Vector 1:0
    // Vector 2:10
    // Vector 3:10
    // Vector 1 Capacity:0
    // Vector 2 Capacity:10
    // Vector 3 Capacity:10
}
```

Figure 2: Αρχικοποιήση vectors

- Προσθήκη τιμών σε ένα vector.

```cpp
#include <iostream>
#include <random>
#include <chrono>
#define SIZE 30
using namespace std;
using namespace std::chrono;

mt19937 mt(high_resolution_clock::now().time_since_epoch().count());

int main()
{
    uniform_int_distribution <int> rand_int(1,5000);
    vector <int> v;

    // Προσθήκη τυχαίων τιμών στο διάστημα [1,5000] στο vector v
    for(int i=0;i<SIZE;i++)
    {
        v.push_back(rand_int(mt));
    }

    for(int i=0;i<v.size();i++)
    {
        cout<<"v["<<i<<"]:"<<v[i]<<endl;
    }
    return 0;
}
```

Figure 3: Vector push_back

- Η βιβλιοθήκη algorithm.

```cpp
#include <iostream>
#include <vector>
using namespace std;

int main()
{
    vector <int> v{2,4,6,7,8,9};
    for(auto itr=v.begin();itr!=v.end();itr++)
    {
        cout<<"*itr:"<<*itr<<endl;
    }
    return 0;
}
```

Figure 4: Αλγόριθμοι πάνω σε vectors

Δεικτοδότηση

- Με τον τελεστή []. Υψηλή απόδοση, η πρόσβαση γίνεται χωρίς έλεγχο ορίων του vector.
- Με την συνάρτηση μέλος at(). Πραγματοποιεί έλεγχο ορίων, προκαλεί την εξαίρεση out_of_range exception αν επιχειρηθεί πρόσβαση εκτός των ορίων του vector.

iterators Οι iterators(αντικείμενο(παρόμοια λειτουργία με τους δείκτες)) χρησιμοποιούνται για να δείξουν τις διευθύνσεις μνήμης των containers της STL. Χρησιμοποιούνται κυρίως σε αλληλουχίες αριθμών, χαρακτήρων κ.λπ. μειώνουν τον χρόνο πολυπλοκότητας και εκτέλεσης του προγράμματος. Οι iterators διαδραματίζουν κρίσιμο ρόλο στη σύνδεση των αλγορίθμων που εφαρμόζονται σε container μαζί με τον χειρισμό των δεδομένων που είναι αποθηκευμένα μέσα στα container. Η πιο προφανής μορφή ενός iterator είναι ένας δείκτης.

Προσθήκη τιμών σε ένα vector.

```cpp
#include <iostream>
#include <vector>
using namespace std;

