Ollscoil na hÉireann, Gaillimh *GX_____*
National University of Ireland, Galway

Spring Examinations, 2008/2009

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. J. Keane
Internal Examiner(s)	Dr. D. Chambers Prof. G. Lyons

Instructions: Answer any 4 questions.
All questions carry equal marks.

Duration	3hrs
No. of Answer Books	1
No. of Pages	5
Department(s)	Information Technology

1. The following Java code provides the outline of a simple bank account class:

```
import java.io.*;
public class Account implement Serializable{
        protected HolderDetails holder;
        protected List transactions;
        protected float balance;

        // Add a suitable constructor here

        // Add methods to make deposits / withdrawls

        // Add a method to print out summary of account transactions

        // Add suitable accessor methods for retrieving / updating attributes
}
```

   a: Complete the implementation of the Account class, providing a suitable constructor, attribute accessor methods, methods for making deposits / withdrawls and a method to print out a transaction summary.
   7 MARKS

   b: Provide implementations for the HolderDetails class and a suitable class to represent transactions. The HolderDetails class is used to store details about the account holder. The Transaction class contains details about past transactions.
   7 MARKS

   c: Define and implement a new class, called CurrentAccount, derived from Account, that allows withdrawls to proceed up to some overdraft limit.
   5 MARKS

   d: The attributes of class Account are defined as *protected*. What is the implication of this definition? 3 MARKS

   e: What does the statement *implements Serializable* mean? What are the implications of this statement? 3 MARKS

2.a: Describe the general structure and purpose of the IO Streams classes provided in the Java programming environment. Is there any mechanism to support random file access in Java? 4 MARKS

b: Write a Java application that inputs a date as a string in the form 17:02:2009 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. 6 MARKS

c: Write a simple Employee class that includes an id number, a name, and salary details and a suitable constructor method. Then write a Java program that uses an ArrayList to store a collection of Employee objects. Also, write the code for a Comparator class i.e. a class that implements the Comparator interface, that can be used to compare two Employee objects based on their id number. Finally, use the version of the Collections.sort() method that allows you to pass your own Comparator object to sort the list of Employee objects. 15 MARKS

3.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)? 5 MARKS

b: Show (using simple code examples) how threads may be created (and started) using the following mechanisms:

(i) Application class extends the Thread class.

(ii) Application class implements the Runnable interface.

Assume you have a bank account class that may be accessed by more than one thread of execution simultaneously. Show how the various methods of the class may be made thread safe. 10 MARKS

c: Outline the design and code implementation 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 one or more Producer threads, provided that it has already been consumed by one of a number of Consumer threads. Each value produced must be consumed at exactly once and there may be multiple producer and consumer threads executing (and attempting to access the buffer) concurrently. 10 MARKS

4.a: The Java Collections framework provides two general-purpose implementations of the *List* interface i.e. *ArrayList* and *LinkedList*. Are there any circumstances under which *LinkedList* might offer better performance than *ArrayList*? Explain the purpose of the following code idiom:

```
int pos = Collections.binarySearch(l, key);
if (pos < 0)
      l.add(-pos-1, key);
```

10 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

5.a: What types of Sockets are supported in the Java networking package and which type of Socket would you recommend for a VOIP type application and a File Transfer type application? Write a simple Java program that uses Datagram type sockets to exchange numeric values i.e. one side sends a single integer value to the other and it then receives the same number back as a response.

10 MARKS

b: Write a network Server program in Java where the Server waits for incoming client connections using stream type sockets. Once a Client connects it sends a text string to the server with a simple query – the server then responds with a text based response. The connection is then terminated. The server should use a separate thread of execution for each new client connection and all interaction between the Server and the Client should be done within this thread.
The answer should include full source code for the server application.

15 MARKS

6. Assume that a Sports Club wishes to store details about its members. Design and implement a Java application to support this requirement. The application should be able to print out and manage information about the members of the club. 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 subscription status i.e. are they fully paid up club members. Each member of the club should also have a unique membership id number, this number is automatically assigned when the member object is created. 10 MARKS

   b: Define another Java class, called SportsClub, that will be used to manage club membership and access details about individual members. Member objects added to the SportsClub should be stored using a suitable collection object. SportsClub should include methods for adding new members, removing members, getting a list of current members and accessing information about an individual member (based on their id number). 10 MARKS

   c: Write a short driver program, in a class called ClubManager, that creates an instance of SportsClub and uses its methods to add, lookup and remove club members. 5 MARKS