SWING

1. How to display the results of a JDBC Query in a JTable?
   a. Create an adapter class that extends AbstractTableModel and read the result of the query into a Vector of Vectors, and have the getValueAt(row,col) get the value from the corresponding Vector element.
   b. Loop through the ResultSet, individually setting the value for one cell at a time in the JTable with DefaultTableModel via the JTable's setValueAt(value,row,column) method.
   c. Pass a reference to the GUI object via a constructor parameter, and then use public methods of that GUI object(not a constructor) to set or change data.
   d. Create an adapter class that implements TableCellEditor, which uses a singleton to find the Resultset.
   e. Extend the JTable class and override the paint() method to open a database connection and loop through the ResultSet, drawing the text for each element in the appropriate cell in the table.

2. You add data to a Swing component, such as a JTable, JTree, or JList, via the:
   a. component's peer.
   b. component's view.
   c. component's model.
   d. component's controller.
   e. component.

3.
```
import java.awt.*;
import javax.swing.*;
import javax.swing.border.*;

public class Grid extends JPanel{
private static int ROWS = 7;
private static int COL = 10;

public void Generator() {
 ImageIcon wIcon = new ImageIcon;
 JPanel jPan1 = new JPanel();
 jPan1.setLayout ((LayoutManager) new
 GridLayout(rows,col,1,1));
 jPan1.setSize(350,350);
 TitleBorder bdr =
 javax.swing.BorderFactory.createTitleBorder(null, “Targeting
 Grid”);
 Bdr.setTitleColor(java.awt.Color.RED);
 jPan1.setLayout ((LayoutManager) new
 GridLayout(rows,col,1,1));
 jPan1.setBorder(bdr);
 JButton b[] = new JButton[rows*col];
  for (i=0;j=rows*col;i<j; i++){
       b[i] = new JButton(wIcon)
       b[i].setSize(20,20);
       b[i].setMaximumSize(new Dimension(20,20));
       b[i].setPreferredSize(new Dimension(20,20));
       System.out.println(“Looptest “+ i);
       jPan1.add(b[i]);
  }
}
}
```

Based on the above sample code, how do you fix the code so the buttons show on the created grid?

a. Replace all instances of JPanel to jPan1
b. Change the class JPanel from public to private, and then rebuild.
c. Remove “jPan1 = JPanel();” and replace all remaining occurrences “jPan1” with “this”.
d. Create the grid prior to adding buttons by inserting Grid = new Grid.
e. Remove all extra set of { } that are present in the code.

4. Sample Code:

```
public void setupFrame(String example) {
      JButton button = new JButton (“Example”);
      button.addActionListener (new ActionListener () {
            public void actionPerformed (ActionEvent e) {
                  System.out.println(example);
      } ) ;
```

Which do you add to the declaration of the parameter name example for the sample code above to compile?

a. private
b. final
c. public
d. volatile
e. transient

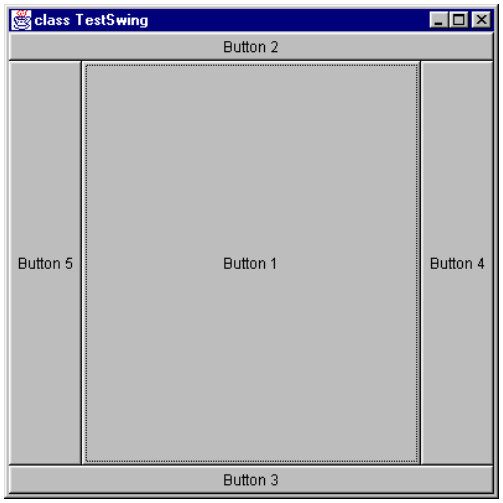
5. Sample Code

```
import javax.swing.*;
import java.util.*;
public class X {
      public static void main(String args[]) {
            List<JButton> list = new ArrayList<JButton>();
            JLabel label = new JLabel("Example");
            list.add(label);
            System.out.println(list);
      }
}
```

Referring to the sample code above, what happens when you add a JLabel to a List of JButton objects?

a. At run time, a ClassCastException is thrown when adding to the list.
b. At run time, an InvalidArgumentException is thrown when adding to the list.
c. The contents of the list are printed.
d. At compile time, the compiler rejects the attempt to add the JLabel.
e. At run time, an IncorrectCastException is thrown when printing the list.

6. How do you perform cleanup when a user is trying to close a window?
    a. Create an inner class called WindowListener, extend WindowEventHandler, and put cleanup code in a processEvent(Event e) method.
    b. Place cleanup code in a try-catch block that catches a FrameClosingException.
    c. Implement WindowListener, and put cleanup code in a public void windowClosing(WindowEvent method.
    d. Implement FrameListener, and place cleanup code in a frameClosing(FrameEvent e) method.
    e. Allow for the garbage collector to do automatic cleanup



7. Which layout manager do you use to produce the layout shown in the image above?
    a. GridLayout
    b. BorderLayout
    c. BoxLayout
    d. ViewportLayout
    e. FlowLayout

8. Line 1 static boolean bufferedImageEquals(BufferedImage b1, BufferedImage b2) {
```
Line 2   if ( b1 == b2 ) {return true;}
Line 3   if ( b1 == null || b2 == null ) { return false; }
Line 4   if ( b1.getWidth() != b2.getWidth()){ return false;}
Line 5   if ( b1.getHeight() != b2.getHeight()){return false;}
Line 6   for ( int i = 0; i < b1.getWidth(); i++) {
Line 7     for ( int j = 0; j < b1.getHeight(); i++ ) {
Line 8       if ( b1.getRGB(i,j) != b2.getRGB(i,j) ) {
Line 9              return false;
Line 10      }
Line 11    }
Line 12   }
Line 13  return true;
Line 14 }
```
Which do you change in the sample code above to eliminate the error:
"java.lang.ArrayIndexOutOfBoundsException: Coordinate out of bounds!"?

a. Remove Line 8 from the code.
b. The "i++" needs to be changed to "j++" in the second for loop.
c. Add if (b1 == null || b2 == null) {return true;} after Line 2.
d. The class needs to be changed to "if (b1.getAlpha() != b2.getAlpha()) { return false; }".
e. Change Line 2 to if (b1 == b2) {return false;}

9.
```
import java.awt.EventQueue;
import javax.swing.text.JTextComponent;
public class TestInnerClass {
      public void setTheTextBox(JTextComponent cmp, String a) {
            EventQueue.invokeLater(new Runnable() {
                  public void run() {
                        cmp.setText(a);
                  }
            });
      }
}
```
Why does the sample code above produce an error?

a. The anonymous inner class cannot extend Runnable because Runnable is an interface and cannot be instantiated.
b. The setTheTextBox() method needs to be declared static so inner classes can be instantiated without an object of the outer class.
c. The run() method needs to have a reference to the parent method to resolve local object references.
d. JTextComponent cmp and String a are not accessible from the context of the inner class because they need to be declared final.
e. EventQueue.invokeLater() requires a Timer object to determine when the method should be invoked.