OLLSCOIL NA GAILLIMHE
UNIVERSITY OF GALWAY

Autumn Examinations 2022-2023

Course Instance Code(s)	3BCT1, 3BP1, 3BP4
Exam(s)	Third Year Computer Science & Information Technology Third Year Electronic and Computer Engineering
Module Code(s)	CT326
Module(s)	Programming III
Paper No.	1
External Examiner(s)	Dr Ramona Trestian
Internal Examiner(s)	Professor M. Madden *Dr. Adrian Clear

Instructions: Answer any 4 questions. All questions will be marked equally.

Duration	2 hours
No. of Pages	5
Discipline(s)	Computer Science
Course Co-ordinator(s)	Dr. Colm O'Riordan

Requirements:

Release in Exam Venue	Yes [X]	No [ ]
MCQ Answersheet	Yes [ ]	No [X]
Handout	None	
Statistical/ Log Tables	None	
Cambridge Tables	None	
Graph Paper	None	
Log Graph Paper	None	
Other Materials	None	
Graphic material in colour	Yes [ ]	No [X]

PTO

Q1:

(a) Explain the difference between threads and processes. Describe with code examples **two** ways of creating and starting a new thread in Java.

10 MARKS

(b) Outline the design and code implementation of a thread-safe Java class for an object that will be used as a buffer to hold an `int` value. After an initial value has been set, 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 exactly once and there may be multiple producer and consumer threads executing (and attempting to access the buffer) concurrently.

15 MARKS

Q2.

(a) List and describe the activities involved in the main stages of a test driven development approach.

5 MARKS

(b) You have been asked to write a class to represent a bank account called `Account`. Write the **unit tests** you would use to test the following functionality.

(i) It should be possible to make a deposit of funds to the account as long as the deposit value is positive.

(ii) It should be possible to make a withdrawal from the account as long as there are sufficient funds in the account. If there are insufficient funds in the account, an appropriate exception should be thrown.

(iii) An `Account` is represented with an account number and a balance. It should be possible to serialize an `Account` (including all its member variables) to a file.

15 MARKS

(c) Assume you have a `List` containing `Account` objects, and that the `Account` class has an appropriate `toString()` method implementation. Use a call to the `Iterable forEach` method to print the `Account` objects to the standard output stream.

5 MARKS

PTO

Q3.

Demonstrate using code examples how you would make use of an `Iterator` to count the number of elements in a `Collection` of strings that contain at least one capital letter, lowercase letter, and number.

12 MARKS

Demonstrate using code examples how you would use a `List` to represent the students registered to a module. You should sort the `List` in such a way that the order of students is based on the lexicographic order of their surname followed by their first name.

Illustrate the code for adding students called Steve Higgins and Mary Higgins to the `List` before sorting it.

What would the program output be if you print the `List` to the console?

Note: you can represent a student by their surname and first name only. Your code should include a `toString()` method that represents a `Student` as "[surname], [first name]"

13 MARKS

PTO

Q4:

(a) Describe the `Set` interface in the Java Collections Framework. What is its relationship to the `Collection` interface? What are the main characteristics of a `Set` collection? List and briefly describe two classes in the Java Collections Framework that implement the `Set` interface, outlining the difference between them.

6 MARKS

(b) Explain fully the purpose and operation of the following code idiom:

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

6 MARKS

(c) Write a `Plant` class that includes an ID number, a `genus` name and a `species` name as class attributes.

Write an appropriate `hashCode()` method for the class based on these three values.

Write an appropriate `equals()` method where two `Plant` instances are considered equal if they have the same ID, species and genus.

Write a `toString()` method that provides an appropriate *formatted* string representation of a `Plant` instance using its three attributes.

13 MARKS

PTO

Q5.

Write a Java class called `Employee` that has the following class attributes: `int idNumber, String name, LocalDate startDate, String jobTitle, float weeklyPay`

(a) The `Employee` class should include a suitable `writeObject()` method, to implement custom object serialisation, that writes out all attributes except the weekly pay to the `ObjectOutputStream` passed to the `writeObject()` method. The weekly pay should instead be appended, in text format, to a CSV (Comma Separated Values) file called payroll.csv in the format "idnumber, weeklypay". Each line of the payroll.csv file will therefore look something like this:

3249930, 420.80

7 MARKS

(b) The `Employee` class should also include a suitable `readObject()` method, that complements the `writeObject()` method, to read all attributes except the weekly pay from the `ObjectInputStream` passed to the `readObject()` method. The weekly pay should instead be read, in text format, from the CSV (Comma Separated Values) file previously created by the `writeObject()` method. The last matching entry, that is an entry with a corresponding idnumber, in the CSV file should be used to set the weeklypay value.

