The University of Melbourne
Grid Computing and Distributed Systems Laboratory
Department of Computer Science and Software Engineering

433-254 Software Design
Second Semester, 2003

Project A

Due Date (5 pm Tuesday, 16th September 2003)

Problem Statement

OzPress, an Australian Publishing House, which has been carrying out all its operations manually, is in the process of computerizing its operations. OzPress publishes and markets printed books and CDs (music in digital media). They also publish pamphlet advertising their products and distributes them to prospective customers. Considering the limited IT staff resources in the press the Management has decided to carry out this project in multiple stages in order to be able to meet the expected completion date with minimal risk. For example, in the 2nd stage they like to Web-enable their application to support e-commerce. To meet these current and future requirements, their system analyst recommended that the software be designed using Object-Oriented methodology. Since OzPress owns computer systems with different CPU architectures and running a variety of operating systems such as Windows, Linux, Solaris, etc., the project manager decided that the software should be implemented using the Java programming language.

High Level Requirements

The model for Project A should be able to handle the following high level requirements.

1. The press should be able to maintain records for all its publications.
2. OzPress publishes 3 types of products: books, CDs, and pamphlets. Each product will have the following information in common.
    a. Title
    b. Price
3. The record for a "book" should have the following information.
    a. A unique book ID – 10 digit characters
    b. Title - string
    c. Author - string
    d. Number of pages - integer
    e. Year of Publication - integer
    f. Category – Science, Engineering, Commerce - string
    g. Price - float

4. The record for a "music CD" should have the following information.
    a. A unique CD ID – 10 digit characters
    b. Title - string
    c. Author - string
    d. Number of songs - integer
    e. Play Duration (in minutes) – float
    f. Year of Publication - integer
    g. Category – Jazz, Pop, Hindi, Gospel - String

h. Price - float

5. The record for a “pamphlet” should have the following information.
    a. A unique CD ID – 10 digit characters
    b. Title - String
    c. Names of all distributors to whom it should be sent – Vector

6. The marketing department needs to maintain sales record for products (books and music CDs) that are sold. For each type of product that are sold for a price, they should maintain the following information:
    a. A unique Item ID - – 10 digit characters
    b. The number of items sold so far. - integer
    c. The number of items sold in each quarter in the current year – array of 4 integers
7. The press should support the following operations.
    a. Publish items and maintain the record of their publications
    b. Sell and/or distribute items and maintain the record of sales
    c. Generating reports of products and sales

Functional Requirements

The requirements specification contains the requirement number (e.g., <ProjA-10>) following by its description of actual requirements.

<ProjA-10> The press shall have three different types of products.

<End ProjA-10>

<ProjA-20> The products such as books and CDs that are sellable will have the price to some amount which is greater than 0.00, whereas non-sellable ones such as pamphlets will have the price set to 0.00.
That is, the products can be sold according to their sale’s price.
<End ProjA-20>

<ProjA-30> The Press shall maintain record of all products and their sells as specified in the high level requirements.
<End ProjA-30>

<ProjA-40> Product ID shall be a unique 10 digit number with the following constraints on the first digit

- Book - 1
- Music CD - 2
- Pamphlet– 3

E.g. 1001200300 is a Book, 2001200400 is a Music CD, and 3002345123 is a Pamphlet.
<End ProjA-40>

<ProjA-50> The following operations shall be supported.

- Add details of a product publication – ADD_PRODUCT
- Add sales data - ADD_SALES

- Individual Item Details Enquiry– PRODUCT_INQUIRY
- Individual Item Sales Enquiry– SALES_INQUIRY
- Generate Report – GENERATE_REPORT

<End ProjA-50>

<ProjA-60> An ADD_PRODUCT operation shall add details of a new product published. Depending on the product type, the required information shall be provided as input parameters. For example, if the product type is Book, the first digit of a product ID will be 1 and the information supplied will be related to the book.

<End ProjA-60>

<ProjA-110> An ADD_SALES operation shall add details of products sold.

Note: This operation is applied to only sellable products such as books and music CD, but not for pamphlets.

<End ProjA-110>

<ProjA-120> A PRODUCT_INQUIRY operation shall report the details of an individual product. The product ID shall be provided as an input parameter.

<End ProjA-120>

<ProjA-121> A SALES_INQUIRY operation shall report the sales details of an individual product. The product ID shall be provided as an input parameter.

<End ProjA-121>

<ProjA-130> A GENERATE_REPORT operation shall report the press products and sales details.

