中山大学软件学院 **2008** 级软件工程专业（2010 学年春季学期）

《SE-305 系统分析与设计》期末考试试卷（B）

(考试形式：开　卷　　考试时间：2 小时)

警示

《中山大学授予学士学位工作细则》第六条

考试作弊不授予学士学位

方向：_____　　姓名：_____　　学号：_____

A. Choose the correct answer（20 points, 2 points each）

1. How many statements are true?
   (1) The Domain Analysis helps us in finding concepts for Domain Model, and candidate methods for our classes.
   (2) Design emphasizes a solution (in software and hardware) and its implementation.
   (3) Divide and Conquer is a common methodology in solving complex problems.
   Answers: A 0　B 1　C 2　D 3
2. How many statements are true?
   (1) The observer pattern is a structural pattern.
   (2) An abstract class has none of implementation at all.
   (3) Inception will take six or more weeks long.
   Answers: A 0　B 1　C 2　D 3
3. How many statements are true?
   (1) An object is an instant of its class/type.
   (2) In a Design Class Diagram, inheritance relation can be used to represent a "has-a" relation.
   (3) Domain Analysis is often applied before Use Case Analysis in building software of a complex domain, but after of a simple one.
   Answers: A 0　B 1　C 2　D 3
4. How many statements are true?
   (1) Analysis, Design, Coding, Testing are the four phases in a Unified Process.
   (2) Both the decorate pattern and proxy pattern using delegation.
   (3) When one class inherits from another, then polymorphism allows a superclass to stand in for the subclass.
   Answers: A 0　B 1　C 2　D 3
5. How many statements are true?
   (1) Instead of avoiding changes, embracing changes and refactoring existing design/code is one of best practices in contemporary software design.
   (2) The Marco Command is a combination of the decorator and command pattern.
   (3) We shouldn't code to an interface instead to an implementation.
   Answers: A 0　B 1　C 2　D 3
6. How many statements are true?
   (1) Elaboration is to fully and carefully define models which are translated into code during

construction.

(2) Assigning responsibilities is the main task of object-oriented design.

(3) The Mythical Man-Month: Essays on Software Engineering is a popular book on Object-Oriented programming which published recently.

Answers: A 0 B 1 C 2 D 3

7. How many statements are true?

(1) We take two or three months to acquire to most of requirements for supporting analysis and design.

(2) Elaboration is not the requirements or design phase; rather, it is a phase where the core architecture is iteratively implemented, and high-risk issues are mitigated

(3) There are analysis, design, coding and testing activities in each iteration.

Answers: A 0 B 1 C 2 D 3

8. How many statements are true?

(1) Java standard editions before version 5 allow two methods within a class different only on their return type.

(2) Composition relationships are shown with a white diamond attached to the composing class.

(3) Some objects have unique identity, they are not equal even all their attributes are equal.

Answers: A 0 B 1 C 2 D 3

9. How many statements are true?

(1) Design principles are not really implementable in code, they are just guidelines in our design practices.

(2) The Unified Modeling Language (UML) is a visual language for specifying, constructing and documenting the artifacts of systems.

(3) A common practice in software engineering is validating the result of each working steps.

Answers: A 0 B 1 C 2 D 3

10. How many statements are true?

(1) The Law of Demeter principle is another name of the Least Knowledge Principle, which means “Talk only to your immediate friends”.

(2) Every object in your system should have a single responsibility, and all the object’s services should be focused on carrying it out.

(3) A class is a blueprint for an object, where the attributes (aka. instance variables) and methods of the object specified.

Answers: A 0 B 1 C 2 D 3

B. Short Answer（30 points, 10 points each）

1. Test Driven Development (TDD) is an important approach for developing high quality software.
   (1) Who proposed TDD firstly?
   (2) What is the most famous tools family supporting the TDD process?
   (3) Describe the motivation of TDD, and explain effectiveness of TDD in agile development briefly.

2. MVC model is one of the most important architecture patterns.
   (1) What means are “M”,”V”, and “C”?
   (2) Explain the intent of the MVC model in design.
   (3) Analysis advantage and disadvantage of the MVC model

3. Frameworks and libraries are common assets in software reuse.
   (1) Give an example for each of them, and what’s the different between them?
   (2) Describe the developer’ usage of the framework at the developing time.
   (3) Describe the interaction between the developer’s code and the framework’s code at the runtime.