int main()
{
    vector <int> v{2,4,6,7,8,9};
    for(auto itr=v.begin();itr!=v.end();itr++)
    {
        cout<<"*itr:"<<*itr<<endl;
    }
    return 0;
}
```

Figure 5: Vector iterators

3 UML

- Η ενοποιημένη γλώσσα μοντελοποίησης (UML) είναι μια γενική, αναπτυξιακή γλώσσα μοντελοποίησης στον τομέα του software engineering που προορίζεται να παρέχει έναν τυπικό τρόπο για την απεικόνιση του σχεδιασμού ενός συστήματος.

Class diagrams
Τα διαγράμματα κλάσεων απότελούνται από τρεις τομείς:

- Πρώτο επίπεδο: Το επάνω τμήμα περιλαμβάνει το όνομα της κλάσης. Μια κλάση είναι μια αναπαράσταση παρόμοιων αντικειμένων που μοιράζονται τις ίδιες σχέσεις, χαρακτηριστικά, λειτουργίες και σημασιολογία. Μερικοί από τους ακόλουθους κανόνες που πρέπει να ληφθούν υπόψη ενώ αντιπροσωπεύουν μια κλάση δίνονται παρακάτω:
  - Το όνομα της κλάσης είναι με κεφαλαία και τοποθετήται στο πρώτο επίπεδο.
- Δεύτερο επίπεδο: Το μεσαίο τμήμα αποτελεί τα χαρακτηριστικά, τα οποία περιγράφουν την ποιότητα της κλάσης. Τα χαρακτηριστικά έχουν τα ακόλουθα χαρακτηριστικά. Τα χαρακτηριστικά είναι γραμμένα μαζί με τους παράγοντες ορατότητας, οι οποίοι είναι δημόσιοι (+), ιδιωτικοί (−), προστατευμένοι (). Η προσβασιμότητα μιας κλάσης χαρακτηριστικών απεικονίζεται από τα επίπεδα ορατότητας. Ένα αντιπροσωπευτικό όνομα θα πρέπει να αντιστοιχιστεί στο χαρακτηριστικό, το οποίο θα εξηγήσει τη χρήση του μέσα στην τάξη.
- Τρίτο επίπεδο: Περιέχει μεθόδους ή λειτουργίες. Οι μέθοδοι αντιπροσωπεύονται με τη μορφή μιας λίστας, όπου κάθε μέθοδος γράφεται σε μία μόνο γραμμή. Δείχνει πώς μια κλάση αλληλεπιδρά με δεδομένα.

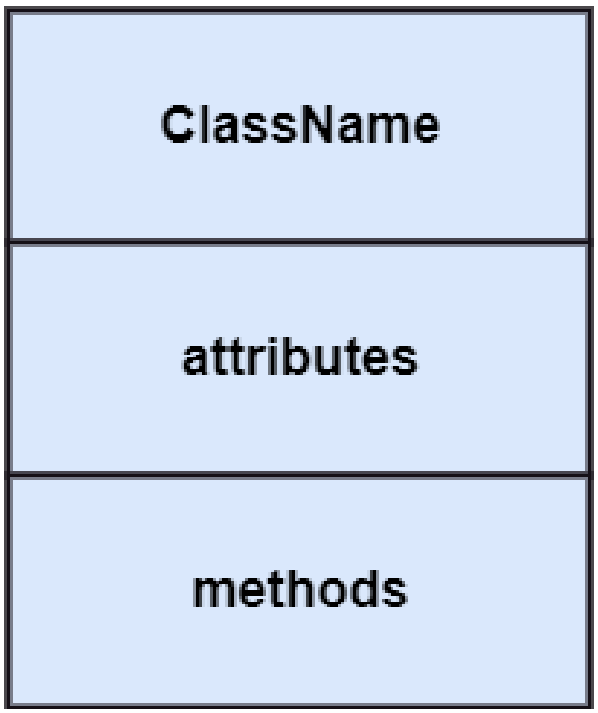

Figure 6: Διάγραμμα κλάσεων, επεξήγηση επιπέδων

Παρακάτω παρουσιάζετε η υλοποιήση ενός διαγράμματος κλάσης σε C++.

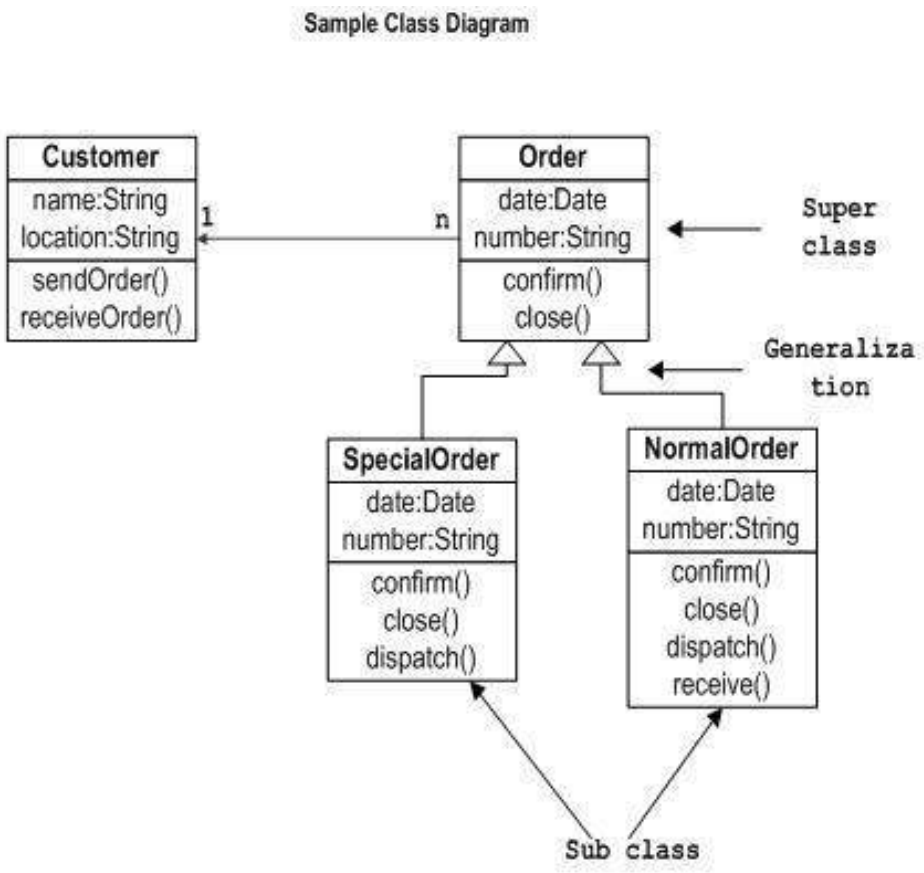

Figure 7: Διάγραμμα κλάσσεων

