INSTITUTE OF TECHNOLOGY BLANCHARDSTOWN

BACHELOR OF SCIENCE IN COMPUTING
(Information Technology)

Object Orientation with Design Patterns
CM302??

Semester I

Internal Examiner(s):	Ms. Orla McMahon
External Examiner(s):	Mr John Dunnion Prof. Gerard Parr

August 2006
Time of examination here

Instructions to candidates:

1) Section A: Attempt any five parts.
2) Section B: Answer any 3 Questions.
3) 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

Jan 2005

a) Describe briefly what is meant by the term 'Design Pattern' and in particular why they are used in software projects.

[5 Marks]

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

Briefly describe the role of the various participants in the MVC pattern.

[5 Marks]

c) What is the intent of the Command pattern?

Illustrate your answer with a simple example.

[5 Marks]

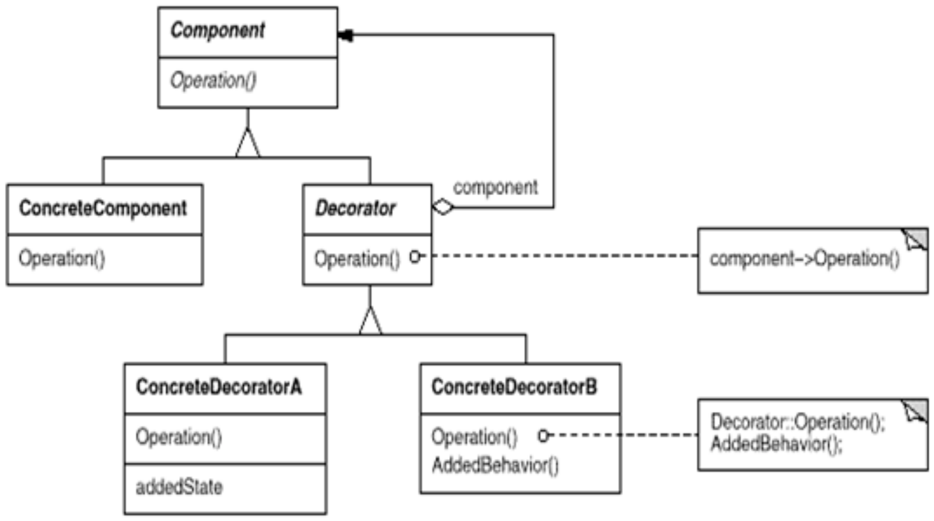
d) Describe with the aid of a code sample how you can easily determine that you are dealing with two identical instances of a Flyweight class.

[5 Marks]
(Aug. 2004 d)

[5 Marks]

Question 1 (Contd.)

e) Given the following UML diagram, explain briefly the role of each participant in the **Decorator** pattern.



[5 Marks]

f) Design patterns can fall into one of three categories.
What are these categories?
Provide a brief description of each category and **two** examples of design patterns that fall within each category.

[5 Marks]

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

[5 Marks]
(Jan. 2004 f)

(Total Marks 25)

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

Question 2 (Jan 2004)

a) In general a pattern has four essential elements.

Identify and describe the four essential elements**.**

[4 Marks]

b) Use an intuitive example to explain the intent of the Flyweight design pattern.

[6 Marks]

c) The easiest way to create a class that can have only one instance is to embed a static variable inside of the class that is set on the first instance and then check for it each time that you enter the constructor.

i) Write a Java program that implements the statements above.

[7 Marks]

ii) Write a Java program that tests this implementation.

[4 Marks]

iii) What design pattern have you implemented?

[1 Marks]

iv) Using **two** real world examples, describe why you might use this pattern.

[3 Marks]

(Total Marks 25)

Question 3 (Jan 2005)

c) Use an intuitive example to explain the intent of the Composite design pattern.

[6 Marks]

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

[6 Marks]

e) The program code in code listing 2 uses a Flyweight pattern to create folders. Basically the system draws an icon for each folder with each person's name displayed under the folder.

