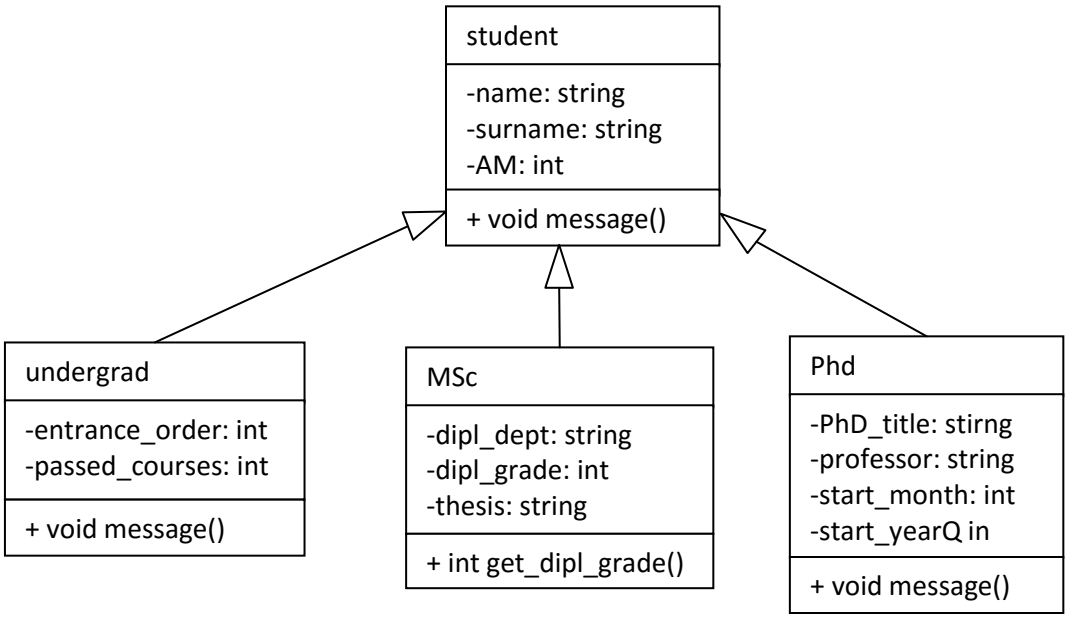
Οντοκεντρικός Προγραμματισμός (C++)
Ακαδ. Έτος: 2016-17

3ο ΣΕΤ ΑΣΚΗΣΕΩΝ

Παραδίδεται μαζί με τις απαντήσεις στις ασκήσεις του φυλλαδίου εργαστηριακών ασκήσεων C++

ΑΣΚΗΣΗ 1

Να κατασκευαστεί πρόγραμμα C++ που να δημιουργεί την ιεραρχία κλάσεων που απεικονίζονται στο διάγραμμα UML που δίνεται:



Σημειώνεται ότι όλες οι υποκλάσεις κληρονομούν με τρόπο public την υπερκλάση. Η μέθοδος message() στην κλάση Student θα εμφανίζει στην κονσόλα το μήνυμα «Hello, I am message() defined in superclass Student», και με αντίστοιχο τρόπο θα λειτουργούν και οι υπόλοιπες message() στις υποκλάσεις. Τοποθετείστε τον κώδικα ορισμού των κλάσεων στον κώδικα που ακολουθεί

```
#include  <iostream>
#include  <cstdlib>

using namespace std;

/*Προσθέστε τον κώδικα των κλάσεων*/


int  main()
{
  student  S1;
  undergrad    U1;
  MSc    M1;
```

```
  PhD     Ph1;

  S1.message();  //grammh 1
  U1.message();  //grammh 2
  M1.message();  //grammh 3
  Ph1.message();  //grammh 4

  system("pause");
  return 0;
}
```

α. Εκτελέστε τον κώδικα και εξηγήστε τα μηνύματα που εμφανίζονται (ποια message() εκτελείται κάθε φορά και γιατί;)
β. Πριν το system ("pause") στη main προσθέστε την εντολή `S1 = U1` και στη συνέχεια καλέστε την `message()` επί του αντικειμένου S1. Ποια υλοποίηση της `message()` εκτελέστηκε και γιατί;