C. Code & UML （30 points, 15 points each）

1. **Struts Action**: The following code is a part of a Web application based on the Struts framework. When a user post a form from her browser, the request will be handled by the execute method of the EnterItemAction on the server.

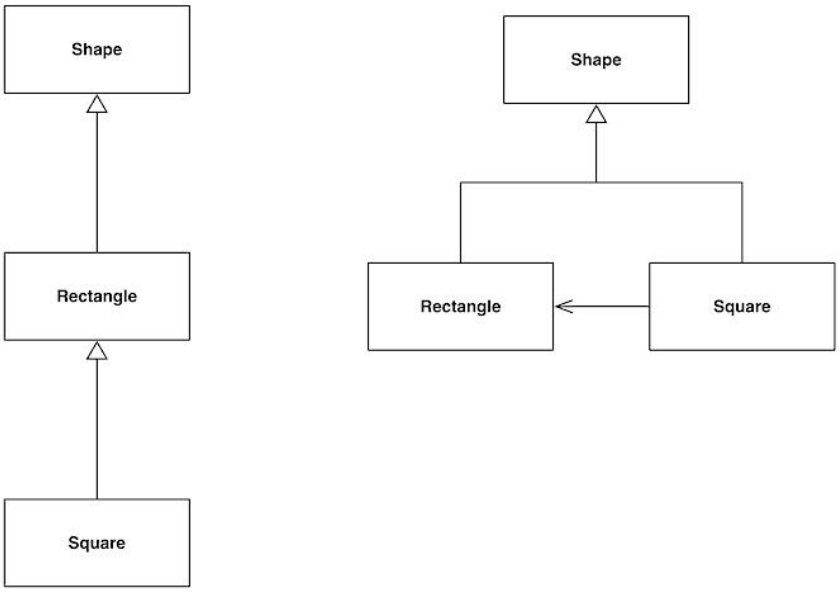
```
==================== code begin ======================
package com.craiglarman.nextgen.ui.web;
// … imports

    // in Struts, an Action object is associated with a
    // web browser button click, and invoked (on the server)
    // when the button is clicked.
public class EnterItemAction extends Action {

    // this is the method invoked on the server
    // when the button is clicked on the client browser
public ActionForward execute( ActionMapping mapping,
                              ActionForm form,
                              HttpServletRequest request,
                              HttpServletResponse response )
                          throws Exception
{

        // the server has a Repository object that
```

```
            // holds references to several things, including
            // the POS "register" object
        Repository repository = (Repository)getServlet().
            getServletContext().getAttribute(Constants.REPOSITORY_KEY);

    Register register = repository.getRegister();

            // extract the itemID and qty from the web form
        String txtId = ((SaleForm)form).getItemID();
        String txtQty = ((SaleForm)form).getQuantity();

            // Transformer is a utility class to
            // transform Strings to other data types
        ItemID id = Transformer.toItemID(txtId);
        int qty = Transformer.toInt(txtQty);

            // here we cross the boundary from the
            // UI layer to the domain layer
            // delegate to the 'domain controller'
            // > > > THIS IS THE KEY STATEMENT < < <

        register.enterItem(id, qty);

        // …
    } // end of method
    } // end of class
=================== code end ======================
```

(1) Draw a UML class diagram for the code, and explain what kind of software architecture the code implements by marking architectural roles of these classes on the diagram and then describing their responsibilities in text.

(2) Draw a UML sequence diagram for the code, illustrating interactions among classes when a user posts a form from her browser.

2. Square question: The figure below contains two class diagrams showing different ways to arrange three classes: Shape, Rectangle, and Square.



(1) Update class diagrams according to the requirement: each shape has a width and a height, and provide its area to clients. Provide a discussion on the two designs. Which one is better, and why?

(2) Implement the design in an Object-Oriented programming language, and demonstrate the advantage of your preferred design in code.

D. Software Design （**20 points**）

1. **Adventure Game**: The software designer of an adventure game wants a player to be able to take or drop various items. These items can be found in rooms of the game. Beside individual items, such as diamonds, weapons, tools, etc, there are bags and boxes which may contain other items or other bags and boxes. Bags and boxes can be opened and closed and items can be added to or removed from a bag or box.

Tasks for this question:

(1) Find conceptual classes in the problem domain. Prepare a domain model (or analysis class diagram).

(2) Use the most appropriate design pattern to address the problem and show how it is applied. In particular, show an appropriate class diagram(s) and enough code fragments to illustrate your use of the pattern to solve the problem.