Describe in detail the role each class plays in order to implement the Flyweight pattern. Within each class, provide an explanation of the class methods.

[13 Marks]

(Total Marks 25)

Code Listing 2

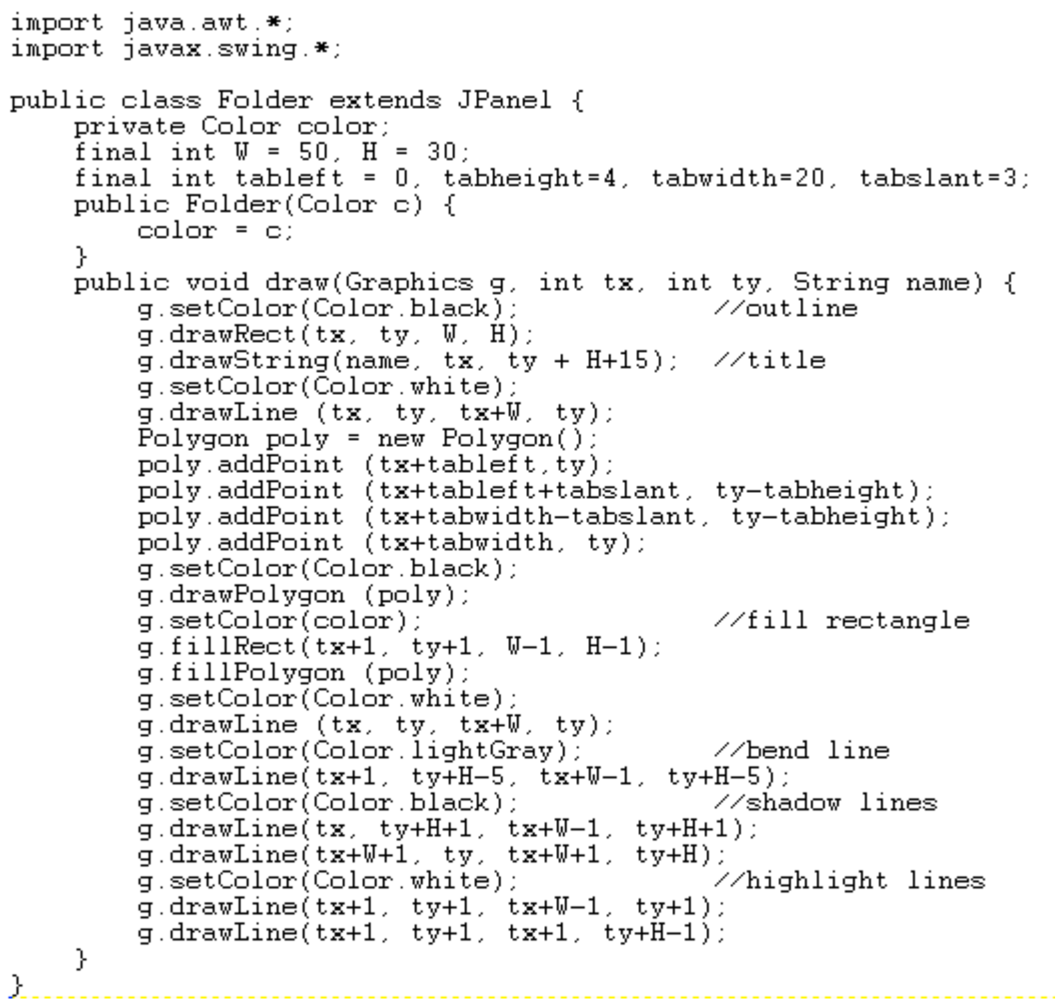
FolderFactory.java

```
import java.awt.*;

public class FolderFactory {
    Folder unSelected, Selected;
    public FolderFactory() {
        Color brown = new Color(0x5f5f1c);
        Selected =  new Folder(brown);
        unSelected = new Folder(Color.yellow);
    }

    public Folder getFolder(boolean isSelected) {
        if (isSelected)
            return Selected;
        else
            return unSelected;
    }
}
```

