

CIS212a/CS218 Coursework (2002/2003)
Programming:Advanced Topics and Techniques

Hand out lecture Thursday 14 November 2002

Hand in lecture Thursday 12 December 2002

Note: All coursework must be signed in at the beginning of the lecture.

Group Name

Members *The coursework can be done in groups of at
 most five students*

1. name

2. name

3. name

4. name

5. name

Marks

Hours spent

Note:

CIS212a/CS218 Coursework (2002/2003)

Programming: Advanced Topics and Techniques

Hand out lecture Thursday 14 November 2002

Hand in lecture Thursday 12 December 2002

Note: All coursework must be signed in at the beginning of the lecture.

Objective

This coursework is to improve students' object oriented programming skills in Java language.

Prerequisite

Review all the topics in lecture notes and chapters recommended. Complete all the laboratory Exercises so far.

Assignment

Develop a program in Java that organises a personal collection of books for sale.

The program is required to have the characteristics below:

- There is a graphical user interface for data capture and recall;
- A screen menu displays several options required by the buyer and the seller;
- Some of the options are working programs and some remain to be developed;
- The program must be object-oriented, e.g. functionality and data structures should be modelled as classes;
- All classes should be properly designed. No huge monolithic programs.

The possible options include:

- Search and view books - displaying all or search for some book items;
- Maintain the book list - initialise, add, delete and count the items of the list.

Guideline

A book item can be described by name, authour, publisher, year, size, number of pages, sale price and brief introduction to the book (e.g. in good condition, with authour's signature etc). To simplify the problem, suppose there is only one copy for each book in the collection.

Typically, a potential buyer (customer) may want to view the whole book list. (S)he may want to select a number of books from the list to view the information about condition of the books. (S)he may be interested in a specific class of books, e.g. books about programming or about music, etc.. The seller may wish to add more book items to the list for sale. (S)he may want to check the number of the books on the list, and the total amount of money from the sale, etc..

Hence, the list of books may be designed to be circular, so the customer can see the book items all over again without re-starting the program from the beginning. The program should allow the seller to add records of books to the list. The book list may be sorted by some field such as by authour's family name, or by the price, or by the publishing date.

You may add your own assumptions to simplify the problem. You may also add your own interpretation of what is required in terms of what should be recorded and what options should be available.

Example

A *preliminary* program with *limited* characteristics required may have *primitive* user interfaces below:

(The text after sign '>' is user's input. *Note:* The data used here are random.)

```
Books for Sale
-----
1. View books
2. Initialise bookcase
3. Add books
0. Quit
```

```
Enter your choice > 2
The bookcase is Empty but ready for books.
Press any key to return to Main Menu >
```

```
Books for Sale
-----
1. View books
2. Initialise bookcase
3. Add books
0. Quit
```

```
Enter your choice > 3
book name > Pascal and Java Programming
authour > Ida Pu
publisher > University of London
publication year > 1018
size > A4
pages > 111
sale price > 1000
brief introduction > subject guide
Add more (y/n)? > y
book name > Data Structures and Algorithms
authour > Ida Pu
publisher > University of London
publication year > 1999
size > A4
pages > 111
sale price > 10
brief introduction > subject guide, new
Add more (y/n)? > n
```

Books for Sale

1. View books
2. Initialise bookcase
3. Add books
0. Quit

Enter your choice > 1

book name : Pascal and Java Programming
authour : Ida Pu
publisher : University of London
publication year : 1018
size : A4
pages : 111
sale price : 1000
brief introduction : subject guide
Purchase (y/n)? > n
View the next (y/n)? > y

book name : Data Structures and Algorithms
authour : Ida Pu
publisher : University of London
publication year : 1999
size : A4
pages : 111
sale price : 10
brief introduction : subject guide, new
Purchase (y/n)? > n
View the next (y/n)? > y

book name : Pascal and Java Programming
authour : Ida Pu
publisher : University of London
publication year : 1018
size : A4
pages : 111
sale price : 1000
brief introduction : subject guide
Purchase (y/n)? > y
View the next (y/n)? > y

book name : Data Structures and Algorithms
authour : Ida Pu
publisher : University of London
publication year : 1999
size : A4
pages : 111
sale price : 10
brief introduction : subject guide, new
Purchase (y/n)? > n
View the next (y/n)? > y

book name : Data Structures and Algorithms
authour : Ida Pu

```

publisher      : University of London
publication year : 1999
size           : A4
pages          : 111
sale price     : 10
brief introduction : subject guide, new
Purchase (y/n)? > n
View the next (y/n)? > n
Press any key to return to Main Menu >

```

```

Books for Sale
-----
1. View books
2. Initialise bookcase
3. Add books
0. Quit

```

```

Enter your choice > 0
Bye!

```

Hint

1. You should follow the ‘design guideline’ introduced in the lecture, see the quote below (p22 Carrano and Prichard’s book):
 - (a) Use OOD and TDD together to produce modular solutions. That is, develop abstract data types and algorithms in tandem.
 - (b) Use OOD for problems that primarily involve data.
 - (c) Use TDD to design algorithms for an object’s operations.
 - (d) Consider TDD to design solutions to problems that emphasize algorithms over data.
 - (e) Focus on *what*, not *how*, when designing both ADTs and algorithms.
 - (f) Consider incorporating previously written software components into your design.
2. It might be easier for you to start with developing applications, then change the program from applications to applets.

Submission requirement¹

Your submission of the coursework includes the following two parts, together to be **handed in** at lecture:

1. A *working* program on a **floppy disk** that is ready to be **displayed** on a Web-site, and;
2. A short report *in print*.

¹Fail to meet any of the submission requirements may cause loss of your marks.

Specifically, you should **hand-in** at lecture the followings:

1. **Program disk:** Your program (on the **floppy disk**) has to be able to run remotely via Netscape, although I may mark your program using an appletviewer.

If you would like to display at the same time the working program on your Web-page to gain some bonus marks, an extra section including a reminder and the Web address have to be included in your report.

In any case, a precise instruction on how to run your program has to be presented in your report.

2. **Report in print:** A short report in print should contain the following sections (in the order shown):

- (a) The filled cover page²
- (b) **Introduction** to your approach of solving the problem, including
 - i. Analysis of the given problem
 - ii. A class diagram
 - iii. A list of classes and methods and a brief description of each of their functions
 - iv. A list of *working* classes and a list of classes to be developed
- (c) An example of the **test done** for *one* class of your choice
- (d) Comments on **limitations** of your program
- (e) (*Optional for bonus*) A reminder and your Web-page address for the program³
- (f) A brief **instruction** on how to run your program
- (g) **Attachment:** The complete source codes of your program in Java with sufficient comments, and the name of the floppy disk.
- (h) **References**

Note: No submission after the deadline will be accepted because some submitted solutions are already on the Web then.

²The empty cover page is provided as the first page of the coursework hand-out and you must fill and return it as the first page of your report.

³The source code may be viewed through Netscape, so do not display your program until the deadline.