CompSci 101 AC - 2001
Terms Test Answers

Question 1:

Output from class `Q1`:

```
6
1
5.0
4.0
5.0
```

Question 2:

The completed `actionPerformed()` method for the class `Q2`:

```
public void actionPerformed(ActionEvent e) {
     String s1 = tOne.getText();
     String s2 = tTwo.getText();
     tOne.setText(s2);
     tTwo.setText(s1);
}
```

Question 3:

The complete `Q3` class:

```
import java.awt.*;
import java.applet.*;

public class Q3 extends Applet {

     public void paint(Graphics g) {

          final int SIZE = 25;

          g.drawRect(0, 0, 2*SIZE, 2*SIZE);
          g.drawRect(2*SIZE, 2*SIZE, 2*SIZE, 2*SIZE);

          g.drawOval(0, 2*SIZE, 2*SIZE, 2*SIZE);
          g.drawOval(2*SIZE, 0, 2*SIZE, 2*SIZE);

          g.drawLine(SIZE, SIZE, 3*SIZE, 3*SIZE);
          g.drawLine(SIZE, 3*SIZE, 3*SIZE, SIZE);


     }

}
```

Question 4:

The complete Q4 class:

```
import java.awt.*;
import java.applet.*;
import java.awt.event.*;

public class Q4 extends Applet implements MouseListener {

     private TextField tXChange, tYChange;
     private int prevX, prevY, currentX, currentY;

     public void init() {

          tXChange = new TextField("0  ");
          tYChange = new TextField("0  ");

          add(new Label("x-change:"));
          add(tXChange);
          add(new Label("y-change:"));
          add(tYChange);

          prevX = 100;
          prevY = 100;
          currentX = 100;
          currentY = 100;

          addMouseListener(this);
     }

     public void paint(Graphics g) {

          final int SIZE = 4;
          g.setColor(Color.red);
          g.drawLine(prevX, prevY, currentX, currentY);
          g.setColor(Color.blue);
          g.fillRect(prevX - SIZE/2, prevY - SIZE/2, SIZE, SIZE);
          g.fillRect(currentX - SIZE/2, currentY - SIZE/2, SIZE, SIZE);
     }

     public void mousePressed(MouseEvent e) {

          prevX = currentX;
          prevY = currentY;

          currentX = e.getX();
          currentY = e.getY();

          tXChange.setText(String.valueOf(currentX - prevX));
          tYChange.setText(String.valueOf(currentY - prevY));

          repaint();
     }

     public void mouseReleased(MouseEvent e) {}
     public void mouseEntered(MouseEvent e) {}
     public void mouseExited(MouseEvent e) {}
     public void mouseClicked(MouseEvent e) {}

}
```

Question 5:

Output from the class `Q5`:

