A20487 *No calculator permitted in this examination*

UNIVERSITY OF BIRMINGHAM

School of Computer Science

First Year – Degree of BSc with Honours
Artificial Intelligence and Computer Science
Computer Science
Computer Science with Business Management

First year – Degree of BEng/MEng with Honours
Computer Science/Software Engineering

First Year – Joint Degree of BEng/MEng with Honours
Electronic and Software Engineering

First Year – Joint Degree of BSc/MSci with Honours
Mathematics and Computer Science

06 18190

Software Workshop 1

Summer Examinations 2009

Time allowed: 3 hours

[Answer ALL Questions]

[Use a Separate Answer Book for EACH Section]

Section A

[Use a Separate Answer Book for this Section]

1 (a) The following for-loop has three parts missing from its first line. Write down that first line with the three parts filled in so that when the for loop runs it will print out the numbers 2, 4, 8, 16, ..., 1024. [3%]

```
for (??? ; ??? ; ??? ) {
    System.out.println(i);
}
```

(b) A class C has a void method m with no parameters. Another class D extends C and overrides m. Each class has a constructor with no parameters. In each of the following, say whether it is legal, and, if so, which definition of m will be used. Explain your answers.

(i)
```
C x = new D();
x.m();
```
(ii)
```
D x = new C();
x.m();
```

[4%]

2. When calculating x^n, i.e. x to the power n, the calculation can be split into two cases according to whether n is even ($n = 2k$ for some integer k) or odd ($n = 2k+1$ for some integer k). Here are the equations used, written for a slightly more general calculation of ax^n for some numbers a, x and n.

$$a * x^{2k} = a * (x^*x)^k$$
$$a * x^{2k+1} = (a^*x) * (x^*x)^k$$

In other words, the calculation of $a * x^n$ can be done by first halving n, squaring x and multiplying a by x if n was odd, thus reducing the calculation to a smaller n.

(a) As an example, show how this idea can be used to calculate $3 * 2^5 = 96$. For large values of n, why is this algorithm more efficient than simply multiplying a by x repeatedly n times? [3%]

(b) The following code is an attempt to write a recursive method based on the algorithm. (For convenience, line numbers have been included. Also, x^n is written as `x^n` in the typewriter font.)

```
/**                                                      //1
 * Calculate an expression a * x^n.                      //2
 * @param a the multiplying factor                       //3
 * @param x the number to be raised to a power           //4
 * @param n the power                                    //5
 */                                                      //6
public void aPower(double a, double x, double n) {       //7
  if (n%2 == 0) {      // n even                         //8
    aPower(a, x*x, n/2);                                 //9
  } else {                                               //10
    aPower(a*x, x*x, n/2);                               //11
  }                                                      //12
}                                                        //13
```

The definition will compile, but it still contains *five* issues. Three are errors that will prevent it running correctly (one of those needs correcting in several different places). The other two, in the declaration and Javadoc, affect how the method is used. For each of the five, say what the issue is, what its effect will be, and how to correct it.
Write out the corrected version in full. [9%]

(c) The above idea provides an efficient way to calculate powers, using `aPower(1,x,n)`. However, the recursion is not essential here – the same idea could be implemented in a while-loop. Briefly explain how and outline the code needed. [3%]

3 It is desired to develop a static method `arrayMin` to find the minimum of the elements in an array `a` of integers. The method will require that the array is non-empty. (This is because there is no sensible “minimum of no numbers”.) A while-loop will be used, with a control variable `i`, and on each iteration a variable `min` will be the minimum of the first `i` elements of the array. The while loop will take the following form.

```
/* loop invariant:
 * ...
 */
while (i < a.length) {
  ...
}
return min;
```

(a) Draw a diagram of the array to show the situation on a typical iteration. Your diagram should make clear the indexes of the first and last elements of the array, and the latest and next elements to be read. What are the initial and final values of `i`? [3%]

(b) Write a loop invariant to express this plan. Your invariant should contain enough information to allow you to know that when looping stops, `min` is the correct answer. Explain how you deduce this. [3%]

(c) How must `i` and `min` be initialized? [1%]

(d) Write the full definition of `arrayMin`, including Javadoc and the loop invariant.

[5%]

(e) Suppose the method is run in the debugger, with actual parameter the 4-element array {3, 4, -2, 1} and a breakpoint on the line “`while (i < a.length) {`“.

(i) How many times will the breakpoint be hit?

(ii) For each time it is hit, what are the values of `i` and `min`? Verify that the invariant is true.

