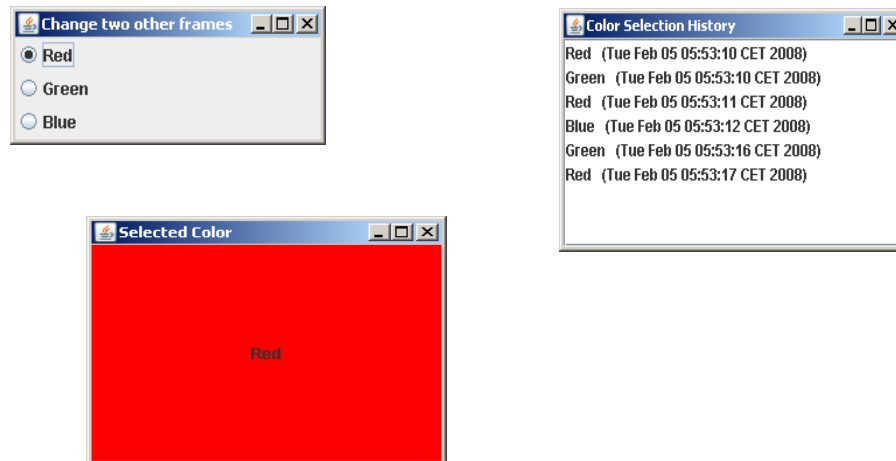
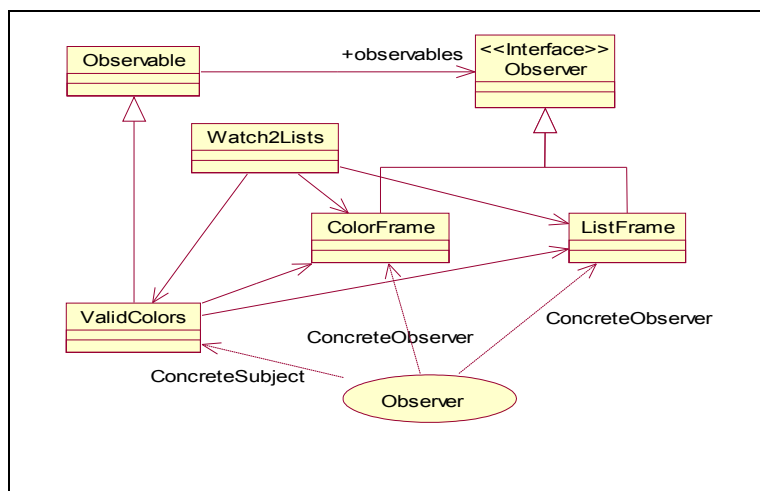


Name: _____ Registration-Nb.: _____

Task 1 Design Pattern

Following screenshots show the result of an executed program – code attached

**1.a Which Design Pattern was used?****[1 Point]****Observer****1.b Design a UML-Diagram describing this system.
Which class takes which role?****[3 points]**

```
//-----  
// Remark: This code was originally developed by Volker Wurst  
//-----  
  
import java.util.Observable;  
import java.util.Vector;  
  
public class ValidColors extends Observable {  
  
    private Vector<String> colors = new Vector<String>();  
  
    public Vector<String> getColors() {  
        return colors;  
    }  
  
    public void setColors(Vector<String> colors) {  
        this.colors = colors;  
        select(colors.firstElement());  
    }  
  
    public void addColor(String colorname) {  
        this.colors.add(colorname);  
        select(colorname);  
    }  
  
    public void select(String colorname) {  
        setChanged();  
        notifyObservers(colorname);  
    }  
}
```



```
//-----
import java.awt.*;
import java.awt.event.*;
import java.util.Iterator;
import javax.swing.*;

public class Watch2Lists extends JFrame implements ItemListener {
    private static final long serialVersionUID = 1L;
    ValidColors colors = new ValidColors();

    public Watch2Lists() {
        super("Change two other frames");
        colors.addColor("Red");
        colors.addColor("Green");
        colors.addColor("Blue");

        JPanel p = new JPanel(true);
        p.setLayout(new BorderLayout());
        getContentPane().add("Center", p);
        Box box = new Box(BoxLayout.Y_AXIS);
        p.add("Center", box);

        ButtonGroup bgr = new ButtonGroup();
        Iterator<String> elements = colors.getColors().iterator();
        while (elements.hasNext()) {
            JRadioButton button;
            box.add(button = new JRadioButton((String) elements.next()));
            button.addItemListener(this);
            bgr.add(button);
        }

        WindowListener l = new WindowAdapter() {
            public void windowClosing(WindowEvent e) {
                System.exit(0);
            }
        };
        addWindowListener(l);

        setBounds(150, 250, 100, 100);
        setVisible(true);
    }

    public void itemStateChanged(ItemEvent e) {
        if (e.getStateChange() == ItemEvent.SELECTED)
            colors.select(((JRadioButton) e.getSource()).getText());
    }

    public ValidColors getSubject() {
        return colors;
    }

    public static void main(String[] args) {
        Watch2Lists main = new Watch2Lists(); // Create window for Subject
        ValidColors subjectcolors = main.getSubject(); // Get the subject
        new ColorFrame(subjectcolors); // Create visualization Observer
        new ListFrame(subjectcolors); // Create textual Observer
    }
}
```



```
//-----  
import java.awt.*;  
import javax.swing.*;  
import java.util.Observable;  
import java.util.Observer;  
  
public class ColorFrame extends JFrame implements Observer {  
  
    private static final long serialVersionUID = 1L;  
    Color color;  
    String color_name;  
    JPanel p;  
    JLabel l;  
  
    public ColorFrame(ValidColors colors) {  
        super("Selected Color");  
  
        setBounds(100, 100, 100, 100);  
        getContentPane().setBackground(Color.gray);  
        setVisible(true);  
  
        l = new JLabel(color_name, JLabel.CENTER);  
        getContentPane().add(l);  
  
        colors.addObserver(this);  
    }  
  
    public void update(Observable o, Object arg) {  
  
        color_name = (String) arg;  
        if (color_name.toUpperCase().equals("RED"))  
            color = Color.red;  
        if (color_name.toUpperCase().equals("BLUE"))  
            color = Color.blue;  
        if (color_name.toUpperCase().equals("GREEN"))  
            color = Color.green;  
  
        getContentPane().setBackground(color);  
        l.setText(color_name);  
    }  
}
```



```
//-----  
import java.awt.*;  
import java.util.Date;  
import java.util.Observable;  
import java.util.Observer;  
import java.util.Vector;  
  
import javax.swing.*;  
  
public class ListFrame extends JFrame implements Observer {  
  
    private static final long serialVersionUID = 1L;  
    JList list;  
    JPanel p;  
    JScrollPane lsp;  
    Vector<String> listData;  
  
    public ListFrame(ValidColors colors) {  
        super("Color Selection History");  
  
        p = new JPanel(true);  
        getContentPane().add("Center", p);  
        p.setLayout(new BorderLayout());  
  
        listData = new Vector<String>();  
        list = new JList(listData);  
        lsp = new JScrollPane();  
        lsp.getViewport().add(list);  
  
        p.add("Center", lsp);  
        setBounds(250, 100, 300, 100);  
        setVisible(true);  
  
        colors.addObserver(this);  
    }  
  
    public void update(Observable o, Object arg) {  
        listData.addElement(  
            (String) arg + "    (" + new Date().toString() + ")");  
        list.setListData(listData);  
        lsp.revalidate();  
        lsp.repaint();  
    }  
}
```