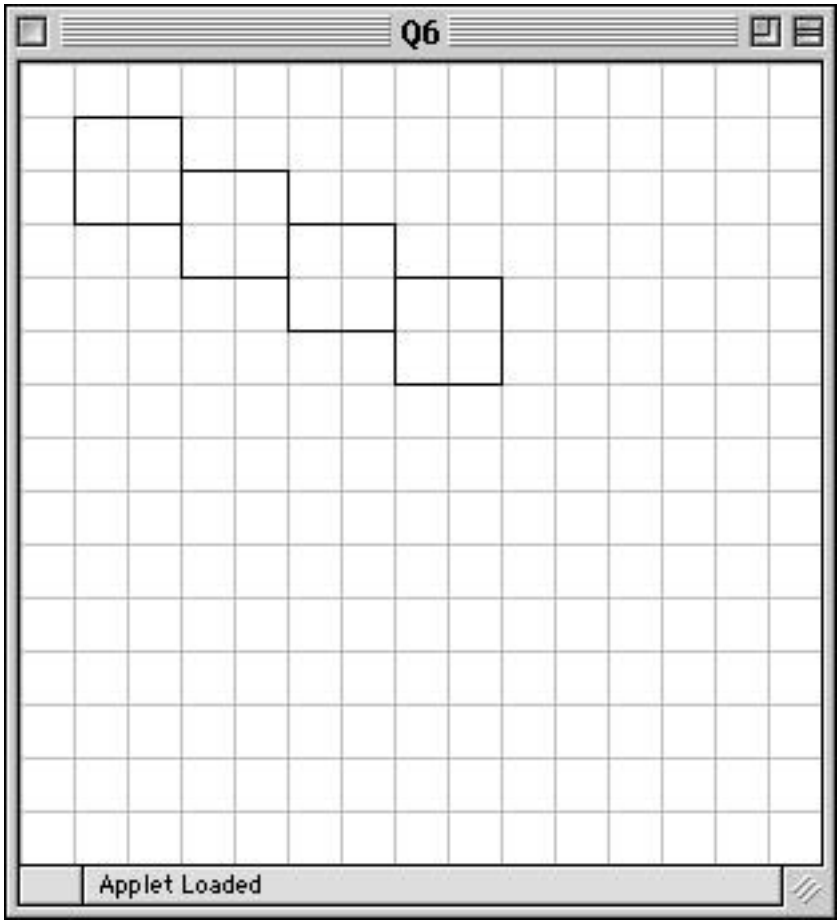
```
1. value: 90
isValid: false
2. value: 83
```

Question 6:

(i) Output from the class `Q6`:

```
value: 4
value: 7
value: 10
value: 13
value: 25
```

(ii) Output from the `paint()` method:



Question 7:

Output from the class `Q7`:

```
10
50.0
one: 110
two: 160.0
three: 340.0
```

Question 8:

The `DrawableString` class:

```
import java.awt.*;

public class DrawableString {

     private String text;
     private int xPos, yPos;

     public DrawableString(String initialText, int x, int y) {
          text = initialText;
          xPos = x;
          yPos = y;
     }

     public void setPos(int x, int y) {
          xPos = x;
          yPos = y;
     }

     public void setText(String s) {
          text = s;
     }

     public void draw(Graphics g) {
          g.drawString(text, xPos, yPos);
     }
}
```

Question 9:

The class Q9:

```
import java.awt.*;
import java.applet.*;
import java.awt.event.*;

public class Q9 extends Applet implements ActionListener {

     // The array of ints
     private int[] theNums = {36, 60, 78, 65, 93, 43, 37, 59, 108, 114};

     private Button bChangeValues;

     public void init() {
          bChangeValues = new Button("CHANGE VALUES");
          bChangeValues.addActionListener(this);
          add(bChangeValues);
     }

     public void actionPerformed( ActionEvent e) {

          for (int i = 0; i < theNums.length; i++) {
               theNums[i] = (int)(Math.random() * 90 + 30);
          }

          repaint();
     }


     public void paint( Graphics g ) {
          final int BAR_LEFT = 10; /** left of bar*/
          final int BAR_RIGHT = 230; /** right of  bar*/
          final int BAR_TOP = 40; /** top of bar*/
          final int BAR_BOTTOM = 50;  /** bottom of bar*/

          /** horizontal step between each line */
          final int X_STEP = 20;

          /** x position of first line */
          final int FIRST_LINE_POS = BAR_LEFT + X_STEP;

          //draw the top bar in red
          g.setColor(Color.red);
          g.fillRect(BAR_LEFT, BAR_TOP, BAR_RIGHT - BAR_LEFT,
                                                  BAR_BOTTOM - BAR_TOP);

          //draw the lines hanging from the bar
          g.setColor(Color.black);
          for (int i = 0; i < theNums.length; i++) {
               g.drawLine(i * X_STEP + FIRST_LINE_POS, BAR_BOTTOM,
                   i * X_STEP + FIRST_LINE_POS, BAR_BOTTOM + theNums[i]);
          }
     }
}
```