INSTITUTE OF TECHNOLOGY BLANCHARDSTOWN

Year	**Year 3**
Semester	**Semester 1**
Date of Examination	
Time of Examination	

Prog Code	**BN013**	**Prog Title**	**Bachelor of Science in Computing**	**Module Code**	**COMP H3012**
Prog Code	**BN104**	**Prog Title**	**Bachelor of Science (Honours) in Computing]**	**Module Code**	**COMP H3012**
Prog Code	**BN302**	**Prog Title**	**Bachelor of Science in Computing in Information Technology**	**Module Code**	**COMP H3012**

Module Title	**Object Orientation with Design Patterns**

Internal Examiner(s): *Ms. Orla McMahon*
External Examiner(s): *Mr. John Dunnion, Dr. Richard Studdert*

Instructions to candidates:

1) **To ensure that you take the correct examination, please check that the module and programme which you are following is listed in the tables above.**
2) **Section A: Attempt any five parts.**
3) **Section B: Answer any 3 Questions.**

4) **All questions carry equal marks.**

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

Section A

Attempt any 5 parts of this question (5 marks each)

Question 1

a) Design patterns were discovered by the Gang of Four.
Briefly describe what led to the discovery of design patterns and why they are so important in software projects.

[5 Marks]

b) 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]

c) Briefly describe how the Proxy Design Pattern works and give one situations where it might be used.

[5 Marks]

d) Define the intent of the Builder pattern.
Illustrate your answer with an example.

e) Define the intent of the Observer pattern.
List one consequence of applying this pattern.

[5 Marks]

f) With the aid of a UML diagram, describe the components of the **Composite** design pattern

[5 Marks]

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

[5 Marks]

(Total Marks 25)

Section B
Candidates should attempt any 3 of the following questions.

Question 2

Read the following case study and answer the questions that follow:

Case Study
You are currently running a construction company which is in the business of building houses.
The types of houses built are bungalows, two-storey Georgian and two-storey dormer.
The windows in the houses have various designs including arched, rectangular and circular.

The systems architect would like to create a system that will plan the design of the various houses and the windows required for each house.

With the aid of a UML diagram, illustrate how you could implement both the simple factory pattern and the abstract factory pattern using the above case study.
For each example include a full explanation describing how the given pattern will be deployed.

[16 Marks]

In order to implement the factory method pattern, the case study should include additional information.
Describe what additions could be made to the case study.
Using a UML diagram, explain how the factory method pattern could be implemented.

Hint
There is an online stock system available that provides details about the windows in stock.

[9 Marks]

(Total Marks 25)

Question 3

a) Even though the Singleton pattern is grouped with other creational patterns, why to some extent is it different?

[4 Marks]

b) Consider a system for processing ATM card transactions. A thief is using a stolen ATM card number to steal funds, concurrent with the legitimate user's withdrawing funds. How could you design a Singleton to reduce this risk?

[9 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();
}
}
```

[12 Marks]

(Total Marks 25)

Question 4

a) In general a pattern has four essential elements.

Identify and describe the four essential elements.

[4 Marks]

b) A common pattern cited in early literature on programming frameworks is the Model-View-Controller (MVC) pattern.

Briefly describe with the aid of an example the role of the various participants in the MVC pattern.

[5 Marks]

c) When someone enters an incorrect value in a cell of a JTable, you might want to change the colour of the row to indicate the problem.

Identify a design pattern that could be used for this purpose.
Describe in detail how this design pattern could solve the given request.

[6 Marks]

d) When you build a Java user interface, you provide controls – menu items, buttons, check boxes, and so on – to allow the user to tell the program what to do. When a user selects one of these controls, the program receives an **ActionEvent** which it must trap by implementing the **ActionListener** interfaces (actionPerformed). This code can get quite cumbersome if there are many controls that make up the user interface.

Describe with the aid of some sample code one method that can be used to reduce the amount of coding in the **actionPerformed** method by forwarding specific commands to specific user interface controls.

[10 Marks]

(Total Marks 25)

Question 5

a) Explain using a simple example how the **Facade** pattern can simplify the use of a complex system for clients.

[5 Marks]

b) What is the intent of the **Chain of Responsibility** pattern?

Describe two consequences of applying the **Chain of Responsibility** pattern to a software design problem.

[8 Marks]

c) The following example allows the user to display points, lines and squares.
In designing the system, the designer decided to include these shapes in a higher level concept that could be called a 'displayable shape'.
To accomplish this, the designer created a shape class and then derived from it the classes that represented points, lines and squares (see fig below).

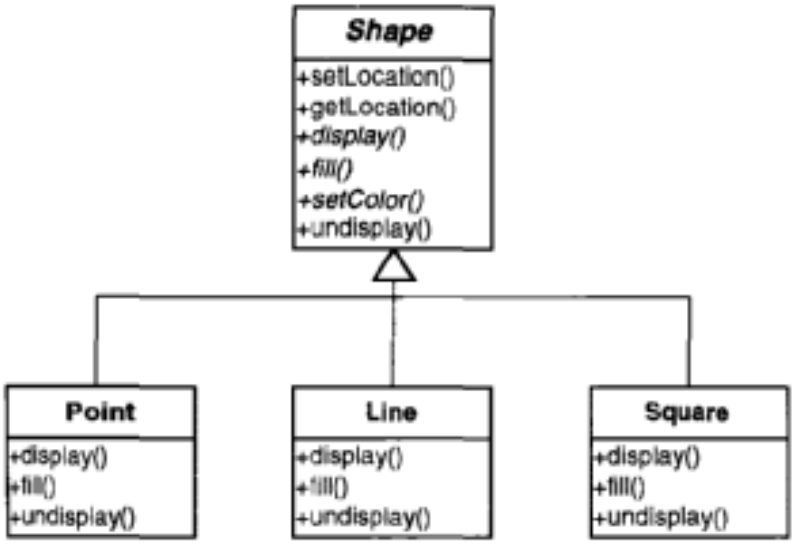

Figure 7-3 Points, Lines, and Squares showing methods.

After implementing the above system, the designer was then asked to include a class that could draw triangles. This class should also contain all of the required methods such as display(), fill() and undisplay().
The designer then discovered that a colleague had written a triangle class that perfomed the exact tasks required. Unfortunately the methods within this class were called displayIt(), fillIt(), undisplayIt().

I. Using the **Adapter** pattern, explain in a step-by-step fashion how you would incorporate the given 'triangle' class into the above system.

[4 Marks]

II. Draw an accurate UML diagram that reflects the updated system.

[4 Marks]

III. Provide a Java code fragment for the class 'triangle' that implements the **Adapter** pattern.

[4 Marks]

(Total Marks 25)