```cpp
#include <iostream>
#include <vector>
using namespace std;

class Customer
{
    string name;
    string location;
    public:
        Customer(string n,string loc):name(n),location(loc) {}
        void sendOrder() {}
        void receiveOrder() {}
};

class Order
{
    string date;
    string number;
    vector <Customer> customers;
    public:
        Order(string d,string n):date(d),number(n) {}
        void confirm() {}
        void close() {}
};

class SpecialOrder:public Order
{
    public:
        SpecialOrder(string d,string n):Order(d,n) {}
        void dispatch() {}
};

class NormalOrder:public Order
{
    public:
        NormalOrder(string d,string n):Order(d,n) {}
        void dispatch() {}
        void receive() {}
};
```

Figure 8: Κώδικας εικόνας 7

4 Ασκήσεις

1. Να κατασκευαστεί κώδικας για το UML διάγραμμα της εικόνας 9. Κατασκευάστε τα κατάλληλα αντικείμενα και παρουσιάστε παραδείγματα εκτέλεσης στην main.

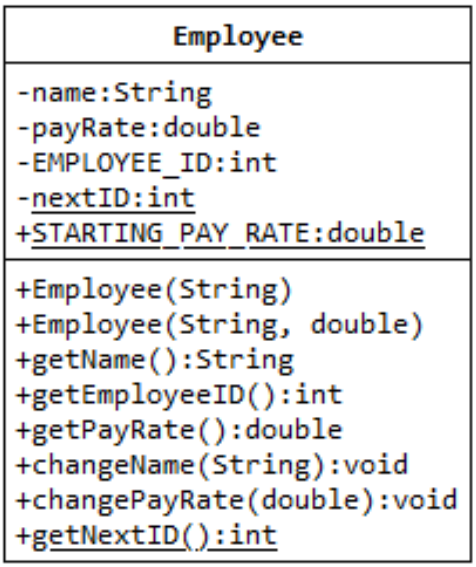

Figure 9: Διάγραμμα κλάσης UML_2

2. Να γραφεί κατηγορία με το όνομα Rectangle. Στα ιδιωτικά της πεδία θα είναι:

   - pleyra1
   - pleyra2
   - count(static member)

   Στα δημόσια πεδία

   - Μια μέθοδος δημιουργίας.
   - Μια μέθοδος με το όνομα area() για υπολογισμό εμβαδού.
   - Μια μέθοδος με το όνομα volume() για υπολογισμό όγκου που θα επιστρέφει 0 και θα είναι υπερβατική
   - Μια μέθοδος με το όνομα details() η οποία θα εμφανίζει το εμβαδόν και τον όγκο

   Στην συνέχεια να φτιαχτεί η κατηγορία Box που θα κληρονομεί την Rectangle με επιπλέον πεδίο την τρίτη πλευρα.

3. Στο αρχείο in.txt υπάρχουν δύο στήλες αριθμών. Στην πρώτη στήλη είναι ο μισθός ενός υπαλλήλου και στην δεύτερη τα bonus που θα πάρει για την νέα χρονιά χωρισμένα με κόμμα όπως παρουσιάζονται στον πίνακα 1. Στην τελευταία γραμμή υπάρχει η καταχώρηση 0 0. Οι δύο αριθμοί χωρίζονται μεταξύ τους με κενό. Για την κλάση Employee θα υλοποιηθεί η συνάρτηση compute_salary() η οποία θα υπολογίζει τον συνολικό μισθό του έτος με βάση της μηνιαίες παρακρατήσης του πίνακα 1.

Μήνας	Bonus	παρακράτηση
L1	**2000**	**50,60,40,50,25,35,100,120,110,30,50,100**
1	50	5%
2	60	6%
3	40	5%
4	50	6%
5	25	5%
6	35	6%
7	100	5%
8	120	6%
9	110	5%
10	30	6%
11	50	5%
12	100	6%

Table 1: Παράδειγμα γραμμής αρχείου δεδομένων

Να γραφεί πρόγραμμα το οποίο:

- θα διαβάζει αυτά τα αρχεία
- θα αποθηκεύει αυτές τις καταχωρήσεις σε ένα vector από αντικείμενα της κατηγορίας Employee
