Hacettepe University
Computer Science and Engineering Department

Advisers	:	RA Ali ÇAĞLAYAN and RA Kazım SARIKAYA {alicaglayan,kazimsarikaya}@cs.hacettepe.edu.tr
	:	Dr. Sevil ŞEN ssen@cs.hacettepe.edu.tr
Course	:	BİL138 Programming Lab. II
Experiment No	:	2
Subject	:	Inheritance
Deadline	:	First Submit: 27.03.2012 16:59 Second Submit: 03.04.2012 16:59

Contents

1 Introduction

Object-oriented programming has advantages such as modeling problems with less complexity and more code reuse. In this experiment, you will observe these advantages by using inheritance mechanism which is an important property of object-oriented programming. By the help of this experiment, you will learn the concept of inheritance, relationships among classes by using object references, control of multiple instances of classes, package structures in Java.

2 Useful Information

Under this section you will find useful information about the project. They are at the beginning level. For advanced information you need to make additional research.

2.1 Inheritance

Object-oriented programming (OOP) covers software in terms similar to those that people use to describe real-world objects. It takes advantage of class relationships, where objects of a certain class, such as a class of vehicles, have the same characteristics cars, trucks, little red wagons and roller skates have much in common. Inheritance is one of important property of OOP. OOP takes advantage of inheritance relationships, where new classes of objects are derived by absorbing characteristics of existing classes and adding unique characteristics of their own. In Java, a class(called the **derived class** or **subclass**) extends from another class(called the **base class** or **superclass**).

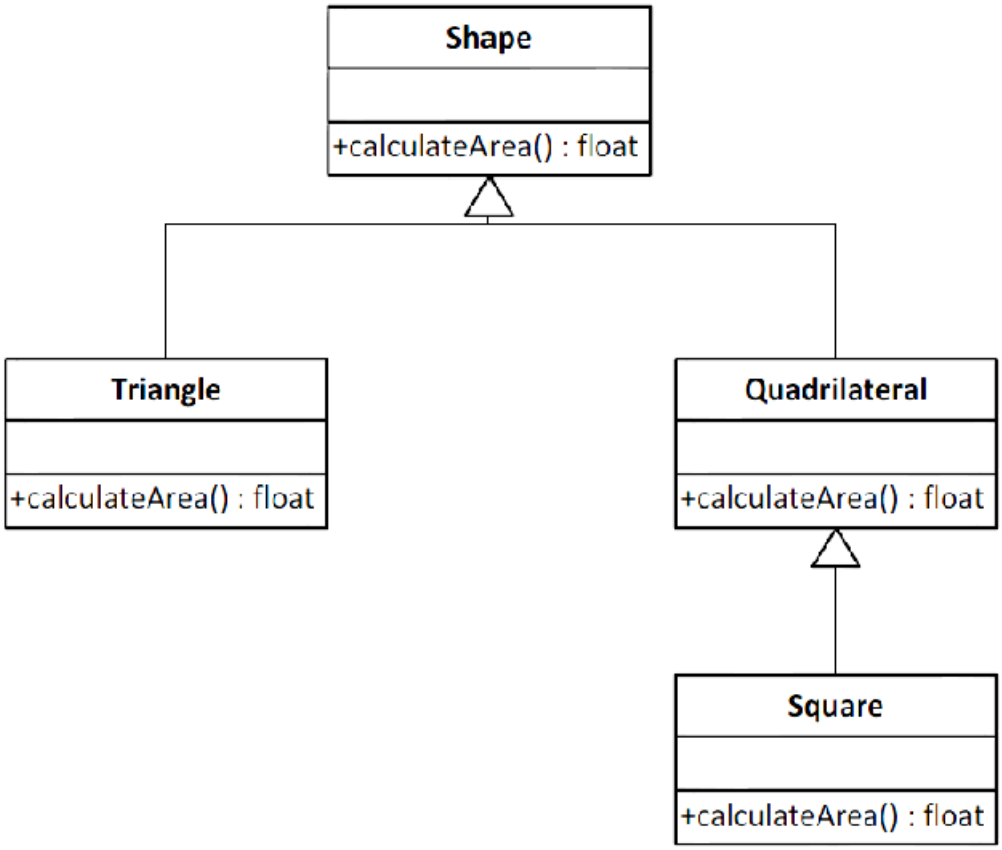

Figure 1: A hierarchy of shape classes

In Fig. 1 a shape hierarchy is seen. **Shape** class is superclass of all the other classes. **Square** class is a subclass of **Quadrilateral**.