[3%]

4 A project includes two Java classes, called Date and Book. In this question you will be required to write some parts of those classes.

(a) In the Date class, each instance stores a valid *date,* with integer fields for day, month and year. Each instance is to be *immutable,* which means that once it is created its field values cannot be changed.

Amongst its other members (fields and methods), the class will include the following.

Getter methods getDay, getMonth and getYear.

An array daysInMonthArray that stores the lengths of the months in a normal (non-leap) year. They are 31, 28, 31, 30, 31, 30, 31, 31, 30, 31, 30, 31.

A private method isValid with three parameters for day, month and year. It returns a boolean to show whether the date combination is valid.

A constructor with three parameters for day, month and year. It should throw an IllegalArgumentException if the parameters are an invalid combination.

Write declarations and definitions for those members as follows.

(i) Declarations for the fields day and daysInMonthArray. [3%]

(iii) Definitions for getDay and the constructor. [5%]

In each case, explain carefully why you make it public or private, whether it can be static, and (for the fields) whether it can be final.
Assuming that the constructor uses isValid, why is it desirable for isValid to be private? [1%]

(b) In the Book class, each instance stores a record of a library book. It includes a private field called returnDate, of type Date, for the date when the book is due to be returned (or null if the book is not on loan).

Write the definition, including Javadoc, for a method

```
public void renew
   (final int day, final int month, final int year)
```

to deal with the renewal of a book loan. It should print out the old return date on System.out, and then change the return date to that specified by the parameters. Include Javadoc. If the parameters are an invalid date, an error message should be printed to System.out. (Because isValid is private, it cannot be called in Book. Instead you must catch an exception.)

[6%]

5. A turtle graphics package uses a Java class Turtle, with methods that include

```
public void move(double length)
public void turn(double angle)
```

to make a turtle instance move the given length, or turn through the given angle (in degrees).

(a) An interface TurtleShape is for objects that can draw turtle shapes. It has a single void method drawMe, with a parameter specifying the turtle to be used. Write down the definition of the interface TurtleShape. [1%]

(b) Write the definition of a class MoveShape, implementing TurtleShape, whose instances move the turtle a given fixed length. The length is stored as a final field and initialized from a constructor parameter. There should not be a turtle field – explain why not. [4%]

(c) Write the drawMe methods for the following two more classes implementing TurtleShape.

(i) SequenceShape has a field sequence that is an array of TurtleShape objects. Its drawMe method draws the elements of the array in order. [2%]

(ii) RepetitionShape has two fields: repeatBody, of type TurtleShape, and repeatNumber, of type integer. Its drawMe method draws the repeatBody shape repeatNumber times. [2%]

For the rest of the question, you should assume that in both these cases, the fields are initialized from constructor parameters.

(d) Write a static method makePolygon that creates a TurtleShape object that draws a regular polygon. The side length and number of sides are supplied as parameters to makePolygon, and the TurtleShape object is returned as result. Include Javadoc, with "requires conditions" (often known as preconditions) as appropriate.

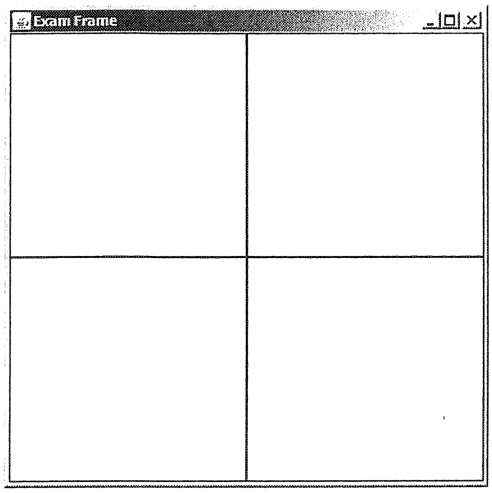
Do not define any new classes that implement TurtleShape. You may use MoveShape, TurnShape (just like MoveShape, but doing a turn instead of a move), SequenceShape and RepetitionShape. [5%]

(e) Given a variable theTurtle of type Turtle, write a single line of code that uses makePolygon to make theTurtle draw a 5-sided polygon with side length 150. [1%]

Section B

[Use a Separate Answer Book for this Section]

6. The window illustrated contains a Panel described by the following Java class:



```
import java.awt.*;

public class ExamPanel
              extends Panel {

   ExamCanvas canv1, canv2,
              canv3, canv4;

   public ExamPanel () {
      setLayout(
          new GridLayout(2,2));
      canv1 = new ExamCanvas();
      canv2 = new ExamCanvas();
      canv3 = new ExamCanvas();
      canv4 = new ExamCanvas();
      add(canv1);  add(canv2);
      add(canv3);  add(canv4);
   }

}
```

An ExamPanel object contains four ExamCanvas objects described by the following Java class.

```
import java.awt.*;

public class ExamCanvas extends Canvas {

   public void paint(Graphics g) {
      Dimension d = getSize();
      g.drawRect(0,0,d.width-1,d.height-1);
   }

}
```

(a) Write a Java class ExamFrame that can be used to create windows like the one illustrated above (of size 400x400 pixels), and write a main method to display one such window.