- θα ταξινομεί αυτόν τον πίνακα με βάση τις συνολικές αποδοχές των υπαλλήλων
- θα εμφανίζει στο τέλος τους δύο υπαλλήλους με τις υψηλότερες αποδοχές.

4. Κατασκευάστε την ακόλουθη ιεραρχία. Την υπερκλάση person (άτομο) με πεδίο δεδομένων age (ηλικία) και τις υποκλάσεις της person: teacher (καθηγητής) με πεδίο δεδομένων profession (εξειδίκευση) και footballer (ποδοσφαιριστής) με πεδίο δεδομένων team (ομάδα).

   - Δημιουργήστε από έναν κατασκευαστή σε κάθε κλάση και έναν virtual καταστροφέα στην κλάση person.
   - Προσθέστε τις κατάλληλες μεθόδους έτσι ώστε να μπορεί να κληθεί η συνάρτηση compute_earnings (υπολογισμός εσόδων) και να επιστρέφει την τιμή 1000.0 για τα αντικείμενα teacher και 100000.0 για τα αντικείμενα footballer.
   - Προσθέστε στην κλάση person στατική μεταβλητή με όνομα number_of_instances που θα καταμετρά το πλήθος των αντικειμένων τύπου person (ή των υποτύπων της) που θα δημιουργούνται. Εμφανίστε το πλήθος στη main.

   Στη main,

   - δημιουργήστε έναν πίνακα 5 θέσεων που ο χρήστης θα γεμίζει με δείκτες προς αντικείμενα teacher ή footballer πραγματοποιώντας εισαγωγή τιμών από το πληκτρολόγιο.
   - καλέστε τη συνάρτηση compute_earnings για κάθε στοιχείο του πίνακα και εμφανίστε τα αποτελέσματα.

5. Κατασκευάστε τις κλάσεις που δείχνει το UML διάγραμμα κλάσεων στην εικόνα 10 και οι οποίες αναπαριστούν έγγραφα, βιβλία και emails. Αναλυτικότερα, κατασκευάστε τα ακόλουθα:

   - Κλάσεις - ιεραρχία κλάσεων
   - Κατασκευαστές
   - Getters και συναρτήσεις add_author[1] και add_recipient[2]
   - Συνάρτηση info[3] και στις 3 κλάσεις.

   Στη main, δημιουργήστε τα ακόλουθα αντικείμενα:

[1] Η συνάρτηση add_author προσθέτει 1 όνομα συγγραφέα στο διάνυσμα των συγγραφέων (authors)
[2] Η συνάρτηση add_recipient προσθέτει 1 όνομα παραλήπτη στο διάνυσμα των παραληπτών email (recipients)
[3] Η συνάρτηση info επιστρέφει ένα λεκτικό με όλα τα στοιχεία του αντικειμένου για το οποίο καλείται

- Αντικείμενο book: title = “book1”, date = “1/1/2010”, authors = “nikos”, “maria”
- Αντικείμενο book: title = “book2”, date = “30/6/2015”, authors = “kostas”
- Αντικείμενο email: sender= “petros”, subject= “email1”, recipients = “maria”, “nikos”, date = “1/6/2017”, authors = “petros”, “kostas”

και Εισάγετε σε ένα vector¡document*¿ δείκτες προς τα παραπάνω αντικείμενα. Τέλος Ταξινομήστε τα περιεχόμενα του vector και εμφανίστε τα περιεχόμενα του vector καλώντας τη συνάρτηση info για κάθε αντικείμενο. Τα αποτελέσματα θα πρέπει να εμφανίζονται όπως παρακάτω:

