COMP 2404 -- Assignment #1

Due: Friday, February 12, 2016 at 12:00 (noon)

Collaboration: You may work in groups of no more than two (2) students

Goal

You will be working with an existing bookstore management program throughout the term. The **base code** that you must start with is posted in *cuLearn*. For this assignment, you will modify the base code to use dynamically allocated memory. You will also implement two new features.

Learning Objectives

- understand an existing object-oriented program
- understand entity/control/UI classes, and their role in a program
- add code to a small program in C++
- practice with dynamically allocated memory in C++

Instructions:

1. Draw the UML class diagram:

- Read and make sure that you thoroughly understand the existing bookstore management program provided in the base code.
- Using a drawing package of your choice, draw a UML class diagram to represent the objects you identified. You must show all classes (except the array classes), their attributes and operations, the associations between the classes (inheritance or composition), and the multiplicity and directionality of all associations. Do not show getter and setter functions, nor constructors or destructors.

2. Modify the program to use dynamically allocated memory

You will change the base code to allocate all objects dynamically. This includes:

- the dynamic allocation of the control, UI and array objects
- the dynamic allocation of all course and book objects
- the modification of both array classes to contain an **array of pointers** to objects
- the explicit deallocation of all dynamically allocated memory; memory leaks will be penalized

3. Implement the "add textbook" feature

You will implement the "add textbook" feature that can be selected from the main menu:

- the user will be prompted to enter the course code that the textbook belongs to
- your program will verify that this course exists
- the user will be prompted to enter all the textbook information
- the new textbook will be created and added to the course's collection of books

4. Implement the "list course textbooks" feature

You will implement the "list course textbooks" feature that can be selected from the main menu:

- the user will be prompted to enter the course code for the course whose textbooks will be printed out
- your program will verify that this course exists
- the data for all textbooks for the selected course will be printed to the screen

Constraints

- your program must follow the existing design of the base code, including the encapsulation of control, UI, entity and array object functionality
- do not use any classes or containers from the C++ standard template library (STL)
- do **not** use any global variables or any global functions other than **`main`**
- do **not** use structs; use classes instead
- objects must always be passed by reference, not by value

Submission

You will submit in *cuLearn*, before the due date and time, the following:

- a UML class diagram (as a PDF file) that corresponds to the program design
- one **`tar`** file that includes:
  - all source and header files for your program
  - a Makefile
  - a readme file that includes:
    - a preamble (program and modifications authors, purpose, list of source/header/data files)
    - compilation, launching and operating instructions

If you are working with a partner:

- only **one partner** submits the assignment, and the other partner submits nothing; partners that make two separate submissions will be considered to have plagiarized each other's assignment and will be reported accordingly
- the readme file must contain the names of both partners
- the submitting partner must enter the names of both partners in the **Online Text** box of the *cuLearn* submission link

***** LATE ASSIGNMENTS WILL NOT BE ACCEPTED FOR ANY REASON *****

Grading

Marking components:

- UML diagram: 25%
  - 10 marks: Control object
    - 4 marks: correct relationship with UI object
    - 4 marks: correct relationship with entity object(s)
    - 2 marks: correct operations
  - 4 marks: Correct relationship between entity objects
  - 5 marks: Correct attributes and operations on the UI object
  - 6 marks: Correct attributes and operations on the entity objects (3 marks each)
- Using dynamically allocated memory: 40%
  - 16 marks: dynamic allocation and deallocation of control, UI and both types of array objects (4 marks each object)
  - 10 marks: dynamic allocation and deallocation of all course and book objects (5 marks each)
  - 14 marks: modification of both array classes to manage an array of pointers to objects (7 marks each)
  - *Deduction*: deduct half marks for missing deallocation
- Add textbook feature: 20%
  - 5 marks: reads course code from user
  - 5 marks: finds course in array
  - 5 marks: creates and initializes new book
  - 5 marks: adds book to course
- List course textbooks feature: 15%
  - 5 marks: reads course code from user
  - 5 marks: finds course in array
  - 5 marks: prints out textbooks to the screen

Deductions:

- Packaging errors:
  - 10 marks for missing Makefile
  - 5 marks for missing readme
  - 10 marks for **consistent** failure to correctly separate code into source and header files
- Major programming and design errors:
  - 50% of a marking component that uses global variables, global functions, or structs
  - 50% of a marking component that consistently fails to use correct design principles
  - 50% of a marking component that uses prohibited library classes or functions
  - 50% of a marking component where unauthorized changes have been made to provided code
- Execution errors:
  - 100% of a marking component that cannot be tested because the code does not compile or execute
  - 100% of a marking component that cannot be tested because the feature is not used in the code