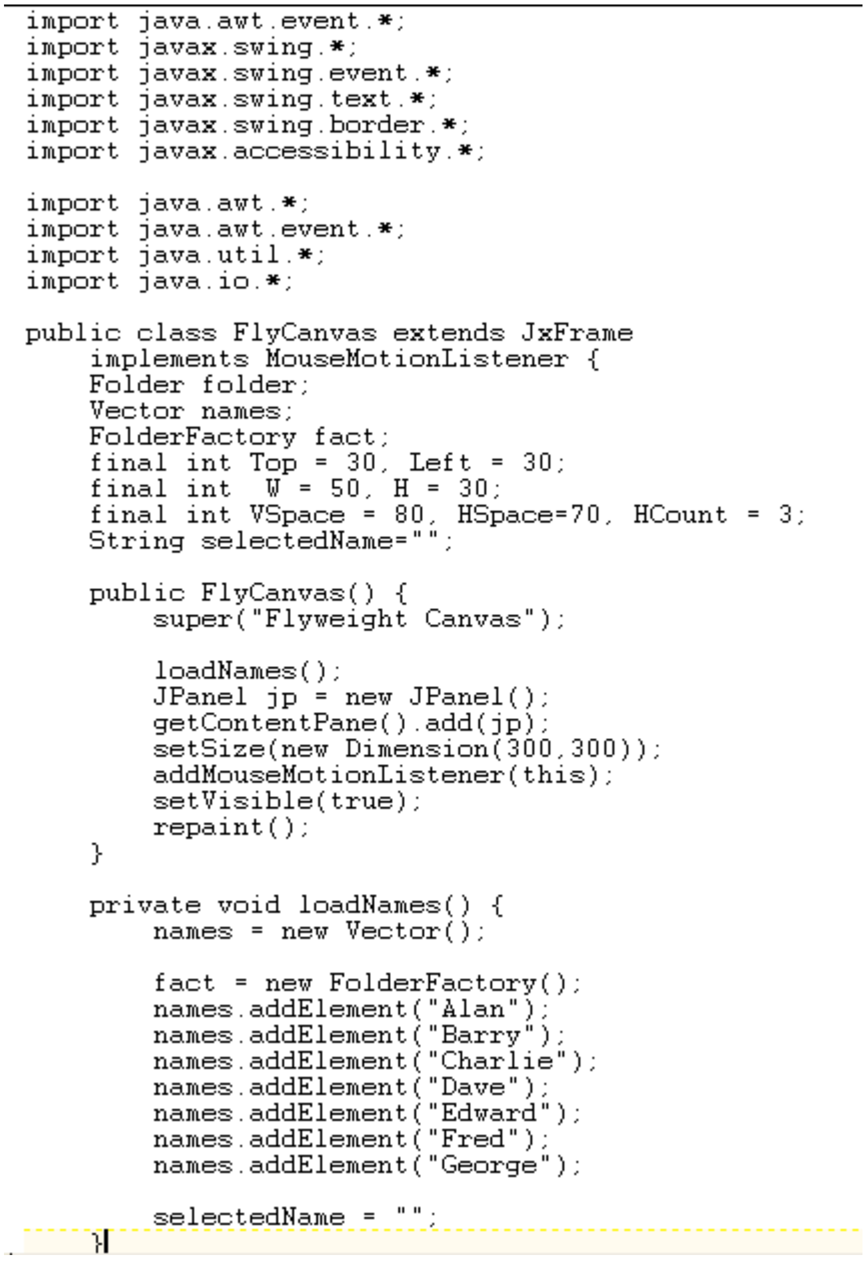
Folder.java



```java
import java.awt.*;
import javax.swing.*;

public class Folder extends JPanel {
    private Color color;
    final int W = 50, H = 30;
    final int tableft = 0, tabheight=4, tabwidth=20, tabslant=3;
    public Folder(Color c) {
        color = c;
    }
    public void draw(Graphics g, int tx, int ty, String name) {
        g.setColor(Color.black);            //outline
        g.drawRect(tx, ty, W, H);
        g.drawString(name, tx, ty + H+15);  //title
        g.setColor(Color.white);
        g.drawLine (tx, ty, tx+W, ty);
        Polygon poly = new Polygon();
        poly.addPoint (tx+tableft,ty);
        poly.addPoint (tx+tableft+tabslant, ty-tabheight);
        poly.addPoint (tx+tabwidth-tabslant, ty-tabheight);
        poly.addPoint (tx+tabwidth, ty);
        g.setColor(Color.black);
        g.drawPolygon (poly);
        g.setColor(color);                  //fill rectangle
        g.fillRect(tx+1, ty+1, W-1, H-1);
        g.fillPolygon (poly);
        g.setColor(Color.white);
        g.drawLine (tx, ty, tx+W, ty);
        g.setColor(Color.lightGray);        //bend line
        g.drawLine(tx+1, ty+H-5, tx+W-1, ty+H-5);
        g.setColor(Color.black);            //shadow lines
        g.drawLine(tx, ty+H+1, tx+W-1, ty+H+1);
        g.drawLine(tx+W+1, ty, tx+W+1, ty+H);
        g.setColor(Color.white);            //highlight lines
        g.drawLine(tx+1, ty+1, tx+W-1, ty+1);
        g.drawLine(tx+1, ty+1, tx+1, ty+H-1);
    }
}
```