```
BOOK-> Title: book2 Date: 30/6/2015 Authors: kostas
BOOK-> Title: book1 Date: 1/1/2010 Authors: nikos maria
EMAIL-> Subject: email1 Sender: petros Recipients: maria nikos Date:
    ↪ 1/6/2017 Authors: petros kostas
```

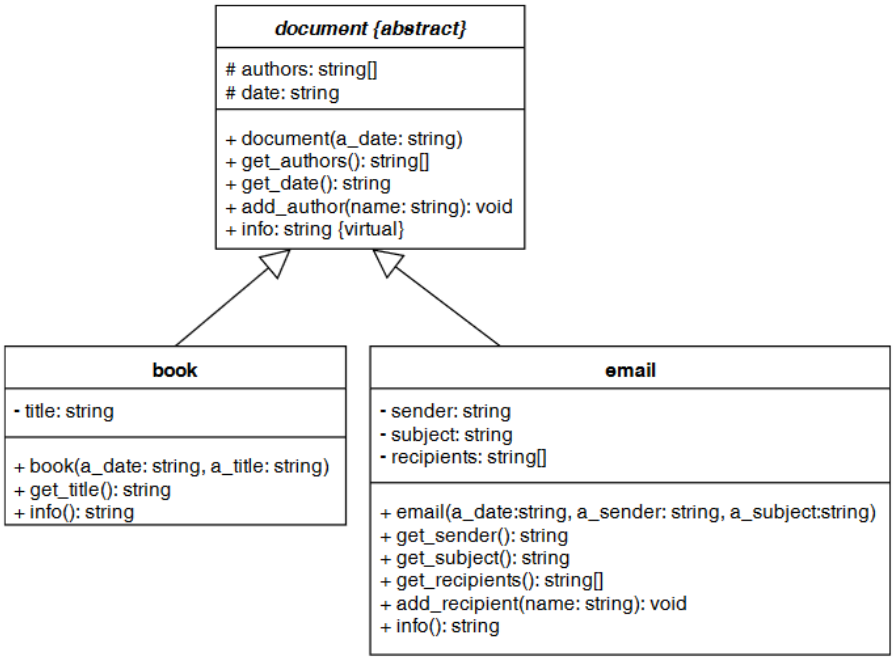

Figure 10: UM διάγραμμα κλάσεων άσκηση 4

6. Κατασκευάστε τις κλάσεις που δείχνει το UML διάγραμμα κλάσεων στην εικόνα 11 και οι οποίες αναπαριστούν ταινίες (movie) που σκηνοθετούνται από ένα άτομο (person) και έχουν ως ηθοποιούς ένα σύνολο (std::vector) από άτομα. Στη συνέχεια δημιουργήστε 2 αντικείμενα ταινίες με τίτλους “Movie 1” και “Movie 2”, ορίστε ότι στη ταινία “Movie 1” είναι σκηνοθέτης ο “Director 1” και παίζουν οι ηθοποιοί “Actor 1” και “Actor 2” ενώ στη ταινία “Movie 2” είναι σκηνοθέτης ο “Director 2” και παίζουν οι ηθοποιοί “Actor2” και “Actor 3”. Εμφανίστε καλώντας τη συνάρτηση μέλος display_info() της κλάσης movie τα στοιχεία της κάθε ταινίας, δηλαδή τίτλο, σκηνοθέτη και ηθοποιούς.