γ. Αλλάξτε τη main ώστε η μεταβλητή S1 να είναι δείκτης προς αντικείμενο τύπου student. Αναθέστε στο S1 ένα νέο αντικείμενο της κλάσης student (καλώντας τον default κατασκευαστή) και καλέστε τη μέθοδο message() στο S1. Στη συνέχεια αναθέστε στο S1 ένα νέο αντικείμενο της κλάσης undergrad (καλώντας τον κατασκευαστή) και καλέστε την message() στο S1. Επαναλάβετε το ίδιο για την υποκλάση PhD. Εξηγήστε τα μηνύματα που εμφανίζονται (ποια message() εκτελείται κάθε φορά και γιατί;);
Ποια αλλαγή πρέπει να γίνει στην message() της student ώστε να εκτελείται ο κώδικας που αντιστοιχεί στην κλάση του αντικειμένου που 'δείχνει΄ κάθε φορά ο δείκτης student; Κάντε την αλλαγή και εκτελέστε τον κώδικα ώστε να επιβεβαιώσετε ότι πλέον υλοποιείται δυναμική διασύνδεση (dynamic ή late binding). *Ίσως χρειαστεί να κάνετε rebuild το project για να φανούν οι αλλαγές.*

δ. Αντί των 4 διαδοχικών κλήσεων της message() στη main που προηγήθηκε, κάντε τις απαραίτητες αλλαγές ώστε να κατασκευάζεται ένας πίνακας και να κρατούνται εκεί δείκτες σε αντικείμενα τύπου student. Τοποθετείστε στον πίνακα δείκτες προς αντικείμενα των κλάσεων της ιεραρχίας που δόθηκαν (χρησιμοποιήστε και τις 4 κλάσεις) και μέσω ενός for loop καλέστε την message() διαδοχικά σε όλα τα στοιχεία του πίνακα. Επιβεβαιώστε ότι ο κώδικας λειτουργεί σωστά.

ΑΣΚΗΣΗ 2

α. Εκτελέστε τον παρακάτω κώδικα. Ποιά είναι η λειτουργία του;

```
#include <iostream>
#include <string>

using namespace std;

int main()
{
   string best_name = "";
   double best_price = 0;
   int best_score = 0;

   bool more = true;
   while (more)
   {
      string next_name;
      double next_price;
      int next_score;

      cout << "Please enter the model name: ";
```

```
      getline(cin, next_name);
      cout << "Please enter the price: ";
      cin >> next_price;
      cout << "Please enter the score: ";
      cin >> next_score;
      string remainder; /* read remainder of line */
      getline(cin, remainder);

      if (next_score / next_price > best_score / best_price)
      {
         best_name = next_name;
         best_score = next_score;
         best_price = next_price;
      }

      cout << "More data? (y/n) ";
      string answer;
      getline(cin, answer);
      if (answer != "y") more = false;
   }

   cout << "The best value is " << best_name
      << " (Score: " << best_score
      << " Price: " << best_price << ")\n";

   return 0;
}
```

β. Εναλλακτικά και με σκοπό να υλοποιηθεί η ίδια λειτουργικότητα αλλά με μια αντικειμενοστρεφή προσέγγιση, ορίστε μια κλάση Product με τα απαραίτητα μέλη (ιδιότητες και μεθόδους) ώστε αν συνδυαστεί με τον κώδικα που ακολουθεί να έχουμε το ίδιο αποτέλεσμα με τον αρχικό κώδικα. Φροντίστε η κλάση product να τηρεί την αρχή της απόκρυψης των πληροφοριών και προσθέστε κατάλληλο προσδιοριστικό στη δήλωση των μεθόδων read() και print() ώστε να μην μπορούν να αλλάξουν τα δεδομένα της κλάσης.

```
#include <iostream>
#include <string>

using namespace std;

//προσθέστε τον ορισμό της κλάσης Product


int main()
{
   Product best;

   bool more = true;
   while (more)
   {
      Product next;
      next.read();
      if (next.is_better_than(best)) best = next;

      cout << "More data? (y/n) ";
      string answer;
      getline(cin, answer);
```

```
      if (answer != "y") more = false;
   }

   cout << "The best value is ";
   best.print();

   return 0;
}
```

γ) Αντικαταστήστε την print() που ορίσατε στον προηγούμενο κώδικα με μια συνάρτηση friend της κλάσης product που θα λειτουργεί με τον ίδιο τρόπο. Παρατηρείστε ότι μια friend συνάρτηση δε μπορεί να είναι const οπότε αφαιρέστε το προσδιοριστικό const. Εκτελέστε τον νέο κώδικα ώστε να επιβεβαιώσετε ότι λειτουργεί σωστά.

δ) Υπερφορτώστε τον τελεστή > ώστε να λειτουργεί όπως η συνάρτηση `is_better_than()` και δοκιμάστε τη λειτουργία του χρησιμοποιώντας τον υπερφορτωμένο τελεστή στη θέση της συνάρτησης στη main.