FlyCanvas.java



```
import java.awt.event.*;
import javax.swing.*;
import javax.swing.event.*;
import javax.swing.text.*;
import javax.swing.border.*;
import javax.accessibility.*;

import java.awt.*;
import java.awt.event.*;
import java.util.*;
import java.io.*;

public class FlyCanvas extends JxFrame
    implements MouseMotionListener {
    Folder folder;
    Vector names;
    FolderFactory fact;
    final int Top = 30, Left = 30;
    final int  W = 50, H = 30;
    final int VSpace = 80, HSpace=70, HCount = 3;
    String selectedName="";

    public FlyCanvas() {
        super("Flyweight Canvas");

        loadNames();
        JPanel jp = new JPanel();
        getContentPane().add(jp);
        setSize(new Dimension(300,300));
        addMouseMotionListener(this);
        setVisible(true);
        repaint();
    }

    private void loadNames() {
        names = new Vector();

        fact = new FolderFactory();
        names.addElement("Alan");
        names.addElement("Barry");
        names.addElement("Charlie");
        names.addElement("Dave");
        names.addElement("Edward");
        names.addElement("Fred");
        names.addElement("George");

        selectedName = "";
    }
    public void paint(Graphics g) {
        Folder f;
        String name;

        int j = 0;       //count number in row
        int row = Top;  //start in upper left
        int x = Left;

        //go through all the names and folders
        for (int i = 0; i< names.size(); i++) {
            name = (String)names.elementAt(i);
            if (name.equals(selectedName))
                f = fact.getFolder(true);
            else
                f = fact.getFolder(false);
            //have that folder draw itself at this spot
            f.draw(g, x, row, name);

            x = x + HSpace;    //change to next posn
            j++;
            if (j >= HCount) { //reset for next row
                j = 0;
                row += VSpace;
                x = Left;
            }
        }
    }

    public void mouseMoved(MouseEvent e) {
        int j = 0;       //count number in row
        int row = Top;  //start in upper left
        int x = Left;

        //go through all the names and folders
        for (int i = 0; i< names.size(); i++) {
            //see if this folder contains the mouse
            Rectangle r = new Rectangle(x,row,W,H);
            if (r.contains(e.getX(), e.getY())) {
                selectedName=(String)names.elementAt(i);
                repaint();
            }
            x = x + HSpace;      //change to next posn
            j++;
            if (j >= HCount) {   //reset for next row
                j = 0;
                row += VSpace;
                x = Left;
            }
        }
    }

    public void mouseDragged(MouseEvent e) {
    }

    static public void main(String[] argv) {
        new FlyCanvas();
    }
}
```

// End of FlyCanvas

Question 4 (Jan. 2004)

a) With the aid of some simple UML diagrams, describe the difference between the Factory Pattern, Abstract Factory Pattern and the Factory method pattern.