2.2 Method Overriding

When a class extend another class, the subclass can use the super class' methods. However sometimes the subclass should change behavior of a method which provided by superclass. The method implementation in the subclass overrides(replaces) the implementation in the superclass. The subclass method and superclass method have the same name, parameters and the same return type. That is called **method overriding**. In Fig. 1 **calculateArea()** method overrides in the each subclass. The method behaves different in each subclass.

2.3 Java Packages

Java packages are the way to organize files into different directories according to their functionality and usability. It is very important especially when we work on a big project. It makes easier to find and use a class and to avoid naming conflicts. The classes in the same packages have the same directory path.

3 Experiment

This section contains four subsections. The first one is about the problem, and the second one is about the report. Second section describes how you will write your report. The third one is the about **constraints** that you should obey. Otherwise your experiment will not be evaluated by your adviser RA Ali ÇAĞLAYAN and RA Kazım SARIKAYA. The last one is about the submission.

3.1 Problem Definition

In this experiment you should develop a simple Library System with Java using extension. You will also use Arrays in this experiment. The library details are given below.

If a class has a attribute named **X**, the class should implements **getter** and **setter** methods of **X** attribute. You should use proper access keywords for attributes and methods.

You should compare each class with the same type of class with comparing all attributes defined on that class.

If an attribute of the class have a constraint then, if constraint can not be satisfied then you should print the error message with the format: "**THE CONSTRAINT OF** $<$ *attribute_name* $>$ **OF THE** $<$ *class_name* $>$ **IS NOT SATISFIED**".

Any **person** can use the library. A person should have the attributes **name**, **surname** and **age**. The attribute age should be greater then **0**.

Inside the library there are **publication**s. A publication should have the attributes **name** that describes the name of the publication and **location** that describes location of publication inside the library. The library could have several copies of the the same publication.

A publication could be **periodical** or **non-periodical**.

A periodical publication have the attributes **period** that should be one of **DAILY**, **WEEKLY**, **MONTHLY** that describes the publish period of the publication. You should check the values could be assigned this attribute inside the proper method. Also do not forget that values of this attribute are case sensitive. The class should have the attribute **publicationDate** that is an instance of **java.util.Date** that describes which date the publishing is made. Periodical publications should have a special attribute named as **issn**.

A non-periodical publication should have the publication **year** attribute between **1900** to **2012**. Also the non periodical publications could have published more then one, if all copies are sold. Hence there should be **edition** attribute that is greater then **0**. This constraint should be checked inside the proper method. Non periodical publications should have a special attribute named as **isbn**.

The library contains **newspaper** and **magazine** publications which are published periodically. The newspapers should have **heading** and **subheading** attributes. The library sells the newspapers hence the newspapers have **price** attribute. The magazines should have the attributes **page count** that should be greater then **0** (hence the constraint should be checked inside the proper method) and **category** of the magazine.

The library contains non periodical publications **book**s, **encyclopedia**s and **DVD**s. Books should have one **author** who is a person. However encyclopedias could have more then one author. Books and encyclopedias should have **page count** information. Encyclopedias could have volumes, hence each encyclopedia should have **volume no** and **volume count** information. DVDs should have the attributes **category** - such as music, documentary film, movie etc - and **duration** of the DVD.

The classes that describes entities of the library should be placed inside the package **exp2.model**.

3.1.1 ArrayBean Classes

An ArrayBean class should have an attribute with type array of a given class. This attribute should not have a **setter** method. The class should have a create method with void return type for adding an instance of the given class. Hence the method should have one parameter with the type given class. The class should have a delete method with boolean return type for deleting a given instance of the given class with parameter of this method. If instance is not inside the array, the method should return false otherwise true.

For newspapers, magazines, books, encyclopedias, DVDs there should be ArrayBean class. This classes should be inside the package **exp2.ab**.

3.1.2 BorrowedBookInfo and BorrowedBookInfoArrayBean Class

The class **BorrowedBookInfo** should have **Book** attribute named as **borrowedBook**, **Person** attribute named as **borrowedBy**, **java.util.Date** attribute named as **borrowDate**, integer attribute **borrowedFor** which describes the duration of borrowing.

This class should be inside the package **exp2.model**.

For this class there should be an ArrayBean class and it should be inside the package **exp2.ab**.

3.1.3 LibraryManager Class

The **LibraryManager** has main operations for Library system. The class should have attributes with the type of ArrayBean classes.

The class should have a method named as **searchBook** which takes a **String** parameter for book name. The returning type of method is **Book**. The method searches the name of the book inside related array bean. If there is any book for that name the method should return the Book instance otherwise **null**.

