Ollscoil na hÉireann, Gaillimh
National University of Ireland, Galway

GX 1596

Spring Examinations, 2002/2003

Exam Code(s)	3IF1, 3BP1
Exam(s)	Third Year Information Technology Third Year Electronic and Computer Engineering
Module Code(s)	CT326
Module(s)	Programming III
Paper No.	1
External Examiner(s)	Prof. P. Nixon
Internal Examiner(s)	Prof. G. Lyons Dr. D. Chambers
Instructions:	Answer any 5 questions. All questions will be marked equally.
Duration	3hrs
No. of Answer Books	1
No. of Pages	5
Department(s)	Information Technology

1:a: Using a simple example, discuss why casting a superclass reference to a subclass reference is potentially dangerous. 5 MARKS

b: What is the difference between **abstract** classes and interfaces? Should all the methods in an **abstract** superclass be declared **abstract**? 5 MARKS

c: Consider the inheritance hierarchy of **Figure** 1 below. For each class, indicate some common attributes and methods consistent with the hierarchy. Assume that a CurrentAccount provides overdraft facilities and that a StudentAccount is similar to a Current Account but has no transaction charges. Write simple Java implementations for each of the classes shown. The base Account class should be declared as an abstract class. 10 MARKS

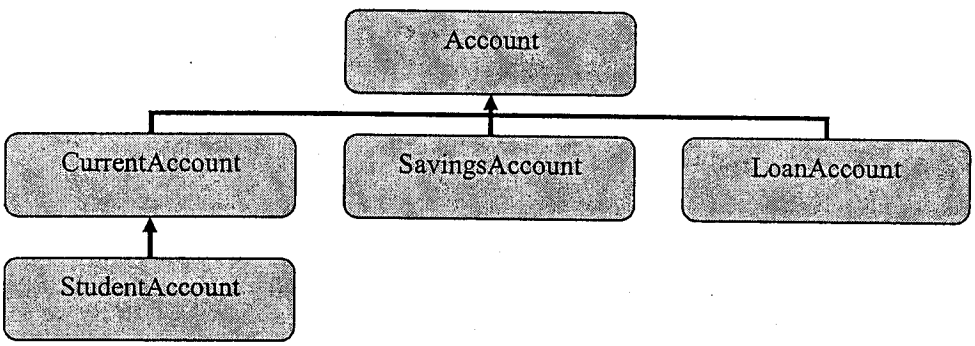

Figure 1 - Inheritance Hierarchy for Bank Accounts

2.a: Describe briefly the general structure of the IO Streams classes provided in the Java programming environment. 5 MARKS

b: Develop a simple GUI-based Java program that may be used to control a washing machine. Use suitable Swing components to allow the washing machine operator to perform the following functions:

1) Switch the machine on.
2) Choose a temperature from a list.
3) Spin speed selection buttons - can be 600, 800 or 122 RPM.
4) Display the current status of the wash cycle.

Show the top-level design of the GUI, including any Panels and related Layout Manager objects that you propose to use. For each of the components you've chosen above, write the code to construct the component, add the component to a container and then setup simple event handling for the component (for those that generate events). The event handlers need only print out a message indicating that they have been called. 15 MARKS

3.a: Identify and correct the errors in each of the following program segments. [*Note:* There may be more than one error in each piece of code]:

i:
```
FileReader reader = FileReader("Data.txt");
String line = readLine();
```
2 MARKS

ii:
```
public void toString()
{
  String s = "Some", "thing";
  return s;
}
```
2 MARKS

iii:
```
int years[] = {1997 1998 1999};
for (int i = 2; i >=0; i--)
  System.out.println(years[i]);
```
2 MARKS

iv:
```
int x = 0;
while (x < 10)
  System.out.println("x = " x);
```
2 MARKS

v:
```
public interface Moveable implements Serializable
{
  public forward(int speed);
  public reverse(int speed);
}
```
2 MARKS

b: Write a Java application that inputs a date as a string in the form 23/02/2002. The program should use an object of class *StringTokenizer* to extract the various components of the date string as tokens. The program should then convert the day, month and year to int values and display them. 10 MARKS

4. Design and Implement a simple Java application to support the basic operations of a Video Library. The library has items for rental including DVDs and Video Tapes. The application should be able to manage information about the members of the club, the items available and also allow members to rent or return these items.