[12 Marks]

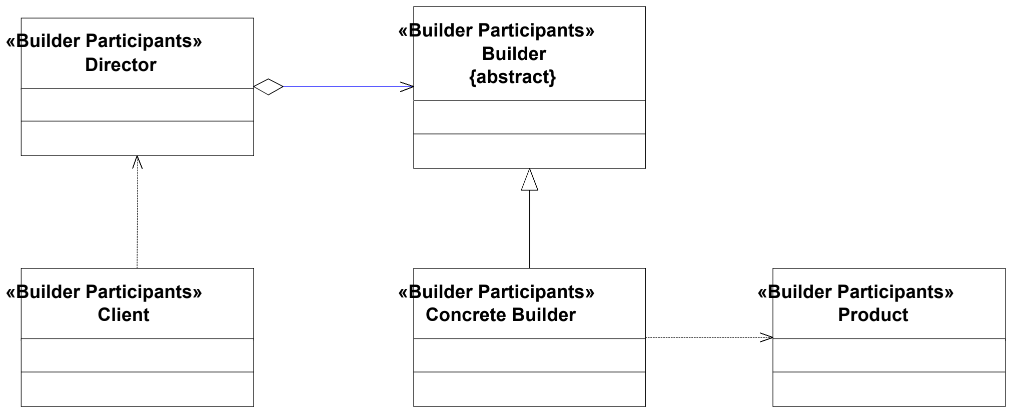
b) What is the Builder Pattern?

Why would you use it.

Illustrate your answer with two real-world examples.

[4 Marks]

c) Given the following UML diagram, explain briefly the role of each participant in the Builder Pattern.


«Builder Participants» Director
«Builder Participants» Builder {abstract}
«Builder Participants» Client
«Builder Participants» Concrete Builder
«Builder Participants» Product

[5 Marks]

d) Describe four consequences of the Builder pattern.

[4 Marks]

(Total Marks 25)

Question 5 (Jan. 2005)

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).

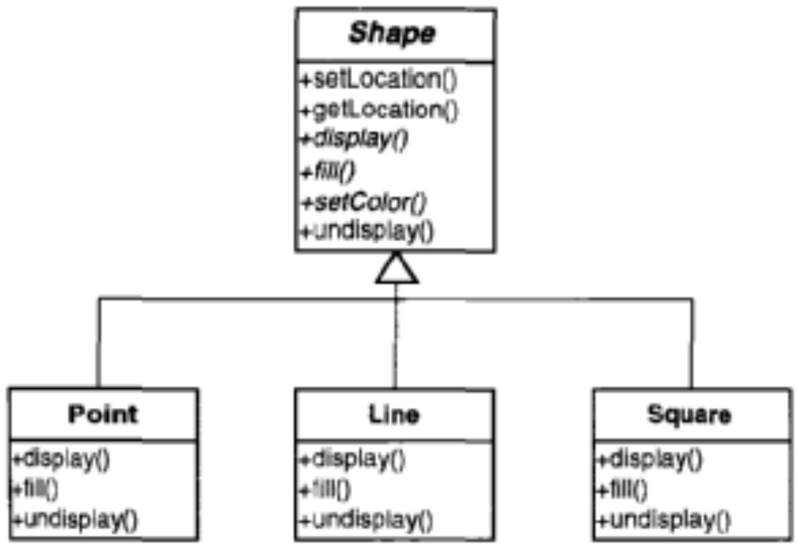


Figure 7-3 Points, Lines, and Squares showing methods.

After implementing the above system, the designer was then asked to include a class that could draw circles. 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 circle 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 ‘circle’ 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 ‘circle’ that implements the **Adapter** pattern.

[4 Marks]

(Total Marks 25)