INSTITUTE OF TECHNOLOGY BLANCHARDSTOWN

NATIONAL DIPLOMA IN COMPUTING
(Information Technology)

Object Orientation with Design Patterns
Module code here

Semester I

Internal Examiner(s): Stephen Sheridan

External Examiner(s): Dr Gerard Parr,
Dr Micheal O'hEigeartaigh

January 2002
Time of examination here

Instructions to candidates:

1) **Section A: Five parts to be attempted. Section B: Answer any 3 Questions.**
2) **All questions carry equal marks.**

DO NOT TURN OVER THIS PAGE UNTIL YOU ARE TOLD TO DO SO

(Section A: Answer any 5 parts of Question 1)

Question 1

a) Define the phrase "Design Pattern".

[5 Marks]

b) Design patterns can fall into one of three categories. What are these categories and how do they differ?

[5 Marks]

c) Graphical representations of design patterns only capture the end product of the design process. Why? List and briefly describe four essential elements that can be used to describe a design pattern.

[5 Marks]

d) What is ***reflection*** and how can it be used to create dynamic adapters.

[5 Marks]

e) Briefly describe the difference between the ***Simple Factory Pattern*** and the ***Abstract Factory Pattern***.

[5 Marks]

f) Describe with the aid of a code sample how you can easily determine that you are dealing with two identical instances of a Flyweight class.

[5 Marks]

g) Briefly describe the *Template Pattern* and the four types of methods that can be found in a template class.

[5 Marks]

[Total Marks 25]

(Section B : 3 Questions to be Attempted)

Question 2

a) Creating objects by using a class name directly can lead to problems. What are these problems and how can they be overcome? Give a simple example.

[6 Marks]

b) With the aid of some sample code, describe the difference between ***shallow copies*** and ***deep copies*** of objects.

[8 Marks]

c) Describe in detail how ***static data members*** and ***methods*** can be used to create the *Singleton pattern*.

[11 Marks]
[Total Marks 25]

Question 3

a) The *Adapter Pattern* allows unrelated classes to work together in a program. Name the two mechanisms that can be used to do this and briefly describe both approaches. Using code samples show one case where the adapter pattern is used in Java.

[6 Marks]

b) The *Composite Pattern* allows the programmer to define a class hierarchy of simple objects and more complex objects so that they appear to be the same to the client program. With the aid of some code samples describe one such situation where this behaviour might be useful.

[8 Marks]

c) The program given in ***code listing 1*** uses a simple *Decorator Pattern* to create a cool button. A cool button is a button for which borders appear only when the user moves the mouse over the button.

Describe in detail the role each class plays in order to implement the *Decorator Pattern*, along with an explanation of each of the class methods.

[11 Marks]
[Total Marks 25]

Question 4

a) The *Chain of Responsibility Pattern* helps to keep separate the knowledge of what each object in a program can do. That is it reduces the coupling between objects so that they can act independently. Describe four situations where this pattern might be used.

[4 Marks]

b) The program given in **code listing 2** creates a simple user interface that allows the user to select menu items, File|Open and File|Exit, and click on a button labelled Blue that turns the background of the window blue.

As long as there are only a few menu items and buttons this approach works fine, but when there are several menu items and buttons the *actionPerformed* code can get pretty unwieldy.

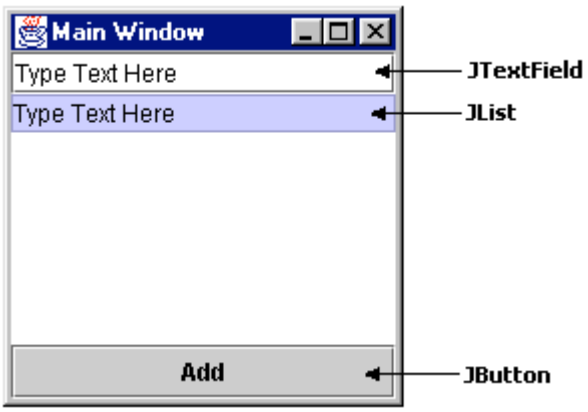
Using a simple *Command Pattern* re-write the appropriate sections of this program so that the conditional block in the *actionPerformed* code is removed.

[10 Marks]

c) Write a program that allows the user to add items to a JList control by typing text into a JTextField and clicking on an add button. Your program should take advantage of the Model View Controller (MVC) architecture.

In developing your program you must incorporate the following classes

- **MainWindow** extends JFrame
- **ListData** extends AbstractListModel



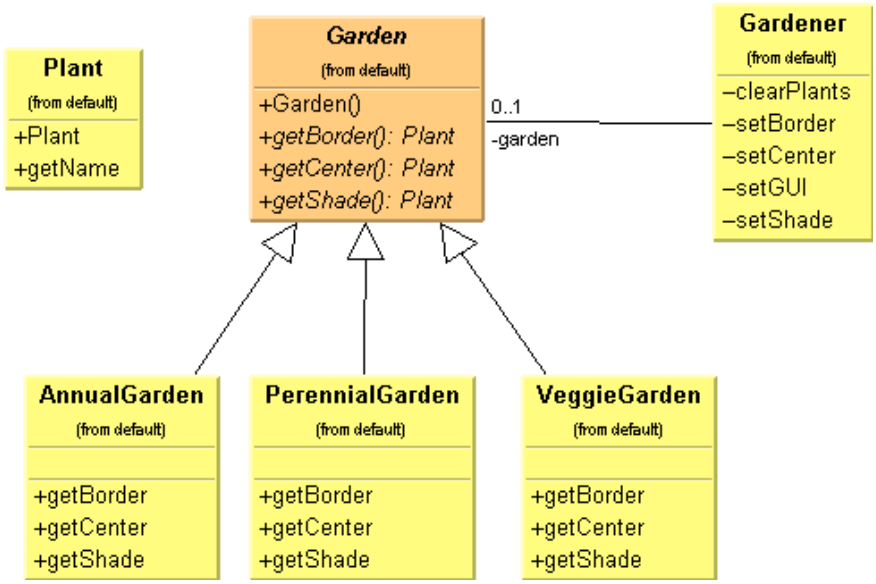
[11 Marks]
[Total Marks 25]

Question 5

a) Define the terms ***Observer*** and ***Subject*** with regard to the Observer Pattern. Write an abstract class called Observer and an abstract class called Subject that could be used to implement the Observer Pattern.

[6 Marks]

b) The following UML class diagram describes a commonly used creational pattern. Name the pattern and describe how each of the classes in the diagram contributes to the patterns implementation.



[8 Marks]

c) The Iterator Pattern is one of the simplest and most frequently used of the design patterns. Java supports the Iterator pattern by providing Enumerations for its Vector and Hashtable classes.

Given a Vector containing a list of names, write a class which implements the Enumeration interface so as to provide a filter that will only iterate through names that begin with some prefix.

So, for example, a test program for the Filter class might look as follows:

```
class MainApp
   {
      private Vector data;

      public MainApp()
      {
            data = new Vector();
            data.addElement("Alan");
            data.addElement("Conor");
            data.addElement("Joanne");
            data.addElement("David");
            data.addElement("John");
            data.addElement("Martin");
      }

      public void filterNames()
      {
            Filter filter = new Filter(data.elements(), "Jo");
            while(filter.hasMoreElements())
            {
                  String s = (String)filter.nextElement();
                  System.out.println(s);
            }
      }

      public static void main(String[] args)
      {
            MainApp app = new MainApp();
            app.filterNames();
      }
   }
```

[11 Marks]
[Total Marks 25]