Notes on tutorial #5 (for week 8 - Feb 27, 2004)

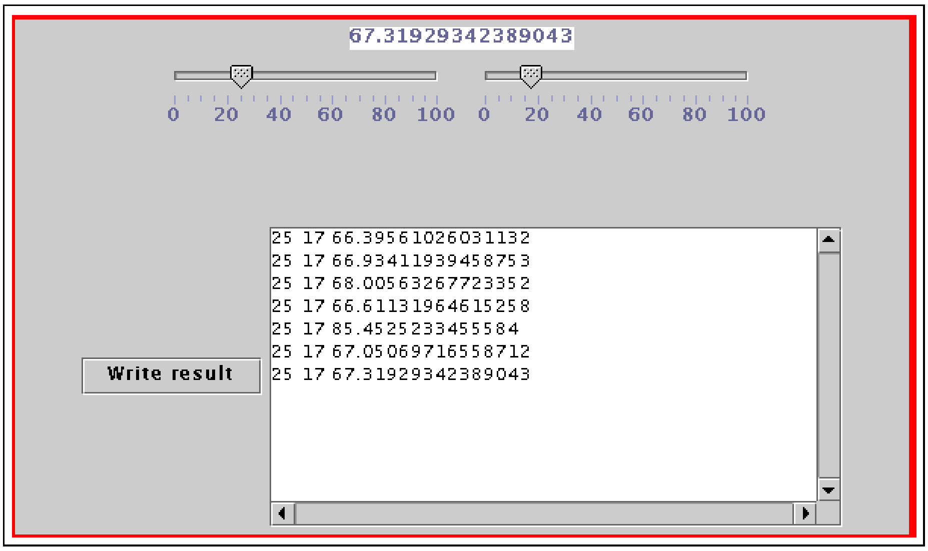
27th February 2004

Q1: Examine the Java/Swing code below. Draw a diagram showing the resultant display, and give a brief description of the function of the code. (In what environment would it be run? What does it look like? What does it do?).

CODE LISTING **Converter.java**

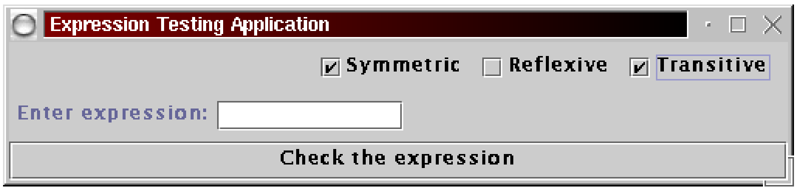
```
public class Converter extends JApplet  {
    JPanel Panel1, Panel2, Panel3;  JSlider Slider1, Slider2;
    JButton WriteButton;            JTextArea textArea;
    JLabel label1;
    int width1=0, width2=0;         Double realwidth;
    static final double a=20,b=0.87,c=1.32,d=22.25;
    public Random rand = new Random(System.currentTimeMillis());
    JComponent contentPane;
    private double DoCalc(){
        return a+(b*width1)+(c*width2)+(d*rand.nextDouble());
    }
    public Converter() {
        Panel1 = new JPanel() {};  Panel2 = new JPanel() {};
        Panel3 = new JPanel() {};
        Slider1 = new JSlider (JSlider.HORIZONTAL,0,100,width1) {};
        Slider1.addChangeListener(new ChangeListener() {
                public void stateChanged( ChangeEvent evt ) {
                    JSlider source = (JSlider) evt.getSource();
                    width1         = (int)((JSlider)evt.getSource()).getValue();
                    realwidth      = new Double(DoCalc());
                    label1.setText(realwidth.toString());
                }});
        Slider1.setMajorTickSpacing(20);  Slider1.setMinorTickSpacing(5);
        Slider1.setPaintTicks(true);      Slider1.setPaintLabels(true);
        Panel1.add(Slider1);
        Slider2 = new JSlider (JSlider.HORIZONTAL,0,100,width2) {};
        Slider2.addChangeListener(new ChangeListener() {
                public void stateChanged( ChangeEvent evt ) {
                    JSlider source = (JSlider) evt.getSource();
                    width2         = (int)((JSlider)evt.getSource()).getValue();
                    realwidth      = new Double(DoCalc());
                    label1.setText(realwidth.toString());
                }});
        Slider2.setMajorTickSpacing(20);  Slider2.setMinorTickSpacing(5);
        Slider2.setPaintTicks(true);      Slider2.setPaintLabels(true);
        Panel1.add(Slider2);
        WriteButton = new JButton( "Write result" );
        WriteButton.addActionListener(new java.awt.event.ActionListener() {
            public void actionPerformed(java.awt.event.ActionEvent evt) {
                realwidth = new Double(DoCalc());
                label1.setText(realwidth.toString());
                textArea.append(width1+" "+width2+" "+realwidth.toString()+"\n");
        }});
        contentPane = new JPanel();   contentPane.setBackground(Color.red);
        contentPane.setLayout(new BorderLayout());
        contentPane.setBorder(BorderFactory.createEmptyBorder(5,5,5,5));
        contentPane.add(Panel1,BorderLayout.CENTER);
        Panel2.add(WriteButton);
        textArea = new JTextArea(12, 30);  textArea.setEditable(false);
        JScrollPane scrollPane = new JScrollPane(textArea,
                                     JScrollPane.VERTICAL_SCROLLBAR_ALWAYS,
                                     JScrollPane.HORIZONTAL_SCROLLBAR_ALWAYS);
        Panel2.add(scrollPane);
        contentPane.add(Panel2,BorderLayout.SOUTH);
        label1 = new JLabel("        ");  label1.setOpaque(true);
        label1.setBackground(Color.white);
        Panel3.add(label1);
        contentPane.add(Panel3,BorderLayout.NORTH);
        realwidth = new Double(DoCalc());
        label1.setText(realwidth.toString());
    }
    public void init() {
        Converter converter = new Converter();
        setContentPane(contentPane);
        contentPane.setVisible(true);
    }
}
```