The following guidelines should be used to construct the application:

a: A Java class, called **Member**, should be defined to store and manage member details. The class should include methods for updating member details and querying their status i.e. have they any items currently rented and are there any arrears due on their account. Each member of the club should have a unique id number, this number can be assigned when the member object is created. This class should also include methods to allow a member to rent and return items.
(Q.4 continued on next page) 6 MARKS

b: Define a Java class, called **RentalItem** to hold details about DVDs and Video Tapes that are available in the library. This class should include methods to update the current rental status of the item. Similarly to the Member class, each item should have a unique id number assigned when the object is created. 5 MARKS

c: Define another Java class, called **VideoLibrary**, that will be used to manage the membership and available items. Member and RentalItem objects added to the library should be stored using suitable collection objects. VideoLibrary should also include methods for managing items and members. 6 MARKS

d: Write a short driver program that creates an instance of VideoLibrary and uses its methods to add some new members and items to the library. 3 MARKS

5: Create a class called **Complex** for performing arithmetic with complex numbers. Complex numbers have the form:

realPart + imaginaryPart * *i* where *i* is the square root of -1.

a: Use floating-point variables to represent the **private** data of the class. Provide a constructor method that enables an object of this class to be fully initialized. Also provide a no-argument conctructor with default values in case no initial values are provided. 4 MARKS

b: Provide a **public** method to add two **Complex** numbers: the real parts are added together and the imaginary parts are added together to create the result. This method should return a new **Complex** object initialized with the result e.g. if **c1** and **c2** are objects of type **Complex**, calling **c3 = c1.add(c2)** would add the value of **c2** to **c1** and then return a new object initialized with the result. The original values of **c1** and **c2** would not change. 5 MARKS

c: Provide a **public** method for subtraction of two **Complex** numbers: the real part of the right operand is subtracted from the real part of the left operand, the imaginary part of the right operand is subtracted from the imaginary part of the left operand. In the same way as for (b), this method should also return a new **Complex** method initialized with the result e.g. if **c1** and **c2** are objects of type **Complex**, calling **c3 = c1.subtract(c2)** would subtract the value of **c2** from **c1** and then return a new object initialized with the result. The original values of **c1** and **c2** would not change 5 MARKS

d: Provide a **public** method for printing **Complex** numbers in the form **(a+b*i*)** where **a** is the real part and **b** is the imaginary part. 3 MARKS

e: Write a short driver program to test your class. 3 MARKS

6.a: By extending class **Vector**, Java's designers were able to create a **Stack** class quickly. Describe briefly the negative aspects of this use of inheritance, particularly for class **Stack**. 4 MARKS

b: Desribe the functionality provided by the following Java code. Would it have been possible to write similar code using an *Enumeration* instead of an *Iterator* ? Explain your answer.

```
import java.util.*;

   static void filter(Collection c) {
      for (Iterator i = c.iterator(); i.hasNext(); )
            if (!cond(i.next()))
               i.remove();
      }
```

8 MARKS

c: Write a Java applicaion that prompts the user to input their Name, Address, Date of Birth and Student ID number using the standard input *System.in* - this information should then be saved to a file named *studentData*. The program should use the *FileWriter* class and an appropriate processing stream to handle the data output. 8 MARKS

7.a: Discuss briefly the differences between a process and a thread. How should executing threads be stopped (assuming they still haven't finished their work)? 4 MARKS

b: Write a JAVA applet that continuously scrolls a text message across the screen from left to right. The message itself and the rate at which the text scrolls should be passed to the applet as HTML based parameters. 8 MARKS

c: Outline the design of the Java class for an object that will be used as a buffer to hold an integer value. The value may be updated randomly by a Producer thread, provided that it has already been consumed by a Consumer thread. Each value produced may be consumed at most once and there may be multiple producer and consumer threads executing (and attempting to access the buffer) concurrently. 8 MARKS