UNIVERSITY *of* LIMERICK

OLLSCOIL LUIMNIGH

COLLEGE *of* INFORMATICS *and* ELECTRONICS
Department of Computer Science and Information Systems

End of Semester Exam

Academic Year:	2006/2007	**Semester:**	Autumn
Module Title:	Imperative Programming 1	**Module Code:**	CS4411
Exam Duration:	2½ Hours	**% of Total Marks:**	75
Lecturer:	A. McElligott		

Instructions to Candidates:

Section A:	ALL questions should be attempted.	55%
Section B:	ONE question should be attempted. ALL questions in this section carry equal marks.	20%

State clearly any assumptions you make.

Section A: all questions should be attempted

Q1 **(13 marks)**

(a) What output is produced to screen on executing

(i) ExamDryRun1, ***[2.5 marks]***

(ii) ExamDryRun2 and ***[2.5 marks]***

(iii) ExamDryRun3 (cf. Figure A1.1)? ***[2.5 marks]***

```
public class ExamDryRun1
{
  public static void main(String[] args)
  {
    short i = 2;
    while (i < 8)
    {
      if (i <= 4) i++;
      else i *= 2;
      System.out.println(i + " ");
    }
  }
}
```

```
public class ExamDryRun2
{
  public static void main(String[] args)
  {
    double i = 2.5;
    do
    {
      i *= 2;
      System.out.println(i);
    } while (i > 3 && i <= 10);
  }
}
```

```
public class ExamDryRun3
{
  public static void main(String[] args)
  {
    for (short i = 0; i < 5; i++)
      System.out.print(i % 2 + ";");
  }
}
```

Figure A1.1

Please turn over →

(b) What output is produced to screen when the program in Figure A2.1 is executed? You should assume that the user enters the input as shown in Figure A2.2. ***[5.5 marks]***

```
import javax.swing.JOptionPane;
public class ExamDryRun4
{
  public static void main(String[] args)
  {
    String r = "", fC = "", fruit;
    do
    {
      fruit = JOptionPane.showInputDialog(null,
                          "Please enter a fruit");
      if (!(fruit.substring(0,1).equals(fC)))
      {
        if (!(fC.equals("")))
          r += "\n";
        fC = fruit.substring(0,1);
        r += "   " + fC + "\n";
      }
      r += fruit + "\n";
     } while (!(fruit.equals("Orange")));
    JOptionPane.showMessageDialog(null,r,"Output",
                        JOptionPane.INFORMATION_MESSAGE);
  }
}
```

Figure A2.1

```
Apple
Apricot
Avocado
Banana
Blueberry
Grape
Lemon
Lime
Orange
```

Figure A2.2

Q2 **(14 marks)**

This question relates to the project work you undertook during this semester.

(a) Rewrite the method in Figure A3.1 using **for** constructs instead of the **while** constructs. In your rewrite you should ensure that only the minimum number of initialization, condition and action parts of any **for clause** are null.

[3 marks]

```
public static void generateUniqueHand(int[] cards)
{
  int deckSize = 52, index = 0, duplicateIndex;
  while (index < cards.length)
  {
    cards[index] = (int) (Math.random() * deckSize);
    duplicateIndex = 0;
    while (cards[duplicateIndex] != cards[index])
      duplicateIndex++;
    if (index == duplicateIndex)
      index++;
  }
}
```

Figure A3.1

(b) You are required to write methods to achieve each of the following.

(i) The method cardsOfSameSuit() accepts an array containing the suit of each of the five cards dealt. It returns true if these five cards are of the same suit otherwise it returns false. Your method should not undertake any unnecessary comparisons.

[3 marks]

(ii) The method cardsInConsecutiveDescendingSequence() accepts an array containing the value of each of the five cards dealt. These values are in descending sequence. This method returns true if these values are in consecutive descending sequence otherwise it returns false. Your method should not undertake any unnecessary comparisons.

[3 marks]

(iii) The method checkOtherPossibleCombinations() accepts an array containing the values of each of the five cards dealt. These values are in descending sequence. This method should return

- 6 if the array of values represents four of a kind
- 4 if the array of values represents a full house
- 3 if the array of values represents three of a kind
- 2 if the array of values represents two pair
- 1 if the array of values represents one pair
- 0 if the array does not represent any of the winning hands mentioned here

Your method should not undertake any unnecessary comparisons.

[5 marks]

Q3 **(14 marks)**

A pangram is a sentence that contains each character of an alphabet in either uppercase or lowercase form. You are required to write a Java application program that requests a sentence from the end user (cf. Figure A3.1) and reports whether or not this sentence is a pangram in accordance with the English alphabet (cf. Figures A3.3 and A3.4). If no input is provided your program should display output similar to that shown in Figure A3.2). Your program should not undertake any unnecessary comparisons.

