Hw7 Problem 4

財金所碩三 r10723057 黃元裕

一個 C++程式開發的任務，如果要用物件導向程式設計（OOP）來實作會比程序式程式設計（Procedural Programming）好得多的例子，是「線上圖書館管理系統」。

在這個系統中，必須處理書籍、借閱者、員工等多種實體和它們之間的互動。以下是用物件導向來開發這個系統的一些優點：

1. 封裝（Encapsulation）：物件導向允許我們將資料和與之相關的行為打包在一起。例如，一個 Book 類別可以包含標題、作者、ISBN 和方法，如借出和歸還。這使得程式更加模組化，易於維護和擴展。
2. 繼承（Inheritance）：系統中的不同類型的書籍可以從基礎的 Book 類別衍生，例如 Digital Book 和 Physical Book。這允許我們在子類別中重用代碼並實現多態性，即用相同的介面操作不同的對象。
3. 多型（Polymorphism）：這允許我們用一個通用的介面來引用不同類型的對象。例如，我們可以用 Book 類型的引用來處理所有類型的書籍，而在執行時，適當的方法將會被調用，不論它是 Digital Book 或 Physical Book。
4. 抽象（Abstraction）：物件導向允許我們創建抽象的類別和介面，這些抽象只顯示與特定上下文相關的操作和特徵，隱藏了實現的細節。例如，我們可以有一個 User 介面，它定義了所有類型的用戶都應該遵守的行為，而不管它們的具體類型。

使用程序式程式設計來開發這個系統將導致許多問題：

- 程式代碼將會變得冗長和重複，因為相同的代碼需要在多個地方複製和修改。
- 維護和擴展系統將會變得困難，因為改變可能會牽涉到整個代碼庫的廣泛修改。
- 代碼的可讀性和可理解性將大幅下降，因為相關的資料和行為分散在不同的部分。
- 測試各個功能將變得更加困難，因為在程序式設計中，功能往往高度耦合且依賴於全局狀態。

總之，對於一個需要管理複雜資料和實體關係的系統，如線上圖書館管理系統，物件導向程式設計提供了一種更清晰、更有組織的方法來建構和維護系統，相對於程序式設計，它提供了更好的靈活性、可維護性和擴展性。