The class should have a method named as **loanBook** which has parameters book with type **Book** and duration with type integer, person with type **Person**. The method should delete the given book from related array bean and create an instance of **BorrowedBookInfo**. The date of borrowing should be set as current Date. And the instance should add the related array bean. The return type of method should be boolean. If the book can be borrowed, the method should return true otherwise false.

The class should have a method named as **retrieveBook** which has a parameter book with the type **Book**. The return type of method should be float. The method should search the book inside the related array bean and compare the the current date with deadline of borrowing. If current date is less or equal the deadline the method should return 0, otherwise the method should return $1.5 for each first seven day, then $3 for each day of next month, then $10 for each other day; if the retrieving is delayed. The retrieved book should append the related array bean.

The class should have a method named as **listPublications** which prints all not borrowed publications to the console with using publication's proper method. For each instance of publications you should print entity type such as book, newspaper etc. with first letter uppercase. Then the following lines you should write each attribute and attribute value line by line with the format

$< attribute\_name > = < attribute\_value >$. The attribute names should be in **lexicographical order**.

The class should have a method named as **listBorrowedBooks** which prints **books** to the console with using book's proper method. For each instance of publications you should print entity type such as book, newspaper etc. with first letter uppercase. Then the following lines you should write each attribute and attribute value line by line with the format $< attribute\_name > = < attribute\_value >$. The attribute names should be in **lexicographical order**.

This class should be inside the package **exp2.model**.

3.2 Report

The structure of report is described below:

- **Description of problem**, describe problem with your **own** sentences.
- **Class diagram and class definition**, describe details of all the classes and draw class diagram. Show attributes and method names in your class diagram.
- **Algorithm**, give the algorithms of the methods used in **LibraryManager** class.

3.3 Constraints

You should obey these constraints. Otherwise your experiment will not be evaluated as well as you expected.

1. The classes' and attributes' names should be satisfied Java naming convention.
2. You should model entities of library with classes.
3. You are responsible for a correct model design of the library system. Your design should be accurate.
4. You should use inheritance mechanism.
5. You should use method overriding.
6. You should use Java package structures.

3.4 Submission

The submissions will be accepted only by `http://submit.cs.hacettepe.edu.tr`. The experiment should be submitted with two parts. The first submit should be done until 27/03/2012, and second submit should be done until 03/04/2012.

At first submit you should submit the java files inside proper folders

- The classes described in the section 3.1(not subsections).
- ArrayBean classses described in the section 3.1.1.

inside **src** folder.

At the second submit you should submit the java files inside proper folders

- exp2.model.BorrowedBookInfo
- exp2.ab.BorrowedBookInfoArrayBean
- exp2.model.LibraryManager

inside **src** folder and **report.pdf** inside **report** folder.

4 Grading

Your Experiment results will be evaluated with the following rules. They are fixed. They will never be changed. Hence obey these rules. Before you contest your grade re-evaluate yourself with these rules.

- Java naming convention is over 5 points.
- The accurate model is over 25 points.
- Inheritance is over 10 points.
- Overriding is over 10 points.
- The class LibraryManager evaluated over 20 points: 4 points for each 5 methods: **searchBook**, **loanBook**, **retrieveBook**, **listPublications**, **listBorrowedBooks**.
- The submit format have 2 points: 1 point for first submit and 1 point for second submit.
- The report have 28 points: 5 points for **Description of problem**, 15 points for **Class diagram and class definition** and 8 points for **Algorithm**.

5 Notes

These notes are very important please read them and obey them.

- **No submissions except Submission System of Computer Engineering Department will be accepted. Do not send your experiment with emails.**
- **Do not miss the deadline. No submissions will be accepted after First Submit: 27.03.2012 16:59 Second Submit: 03.04.2012 16:59.**
- Respect your adviser RA Ali ÇAĞLAYAN and RA Kazım SARIKAYA office hours: Wednesday, 14:00-16:00 (RA A. ÇAĞLAYAN) and Tuesday, 15:00-16:45 (RA K. SARIKAYA).
- For every question or problem please contact with your adviser RA Ali ÇAĞLAYAN and RA Kazım SARIKAYA. Do not contact with teacher of experiment directly.
- You can send email to your adviser's email {alicaglayan,kazimsarikaya}@cs.hacettepe.edu.tr.
- **Cheating is forbidden. Collaboration is forbidden. Both of them will be graded with F3**.
- Use understandable names for your variables, procedures and functions.
- Obey general software engineering principles.
- Save all your work until the experiment is graded
- **First do research, and then ask! RAs are not Google search engine!**