INSTRUCTIONS TO CANDIDATES

Answer all questions.

If you attempt to answer more questions than the required number of questions (in any section), the marks awarded for the excess questions answered will be discarded (starting with your lowest mark).
In this exam, you are asked to develop some Java code that could be used for a simple tool that allows a user to draw a variety of shapes on a canvas. You are also asked to comment on some relevant design and programming issues. Some questions ask you to write Java code: for these questions, you should write code that is as detailed and as close to Java syntax as possible; however, minor divergences from Java syntax, and minor errors in the names of classes and methods in Java packages such as the AWT, will not be penalised. If you are uncertain about the syntactical constructs, some marks will be given for pseudocode, but you should make the pseudocode as detailed as possible.

1. If a shape is to be drawn on a canvas, it will have a particular location on the canvas; this location can be represented by x- and y-coordinates representing an ‘anchor point’ for the shape (for example, the anchor point for a rectangle might be its top left corner; for a circle, the anchor point might be the centre of the circle). A shape will also have a particular colour. Thus, a shape might be represented in Java by a class which has fields to represent the x- and y-coordinates of its anchor point, and a field to represent a colour (say using the java.awt.Color class). These fields represent the kind of data that is common to all shapes; obviously, however, particular shapes differ in the way that they are drawn: circles look different to rectangles. The following is an abstract Java class that is intended to capture the common data representation of anchor points and colours; it will have concrete subclasses to represent the different kinds of shapes, so there will be one concrete subclass for circles, another for rectangles, and so on. Each concrete subclass will implement the draw() method in a way that is appropriate for drawing a particular kind of shape; e.g., in the concrete subclass for circles, the draw() method will be implemented to draw a circular shape on the canvas, in the concrete subclass for rectangles, it will draw a rectangular shape, and so on.

```java
import java.awt.Graphics;
import java.awt.Color;

public abstract class AbstractShape
{
    // The x-coordinate of the anchor point
    private int x;

    // The y-coordinate of the anchor point
    private int y;

    // The colour of the shape
    private Color color;

    public AbstractShape(int x, int y, Color c)
    {
        this.x = x;
        this.y = y;
        color = c;
    }

    // Abstract method which will be implemented in concrete subclasses
    public abstract void draw(Graphics g);
}
```
public Color getColor()
{
    return color;
}

// abstract method to draw the shape,
// given a Graphics context
public abstract void draw(Graphics g);

(a)  i. What are the differences between the scope modifiers private, protected and public?[9 marks]

    ii. The x and y fields in AbstractShape are declared private. What problem
        might this raise when implementing concrete subclasses of AbstractShape,
        and what changes could be made to the AbstractShape class to deal with the
        problem?[6 marks]

(b) Give Java code for a concrete subclass Rectangle of AbstractShape, which
    will represent a rectangle that can be drawn on a canvas. (A useful method for this
    class is the

    java.awt.Graphics.fillRect(int,int,int,int),

    method, which draws a rectangle with top left corner at the x- and y-coordinates spec-
    ified by the first two parameters, and whose width and height are specified by the third
    and fourth parameters.) You may assume that the AbstractShape class contains
    any modifications given in your answer to Question 1(a)(ii) above.[10 marks]

2. The tool should allow the user to create new shapes, and each new shape that the user
    creates needs to be stored and displayed. These two requirements might be met by creating
    a subclass, say ShapeCanvas, of java.awt.Canvas that does the following:

    • allow the user to add a new shape of a chosen kind and colour by dragging the mouse
      on the canvas, and
    • store and display all the shapes that have been drawn on the canvas.

    The shapes that have been drawn on the canvas could be stored in a vector, shapes, of
type Vector<AbstractShape>. Note that this uses Java’s generic types: shapes is
    a vector whose elements are all of type AbstractShape.

(a) Give Java code that declares and initialises the variable shapes as a field of the
    ShapeCanvas class.[5 marks]

(b) Displaying all the shapes can be achieved by overriding the paint(Graphics g)
    method in the Canvas class. Give suitable code for the body of this method that
draws each shape stored in the shapes variable.[10 marks]
3. A graphical user-interface (GUI) for a tool to allow users to draw shapes on a canvas might be built up using the following code, which would be called from the constructor of a top-level class, say called DrawTool, that extends the javax.swing.JFrame class, using the ShapeCanvas class from Question 2:

```java
Container pane = getContentPane();

// A Canvas to display the drawn shapes
final ShapeCanvas canvas = new ShapeCanvas();
pane.add(canvas, BorderLayout.CENTER);

// controls to allow the user to select a shape
// to be drawn
Panel shapePanel = new Panel();
// add controls here ...

// controls to allow the user to select a colour
// for the shape to be drawn
Panel colorsPanel = new Panel();
// add controls here ...

// add the controls at the right-hand side
Panel choices = new Panel();
choices.setLayout(new GridLayout(2,1));
choices.add(shapesPanel);
choices.add(colorsPanel);
pane.add(choices, BorderLayout.EAST);
```

The controls at the right-hand side (BorderLayout.EAST) allow the user to select a shape (e.g., a circle, or a rectangle) to draw, and a colour for the shape.

(a) What GUI components, from either the java.awt or javax.swing packages would be appropriate for allowing the user to select a kind of shape to draw on the canvas?  

(b) Assuming that the user can only choose between drawing rectangles or circles, give Java code that adds the appropriate GUI components you identified in part (a) above.

4. The user might add a new shape to the canvas by selecting the kind of shape and its colour and then dragging the mouse on the canvas. Assume the available kinds of shape are rectangles and circles, implemented by concrete subclasses of AbstractShape called Rectangle and Circle, respectively. To draw a rectangle, the user clicks and holds the mouse where the top left corner of the rectangle is to be placed, and drags the mouse and releases it where the bottom right corner is to be placed. To draw a circle, the user clicks and holds the mouse where the centre of the circle is to be placed and drags the mouse and releases it at any point on the edge of the circle.
To capture these mouse events, the ShapeCanvas class might implement the java.awt.event.MouseAdapter

interface. The relevant methods in this interface are

    public void mousePressed(MouseEvent e);
    public void mouseReleased(MouseEvent e);

When the mouse is pressed, the x- and y-coordinates of the mouse’s location are stored in fields anchorPointX and anchorPointY. This could be achieved by including the following code in the ShapeCanvas class:

    private int anchorPointX;
    private int anchorPointY;

    public void mousePressed(MouseEvent e)
    {
        anchorPointX = e.getX();
        anchorPointY = e.getY();
    }

(a) Give Java code for the method mouseReleased that creates an instance of a red Rectangle of the appropriate width and height and at the appropriate location, and adds it to the vector shapes. [15 marks]

(b) Question 4(a) assumes that the user always wants to draw red rectangles; what changes would you need to make to the code in order to allow the user to specify the type and colour of shape to be drawn by using the controls given in Question 3. Give a detailed sketch of these changes, stating which kinds of event listener would be needed. [15 marks]

5. Describe the ‘Model-View-Controller’ architecture, and say to what extent the classes DrawTool, ShapeCanvas, and AbstractShape, together with its concrete subclasses for rectangles, etc., implement this architecture. [15 marks]