<End ProjA-130>

Input/Output Requirements

<ProjA-300> All input commands shall be redirected from a standard input device (Keyboard) in the case interactive execution or redirected through a file at the command line (e.g., java MyApp < Input_File). All commands shall have a command id (integer starting at 1 and incrementing sequentially followed by the command name (e.g. ADD_PRODUCT) and the arguments for the commands. The command ID basically used to keep track of the status of operation whether it is SUCCEEDED or FAILED. The specific format of the commands is shown in the table below.

- Add details of a product publication – ADD_PRODUCT
- Add sales data - ADD_SALES
- Individual Item Details Enquiry– PRODUCT_INQUIRY
- Individual Item Sales Enquiry– SALES_INQUIRY
- Generate Report – GENERATE_REPORT
- End of Operation - END

Note that each command is represented in a single line. In the explanation some appears to on two lines, but they should actually be treated as a single continuous line. *The input items are separated by commas*. Also, please ignore one or more white spaces that appear between the separator such as colon or comma and the actual item.

Command Name	Command Format
ADD_PRODUCT	**Add a Book:** [CommandId]:ADD_PRODUCT:[ProductID],[BookTitle],[BookAuthor],[NumberOfPages],[PublicationYear],[Category],[Price] E.g. 10:ADD_PRODUCT:1212121212,OO_Programming_in_Java, Raj_Buyya, 400, 2003, Computer_Science, 25.00
	Adding a Music CD: [CommandId]:ADD_PRODUCT:[ProductID],[MusicTitle],[MusicAuthor],[NumberOfSongs], [PlayDuration],[PublicationYear],[Category],[Price] E.g. 10:ADD_PRODUCT:2232421212, I_Love_You, Christ, 5, 25.5, 2002, Gospel, 18.00 11:ADD_PRODUCT:2233421212, Love_U, Madonna, 6, 30.0, 2003, Romantic, 50.00
	Adding a Pamphlet: [CommandId]:ADD_PRODUCT:[ProductID],[Title],[Distributor1],[Distributor2],…, [DistributorN] E.g. 12:ADD_PRODUCT:3245421212, CS_Books, Pearson_Australia 13:ADD_PRODUCT:3235457212, Java_Books_Flyer,RMIT_Bookshop, Melbourne_Bookshop, McGraw_Hill
ADD_SALES	[CommandId]:ADD_SALES:[ProductID],[NoOfItemsSoldSoFar],[Quarter1_SalesCount], [Quarter2_SalesCount], [Quarter3_SalesCount], [Quarter4_SalesCount] E.g. 20: ADD_SALES: 1212121212, 5000, 300, 250, 1000,500 21: ADD_SALES: 2233421212, 3000, 300, 150, 200,500
PRODUCT_INQUIRY	[CommandId]: PRODUCT_INQUIRY:[ProductID] 30: PRODUCT_INQUIRY: 2233421212 **The output will be as follows:** 2233421212, Love_U, Madonna, 6, 30.0, 2003, Romantic, 50.00
SALES_INQUIRY	[CommandId]: SALES_INQUIRY:[ProductID] 40: SALES_INQUIRY: 1212121212 **The output will be as follows:** 1212121212, 5000, 300, 250, 1000,500
GENERATE_REPORT	[CommandId]:GENERATE_REPORT: 50:GENERATE_REPORT: Display all records related to all products and their sales in the same format as entered, but don't include CommandID and Operation ID. **The output will be as follows:** PRODUCTS_REPORT: [Display product1 record as a single line output, but on a new line] [Display product2 record as a single line output, but on a new line] …

	1212121212,OO_Programming_in_Java, Raj_Buyya, 400, 2003, Computer_Science, 25.00 3235457212, Java_Books_Flyer, RMIT_Bookshop, Melbourne_Bookshop, McGraw_Hill … SALES_REPORT: [Display product1 sales record on a new line] [Display product2 sales record on a new line] 1212121212, 5000, 300, 250, 1000,500 2233421212, 3000, 300, 150, 200,500 … **NOTE:** In the above, some items output appeared on two lines due to text wrapping. The output details of each item appear on a single line.
END	Stop processing inputs and end the program execution. [CommandId]:END:

<End ProjA-300>

<ProjA-310> Empty lines and comment lines (beginning with #) in the standard input, shall be ignored.

