COMP 2404 -- Tutorial #8

Inheritance and Polymorphism

Learning Outcomes

After this tutorial, you will be able to:

- use inheritance and polymorphism in C++

Instructions

1. You will begin with the code you saved from Tutorial #7.

2. We are going to update our inheritance hierarchy of books to implement polymorphism. The `Book` class will become abstract, and the two sub-classes `FictionBook` and `NonFictionBook` will be concrete. The main difference between the two is how they will be ordered in each library. Fiction books will be added in ascending alphabetical order of author name, and non-fiction books will be added in ascending order of their *call number*, which is based on the Dewey Decimal Classification system. To implement this, the `Book` class's `lessThan()` function will become a pure virtual function, and the appropriate behaviour will be implemented in the concrete classes.

   Draw a UML class diagram to represent the new class associations in the program.

3. Modify the `Book` class as follows:
   - add a data member for the call number, which can be a string
     - update the constructor and print function accordingly
   - make `lessThan()` a pure virtual function
   - existing private members may be better re-classified as protected
   - we will need getter member functions for both the author name and the call number, since they will be used every time a book is added
   - you will need to add a virtual destructor as well

4. Update the two derived classes `FictionBook` and `NonFictionBook` as follows:
   - update the parameter list of each constructor to include the call number, and make sure the constructors invokes the base class constructor correctly
   - declare and implement each class's virtual function `bool lessThan(Book*)`
     - the function for `FictionBook` will compare authors
     - the function for `NonFictionBook` will compare call numbers

5. Modify the `View` class as follows:
   - update the read book information function to prompt the user for a call number; while it only makes sense for non-fiction books to have a call number, it's simpler to allow them on all books

6. Modify the `Control` class to create the `FictionBook` and `NonFictionBook` objects with the new data member, and add the book to the correct library.

7. Modify the `in.txt` file so that your program is thoroughly tested.

8. Build and run the program. Check that each book is in the correct library, and that the books are ordered correctly, both in the forward direction and the backward direction, when both libraries are printed out at the end of the program. Fiction books should be ordered by author, and non-fiction books by call number.

9. Make sure that all dynamically allocated memory is explicitly deallocated when it is no longer used. Use valgrind to check for memory leaks.

10. Package together the tutorial code into a tar file, and upload it into cuLearn. Save your work to a permanent location, like a memory stick or your Z-drive.