[4%]

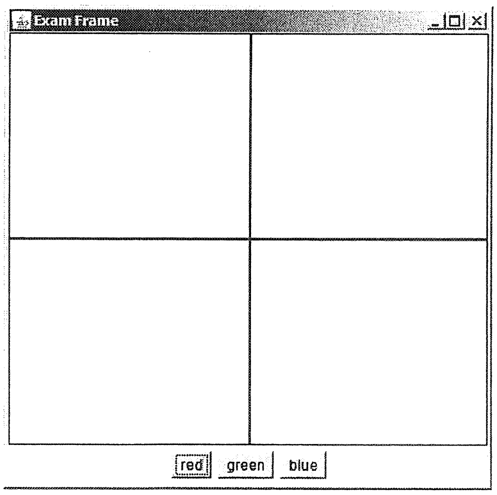
(b) Now modify the given `ExamCanvas` class so that it has an attribute called `highlight` of type `java.awt.Color` with a default value `Color.RED`. (`setHighlight` and `getHighlight` methods should also be defined.) Each time the mouse pointer enters the region on the screen defined by the canvas, the background colour of the canvas should change to the current highlight colour and when the mouse exits the region, the background colour should change back to white. You will need the `MouseListener` interface, from the Java API, which is defined as:

```
package java.awt.event;
import java.util.EventListener;

public interface MouseListener extends EventListener
{
   public void mouseClicked(MouseEvent e);
   public void mousePressed(MouseEvent e);
   public void mouseReleased(MouseEvent e);
   public void mouseEntered(MouseEvent e);
   public void mouseExited(MouseEvent e);
}
```

[5%]

(c) Now modify your Frame class so that the window displayed includes a bottom `Panel` with three buttons as shown. Your `ExamPanel` class should be modified to implement `ActionListener` so that it can respond to clicks on the buttons by setting the highlight colour for each of its canvas objects to the colour specified on the button.



The following is a listing of the interface class ActionListener from the Java API:

```
package java.awt.event;

import java.util.EventListener;

public interface ActionListener extends EventListener {

    public void actionPerformed(ActionEvent e);

}
```

[7%]

7. (a) The heading for the class TreeSet in the Java library package java.util includes the following elements:

```
public class TreeSet<E> ...
```

The method add defined in the class TreeSet starts with:

```
    public boolean add(E e) {
```

Explain the use of the type E in the above code fragments.

[2%]

(b) Write a declaration for a TreeSet variable and construct a TreeSet object that can be used to store a set of words (Strings) in alphabetical order.

[2%]

(c) Explain briefly using a diagram how data is organised in a TreeSet.

[3%]

(d) A small shop has a very simple stock control system. An item for sale is currently represented by a Java object constructed from the class:

```
public class StockItem
{
   private String stockCode, description;
   private double price; // price per item
   private int quantity; // no of items in stock

   public StockItem(String s, String d, double p, int q)
   {  setStockCode(s);  setDescription(d);
      setPrice(p);  setQuantityInStock(q);
   }

   // plus get and set methods for
   // stockCode, description, price, quantity

}
```

The entire stock is going to be stored in a TreeSet in alphabetical order of stockCode using the following class:

```
import java.util.*;

public class StockSet extends TreeSet<StockItem> {
  ...
  ...
}
```

What additions need to be made to StockItem before the inherited add method can be successfully used?

[3%]

(e) Write, for inclusion in the class `StockSet`, a method `outOfStock` that returns a `String` listing the `stockCode` values of all items for which the `quantityInStock` value is `0`.

[3%]

(f) The interface `Map` in the Java library package `java.util` includes the following:

```
public interface Map<K,V>
{
    V put(K key, V value);

    V get(Object key);
```

Instead of just storing a *set* of stock items as in (d) and (e), it is required to store the stock in such a way that a `stockCode` `String` can be used to look up the corresponding `StockItem` object. Demonstrate how a `TreeMap` (which implements the `Map` interface) could be used to provide this functionality.

[4%]