CO882 Assignment 1 - Inheritance & Polymorphism

Introduction

This assignment is designed to help you explore the subject of *inheritance* and *polymorphism* in object-oriented languages. In particular, you will be exploring how shared characteristics of related classes can be represented in a *superclass*, with the specialised elements of the related classes represented in multiple descendent *subclasses*.

You will work with an existing project that does not currently use inheritance. The scenario is a local library that uses a database to store its collections of publications, e.g. books and monthly journals.

Deadline and Submission

- The submission deadline is at **11:55pm on Friday in week 10 (KV week 18)**.

  A mark of zero will be awarded for late or non submission of the coursework.

- You should put all your work in a single ZIP file (.zip) and then upload the file using "A1 submission link" on the module Moodle page before the submission deadline.

Important

Before working on the tasks below, you should do the followings:

- Download the project **`co882-A1-Inheritance.zip`** using the link on the module Moodle page and unzip the file before working on the given project.
- Study carefully the fields and methods defined in each of the given classes and find out what they are and how to use them.
- Invoke the method `testAll` of the `DatabaseTest` class to run the tests on the methods of the `Database` class. Study the code in the method `testAll` to understand how these tests are done and why.

Tasks

1. **Introduce inheritance into the project (12 marks)**

   The `Book` and `Journal` classes share some common fields (e.g. `title` and `year`). They also have some common methods (e.g. `getTitle` and `getYear`). Capture these common elements in a new class named `Publication`, that becomes the *superclass* of both `Book` and `Journal`.

   This change involves placing the common fields and methods into the superclass `Publication` and removing them from both `Book` and `Journal` classes. Rather than making the changes all in one go, it will be safer to move one field at a time.

   a) Moving the `title` field

      In order to move the `title` field you will need to engage in a process called **refactoring**. The aim is to improve the class structures by moving code around, but we are not aiming to introduce new functionality. Once these changes are made, all the existing tests should still pass.

      - Start by creating a new class called `Publication`.

- Modify `Book` and `Journal` to indicate that both are subclasses of `Publication`.
- Place a `title` field in `Publication` and remove the `title` fields from both `Book` and `Journal` classes. Make sure that the `title` has an appropriate comment.
- Place a `getTitle` accessor method in `Publication` and remove similar definitions from `Book` and `Journal`. Make sure the text of the method comment is appropriate to a shared method.
- With `title` now a private field of `Publication`, subclasses cannot use the field directly in their methods. Both `Book` and `Journal` must replace direct accesses with calls to the public method `getTitle` they inherit from `Publication`. Make these changes to `Book` and `Journal` and check that all classes compile correctly.
- Check for any errors introduced during the refactoring process. One particular problem to look out for is whether a book/journal is initialised correctly. If you have not already done so, you might need to think about how the title of the book/journal is set when it is created. This means you will have to ensure that the subclass constructors call the superclass constructor correctly.

b) Moving the `year` field

When you have successfully moved the `title` field to the `Publication` class you can move the `year` field. The process will be similar.

c) Define the method `toString` in `Publication` that takes nothing and returns a String containing the details of the publication. And then modify the `toString` method in both `Book` and `Journal` classes to make use of the `toString` method in the superclass `Publication`.

You should test thoroughly to check for any errors that might have been introduced.

2. Introduce polymorphism (6 marks)

a) The Database class has two fields of ArryList: `bookList` and `journalList`. Replace these fields by a single field `publicationList` of ArrayList. Make sure it has an appropriate comment. Note also that you need to change the constructor and `getTotal` methods.

b) Methods `addBook` and `addJournal` are very similar. Replace these two methods by a single method `addPublication`. It takes an object of `Publication` and adds it to `publicationList`. Make sure the text of the method comment is appropriate.

c) There are two loops in the method `printList` which are very similar. Replace these two loops by a single loop to print out the details of all elements in `publicationList`.

You should test thoroughly to check for any errors that might have been introduced.

3. Improve the Database class (17 marks)

a) Modify the `addPublication` method such that it checks, before adding a publication into `publicationList`, if it is already in the list. If so, it should instead print out an appropriate message with the details of the publication.

b) Write a method that returns a list of the books published in a given year.

c) Write a method that prints out the details of all publications. The output should be ordered on the category of the publications, and then on the title within each category.

Note: Currently there are 2 categories: Book and Journal. It may be expended by adding other categories, e.g. Newspaper etc.

Add further tests in the `DatabaseTest` class to check that your methods work as expected.