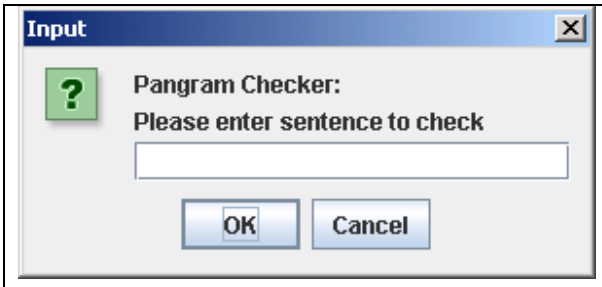

Figure A3.1

Figure A3.3

Figure A3.2

Figure A3.4

Q4 **(14 marks)**

The Universal Product Code (UPC) is a 12-digit number and bar code that identifies an individual consumer product (cf. Figure A4.1). The first six digits are a manufacturer identifier, the next five digits identify a specific product and the last digit is a check digit. The last digit lets the scanner determine if it scanned the number correctly. Taking the bar code in Figure A4.1 as an example the following are the steps used to calculate the check digit from the first 11-digits.

Step 1: Add together the value of all of the digits in odd positions (digits 1, 3, 5, 7, 9 and 11).
8 + 1 + 7 + 0 + 1 + 3 = 20
Step 2: Multiply the result of Step 1 by 3.
20 x 3 = 60
Step 3: Add together the value of all of the digits in even positions (digits 2, 4, 6, 8 and 10).
0 + 1 + 0 + 0 + 2 = 3
Step 4: Add the results from Step 2 and Step 3.
60 + 3 = 63
Step 5: Take the result of Step 4. To create the check digit, determine the number that, when added to the number in Step 4, is a multiple of 10.
63 + 7 = 70
(Note: the calculated check digit is 7**)**
Step 6: If the check digit from Step 5 (i.e., 7 in this case) is equal to the last digit of the12-digit entry (i.e., in this case 7) this implies that the 12-digit entry is a valid UPC.

Thus, when scanning an item the scanner performs this calculation. If the check digit it calculates is different from the check digit it reads, the scanner knows that something went wrong and the item needs to be rescanned.

You are required to write a Java application program that will allow the end user to enter a 12-digit number and your program will report whether or not this number is actually a valid UPC. If no input is received from the end user or if the format of the user input is incorrect your program should display appropriate error messages. If the end user enters a UPC of the correct format your program should report whether this UPC is valid or invalid by displaying appropriate messages.

Figure A4.1

Section B: attempt one question

Q1 **(20 marks)**
(a) You are required to write a Java application program to accept a telephone number in its Irish national format. A number in its Irish national format comprises an area code followed by an actual telephone number. Your program should convert any valid number input into its equivalent international format. You may assume that an area code comprises 2, 3 or 4 digits where the first digit is always a 0. If an area code has two digits the second will always be in the range 1 to 9. If an area code has three digits the second two digits will always be in the range 1 to 9. If an area code has four digits the second and fourth digits will always be in the range 1 to 9 with the third digit always being a 0. The actual telephone number will always be 6 digits or 7 digits in length.
The end user should be required to surround the area code by round parenthesis when entering a number in its national format. Examples of valid entries include: (061)1234567, (01)123456, and (0405)123456 while examples of invalid entries include: 061-1234567, (01234)123456, and (021)1234.
The international equivalent of a number begins with +353- followed by the area code but excluding the first digit of the area code, followed by a - and finally the telephone number. For example, the international equivalent of (061)1234567 is +353-61-1234567 while the international equivalent of

(01)123456 is +353-1-123456. If the user fails to enter data or the input is not of the correct format appropriate error messages should be displayed. You should use an input dialog box to accept data and a message dialog box to display data. ***[8 marks]***

(b) Write a Java application program that requests a sentence from the end user. Ignoring punctuation and capitalization your program should report whether or not this sentence is a palindrome (on a word by word basis). For example,

- I am; therefore, am I?
- You can cage a swallow, can't you, but you can't swallow a cage, can you?

would be palindromes since they read the same word by word forward and backward whereas

- Monkey see, monkey do.

would not be a palindrome on a word by word basis.

[12 marks]

Q2 **(20 marks)**

(a) Show how the following data items would be ordered in descending sequence using the bubble sort algorithm. You are expected to show for each pass the comparisons made and any exchanges undertaken.

6, 9, 3, 4, 2, 7, 5 ***[10 marks]***

(b) Assuming the existence of the code in Figure B2.1 you are required to write the code for each of the methods referred to in the main() method. The first of these methods should allow the end user to enter seven integer values similar to those in part **(a)** of this question. The second method should accept these values and order them in descending sequences. Your method for ordering the data should not continue to pass through the data once it has been determined that the data is already ordered. The third method should accept the ordered items and write the contents of the ordered array to the standard output device. ***[10 marks]***

```
import javax.swing.JOptionPane;
public class OrderData
{

  public static void main(String[] args)
  {
    int[] data = new int[7];
    loadData(data);
    orderData(data);
    displayData(data);
  }

}
```

Figure B2.1