Answer: *It is an applet for inclusion in a web page. You move sliders, and when you press the button, some information is calculated and put on the screen.*



```
25 17 66.39561026031132
25 17 66.93411939458753
25 17 68.00563267723352
25 17 66.61131964615258
25 17 85.45252334555 84
25 17 67.05069716558712
25 17 67.31929342389043
```

Q2: Write the Java/Swing code for a small application with three checkbuttons, a text entry box and a button with a label, laid out as shown in this image. No functionality need be included.



☑ Symmetric ☐ Reflexive ☑ Transitive

Answer: *Something like this:*

CODE LISTING **Sample.java**

```
public class Sample extends javax.swing.JFrame {
   public Sample () {
      initComponents();
   }
    private void initComponents() {
      javax.swing.JPanel p = new javax.swing.JPanel ();
      p.setLayout (new java.awt.FlowLayout (java.awt.FlowLayout.RIGHT));
      SymCheck = new javax.swing.JCheckBox ("Symmetric");
      p.add (SymCheck);
      ReflCheck = new javax.swing.JCheckBox ("Reflexive");
      p.add (ReflCheck);
      TransCheck = new javax.swing.JCheckBox ("Transitive");
      p.add (TransCheck);
      getContentPane ().add (p, "North");
      javax.swing.JPanel pn = new javax.swing.JPanel ();
      pn.setLayout (new java.awt.FlowLayout (java.awt.FlowLayout.LEFT));
      pn.add (new javax.swing.JLabel ("Enter expression:"));
      l_to = new javax.swing.JTextField (10);
      pn.add (l_to);
      getContentPane ().add (pn, "Center");
      checkit = new javax.swing.JButton ("Check the expression");
      getContentPane ().add (checkit, "South");
      setTitle ("Expression Testing Application");
      addWindowListener (new java.awt.event.WindowAdapter () {
         public void windowClosing (java.awt.event.WindowEvent evt) {
            System.exit (0);
         }
      });
   }
   public static void main (String[]args) {
      new Sample().show ();
   }
   private javax.swing.JScrollPane scrollPane;
   private javax.swing.JTextArea textArea;
   private javax.swing.JTextField l_to, l_title;
    private javax.swing.JButton checkit;
   private javax.swing.JCheckBox SymCheck, ReflCheck, TransCheck;
}
```

Q3: Your employer wishes you to develop an application to provide an on-line help desk function to staff distributed around Asia. The help desk application will provide instant text messaging, chat and voice chat and searchable access to a database.

(a) Give at least two arguments/reasons/justifications as to why you ***should*** implement this application as a Java applet.

Answer: *For example:*

(i) *To make the application as widely available as possible, we should use java applets for each application. The clients can be hosted on any platform (Mac, Windows, Linux) that supports the JVM in a web-browser.*

(ii) *To ensure that the latest revision of the help-desk software is always used by the clients.*

(b) Give at least two arguments/reasons/justifications as to why you ***should not*** implement this application as a Java applet.

Answer: *For example:*

(i) *As there is a need for real-time/speed in the voice-chat, we may not be able to do this with an applet, which would funnel all network connections through the single server which will quickly become loaded. A separate application supporting peer-to-peer voice ciommunication may be more appropriate.*

(ii) *The java applet security model only allows access back to the same host that provided the applet. If we require access to a remote database, the applet security model will not allow this.*

(iii) *There may be an issue with slow startup speeds, as this application will be slow and take quite a while to download over a network to low-bandwidth clients.*

(iv) *There may be security issues with running an applet - particularly if the server site is spoofed. Perhaps it would be safer to have an application residing permanently and safely on the client.*

Q4: When programming using Java it is common to use threads. Give two plausible examples of the use of threads in Java, not taken from the notes given in class. For each example, briefly describe it and state the need for the thread as opposed to just coding it in-line. You may draw your examples from your own experience.

Answer: *For example:*

(i) *Downloading data and outputting it simultaneously but at a different rate - for example when downloading and playing mp3 (audio) files.*

(ii) *Processing data from a database while allowing data entry - for example doing interest calculations on database records.*

(iii) *Maintaining a communication protocol with a remote device - for example a remote weather station may require interaction every 20 seconds to maintain the link.*

(iv) *To handle multiple client connections - for example if our application included a server for remote clients, a new thread could be started for each client.*