PRINCE OF SONGKLA UNIVERSITY

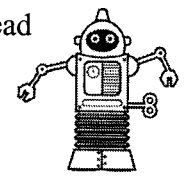
FACULTY OF ENGINEERING

Department of Computer Engineering

Final Examination: Semester 2 **Academic Year**: 2008-2009

Date: 23rd February, 2009 **Time**: 13:30 – 16:30 (**3 hours**)

Subject Number: 241-211 **Room**: Robot Head



Subject Title: Object Oriented Programming

Lecturer: Aj. Andrew Davison

Exam Duration: 3 hours

This paper has 3 pages.

Authorised Materials:

- Writing instruments (e.g. pens, pencils).
- Books (e.g. dictionaries) and calculators are **not** permitted.

Instructions to Students:

- *Answer questions in English.* Perfect English is **not** required.
- Attempt all questions.
- Write your answers in an answer book.
- Start your answer to each question on a new page
- Clearly number your answers.
- Any unreadable parts will be considered wrong.
- When writing programs, use good layout, and short comments; marks will not be deducted for minor syntax errors.
- The marks for each part of a question are given in brackets (...).

Question 1

(40 marks; 40 minutes)

a) What is a subclass and superclass? (10)

b) What is a protected method? (5)

c) What is a *polymorphic* data structure? (15)

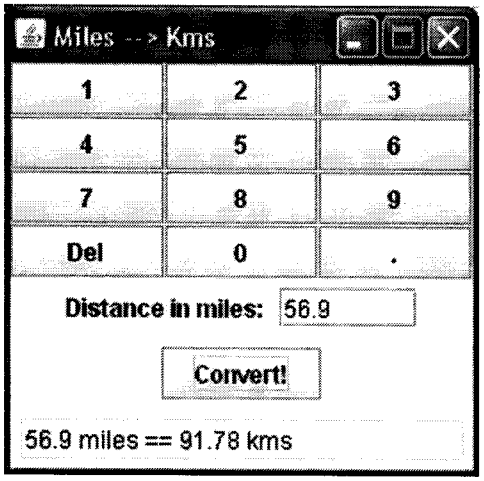
d) What are the main differences between an Interface and an Abstract class? (10)

Your answer should include diagrams and **small** code fragments where possible.

Question 2

(60 marks; 60 minutes)

Implement the following Java application which converts miles into kilometers.



The user inputs the miles value using the buttons, and it appears in the "Distance in miles" text field. The field can be cleared by the user pressing the "Del" button.

When the user presses the "Convert!" button, the text field at the bottom of the window displays the kilometers equivalent to the miles input.

Hints: the 12 buttons at the top of the window are grouped together in a JPanel acting as a container with its own layout manager..

The formula I use for converting miles to kilometers is:

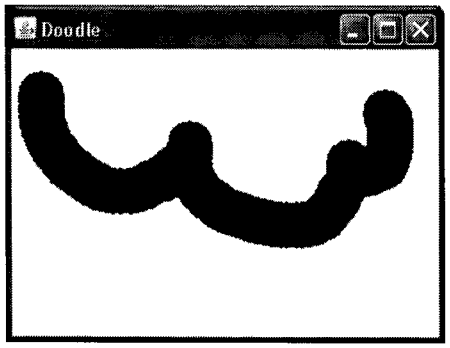
```
double kms = Math.round((miles/0.62) * 100 + 0.5)/100.0;
```

My code utilizes a single listener which has to decide which of the 13 buttons was pressed.

Question 3

(80 marks; 80 minutes)

The following Java application draw a series of large black dots onto a JPanel inside a JFrame:



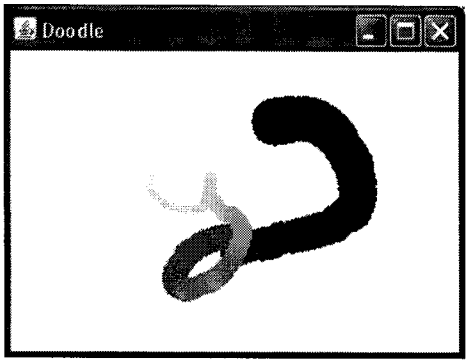
Dots are only added to the JPanel while the mouse is being *dragged* over the panel.

The dots are stored in an array with a maximum size of 200 elements. When a new dot is created which takes the total over 200, the oldest dot in the array is deleted, to make room for the new one.

a) Implement the application shown above.

Hints: your code should consist of three classes: one for the JFrame, one for the JPanel, *and* one for managing the array of dots (called PointsManager). My PointsManager class has two public methods called addPoint() and draw(). The graphics method for drawing a dot is Graphics.fillOval(). (60)

b) Modify PointsManager to draw points at different sizes and shades of gray, as shown below. When a new dot is *first drawn* in the panel, it is large and black. But, each time it is *redrawn*, it becomes a bit smaller and whiter.



Hints: this only requires a *few* changes to PointsManager.draw(), and the addition of some new global variables to PointsManager. Do **not** modify your JFrame or JPanel classes from part (a); do **not** write them out again in this answer.

The shades of gray (from black to white) require you to change the drawing colour of a dot using Graphics.setColor(). A new colour object is created using:

```
Color c = new Color(RedValue, GreenValue, BlueValue);
```

RedValue, GreenValue, BlueValue are floats that can range beween 0.0f and 1.0f. If all the values are 0.0f, then the resulting colour is black; if they are all 1.0f, then the colour is white. (20)

--- End of Examination ---