<End ProjA-310>

<ProjA-320> Output results shall be written to a standard output device (i.e. screen) using the System.out.println() method. This output will them be redirected to a file (e.g., java MyApp > Output_File).

- Successful command ;
  [Command ID] SUCCEEDED

  E.g. 1 SUCCEEDED

- Faild command ;
  [Command ID] FAILED [Reason for failure]

  If the failure is due to a constraint imposed by the requirements the reason for the failure shall be the requirement number.

  E.g. If ADD_PRODUCT is attempted to add a product with for a product ID with type (first digit) is not valid (valid ones are 1-3) or wrong number of arguments, the failure should be reported as,

  2 FAILED ADD_PRODUCT <ProjA-60>

  If the failure is due to a bad command / unrecognized command or a parsing error the failure reported shall be

  [Command ID] FAILED BAD_COMMAND

  For any other failure the reported error shall be

  [Command ID] FAILED OTHER

<End ProjA-320>

<ProjA-330> In the case of a successful PRODUCT_INQUIRY, SALES_INQUIRY, or GENERATE_REPORT command, the system will also display the output on a standard output device.

[The output can also be stored in a file by redirecting to the output file]

<End ProjA-330>

<ProjA-330> In the event a particular command fails, the system shall report the failure as specified in <ProjA-320> and continue to process the rest of the commands until it encounters a command called END.

<End ProjA-330>

<ProjA-340> Your program should be able to recognize bad commands in the input (e.g A command with the wrong number of arguments) and fail the particular command and proceed to the next command. In case all the arguments are provided you can assume that the argument is of the correct data type.

<End ProjA-340>

Use the object oriented design paradigm to develop a Java program to meet the above requirements.

Deliverables:

1. Class diagram for the system showing the classes used and the relationships. Use the object oriented concepts learned so far in your lectures. The class diagram should show
   a. Inheritance relationships
   b. And simple connections between classes in cases where classes communicate with each other

   It is not necessary to show the details (attributes and methods) in the class diagram.
   Note: You will be learning more advanced concepts about class diagrams in your UML lectures but in this case what is expected is a simple diagram which shows the class relationships.

2. Well documented code that implements the above requirements.

3. Assumptions you had to make in the design/coding due to requirements not explicitly being specified.

4. Java code for the system.

Assessment

1. Class Diagram - 2 marks.
2. Java Code - Well documented code should be submitted (8 marks), Marks will be for
   (a) Style,
   (b) Readability,
   (c) Documentation
   (d) Correctness (design, code, and output)
   (e) Use of Object Oriented and Java features (e.g Design for a Large Code Reusability, Classes, Abstract Classes, Inheritance, Encapsulation, Interfaces, class and instance variables, constants, Collections).

Submission

You should submit your class diagram and assumptions you had to make (if you had to make any) in hardcopy form to the submission box marked 254 Project A, in Level 1. Please make sure the name, login and student number are clearly written in the document, especially on the front page/cover.

All your source files (.java files) along with Keyboard.java if utilised its methods in your program, should be submitted using the submit command:

submit 254 A [File Names]

e.g. submit 254 A OzPres.java Book.java Keyboard.java ……

Please make sure that the entry point of your program (main) is in a file called OzPress.java
i.e. your program will be invoked with the command

java OzPress < Input_command_file_name > Output_file_name

You can submit as many java files as you like but make sure that your main program is OzPress.java or else the project will fail automated testing.

You should then verify the submission using the command:

Verify 254 A

It is important to ensure your system can interpret the commands stated in the requirements.

Late submissions should be sent to A.late using the above commands. Late submissions will be penalized at 1 mark per day late.

Note on the Output:

It should be noted that the project outputs need to be very specific and students should follow the given templates very carefully and produce the exact same output, so that the system functionality can be verified automatically. Also, students are responsible for testing their system and making sure that it works according to the specs and submit it properly.

Individual Work

You are reminded that all submitted work in this subject is to be your own individual work. Students submitting the work of others for assessment will be penalized by the Department, and risk prosecution under the University's Discipline Regulations. Students who allow other students access to their work will be penalized, even if they themselves are sole authors of the program in question. Automated similarity checking will be used to compare submissions.

Questions and further Information

The subject Web page will be kept updated with any further information relevant to the project and will be considered as part of the specification for the project. Questions about the project specification should be directed to the newsgroup cs.254. Even with these resources, note that there may still be some cases where there is ambiguity about how the program should behave. In such situations, you should make a sensible assumption and justify and document it.