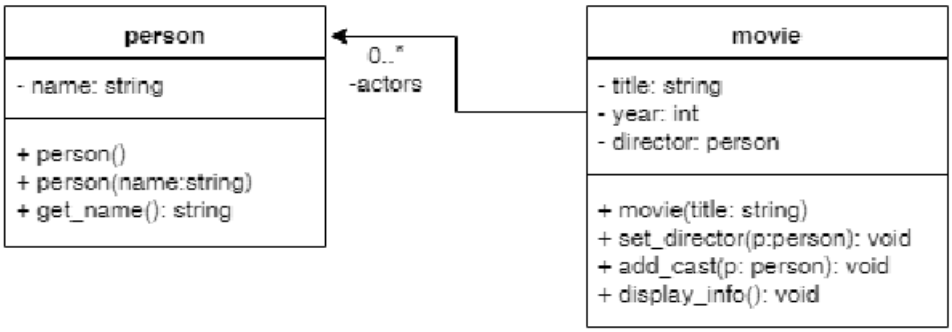

Figure 11: UML διάγραμμα κλάσεων άσκηση 5

7. Κατασκευάστε τις κλάσεις που αναπαρίστανται στο ακόλουθο UML διάγραμμα κλάσεων (Σχήμα 2). Η κλάση student αναπαριστά φοιτητές και η κλάση lab_class αναπαριστά τμήματα εργαστηρίου. Κάθε φοιτητής μπορεί να είναι εγγεγραμμένος σε ένα εργαστήριο το πολύ και κάθε εργαστήριο μπορεί να έχει πολλούς φοιτητές.

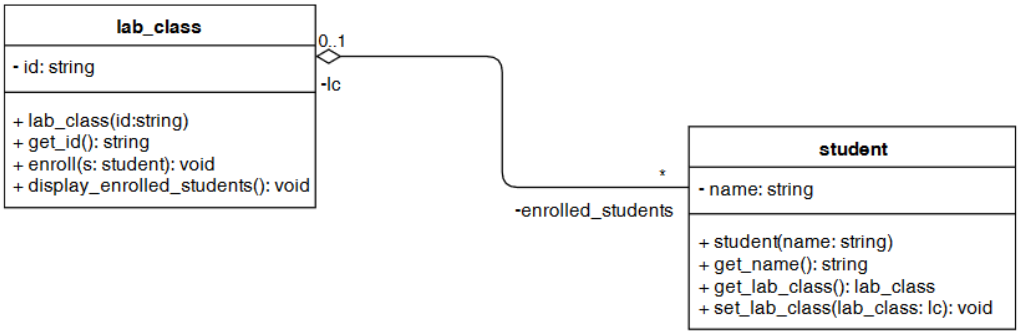

Figure 12: UML διάγραμμα κλάσεων άσκηση 6

Στη main() του προγράμματος:

- Δημιουργήστε πέντε αντικείμενα φοιτητές και φοιτήτριες με ονόματα kostas, petros, maria, christina και tasos και δύο αντικείμενα τμήματα εργαστηρίου με κωδικούς OOP1 και OOP2.
- Εγγράψτε τους 2 πρώτους φοιτητές στο τμήμα OOP1 και τους δύο τελευταίους στο τμήμα OOP2. Προσπαθήστε να εγγράψετε τον kostas στο εργαστήριο OOP2 έτσι ώστε να εμφανιστεί μήνυμα σφάλματος καθώς είναι ήδη εγγεγραμμένος στο τμήμα OOP1.
- Εμφανίστε λίστα εγγεγραμμένων φοιτητών για κάθε τμήμα.