8 MARKS

(c) Write a Java program that creates two employees. The program should then write out the `Employee` objects, using Object Serialisation, to a file before reading the `Employee` objects from the file, again using Object Serialisation.

10 MARKS

END

NUI Galway
OÉ Gaillimh

Autumn Examinations 2021-2022

Course Instance Code(s)	3BCT, 3BP
Exam(s)	Third Year Computer Science & Information Technology Third Year Electronic and Computer Engineering
Module Code(s)	CT326
Module(s)	Programming III
Paper No.	1
External Examiner(s)	Dr Ramona Trestian
Internal Examiner(s)	Prof Michael Madden *Dr Adrian Clear

Instructions: Answer any 4 questions. All questions will be marked equally.

Duration	2 hours
No. of Pages	5
Discipline(s)	Computer Science
Course Co-ordinator(s)	Dr Colm O'Riordan

Requirements:

Release in Exam Venue	Yes [X]	No [ ]
MCQ Answersheet	Yes [ ]	No [X]
Handout	None	
Statistical/ Log Tables	None	
Cambridge Tables	None	
Graph Paper	None	
Log Graph Paper	None	
Other Materials	None	
Graphic material in colour	Yes [ ] No [X]	

Q1:

(a) Implement an Enum in Java called `CouncilTaxBand` which enumerates the following Council Tax bands for property owners based on their property value.

A – above £100,000
B – £100,001 to £150,000
C – £150,001 to £200,000
D – £200,001 to £300,000
E – £300,001 to £500,000

Your Enum class should include a method that takes an integer parameter representing a property value, and returns the band (A, B, C, D, or E) that the value fits within, otherwise `null`.

10 MARKS

(b) Write a `Property` class that includes an address, a value, a number of bedrooms, a number of bathrooms, and a `CouncilTaxBand` as class attributes. The `Property` class should implement the `Comparable` interface to define the natural order for these objects such that the `CouncilTaxBand` is compared first and then the property value.

Write a Java program that uses an `ArrayList` to store a collection of `Property` objects and then sort the list based on natural order.

Also, write the code for a `Comparator` class i.e., a class that implements the `Comparator` interface, that can be used to compare two `Property` objects based on the property value first and then based on the number of rooms.

Finally, use the version of the `Collections.sort()` method that allows you to pass your own Comparator object to re-sort the list of `Property` objects.

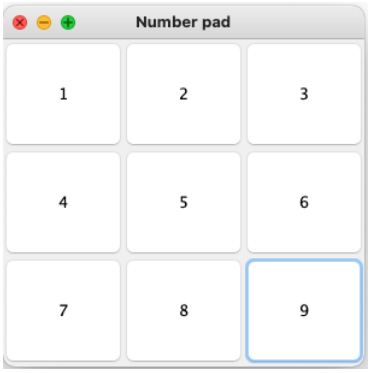
15 MARKS

Q2:

(a) List and describe with the use of diagrams **three** layout managers that can be used to organise components in Swing applications. For each one, provide a code example of how to add a component to a container that has been defined to use that layout manager.

10 MARKS

(b) Write a Java program to produce the GUI below using Swing, which illustrates a Number Pad. When a button is clicked by the user (like button 9 in the figure), it should print the button's number to the console with a message like "Button 9 pressed". The GUI should have a size of 300x300 pixels, a title, and should terminate the application when the close button ('x') is pressed.



15 MARKS

Q3:

(a) The JDK contains two general-purpose List implementations i.e. *ArrayList* and *LinkedList*.
Why is *ArrayList* generally the best performing implementation? Describe the circumstances under which *LinkedList* might offer better performance.

Describe three of the polymorphic algorithms provided in the Java Collections framework. In relation to these algorithms, explain fully the purpose and operation of the following code idiom:

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

13 MARKS

(b) Explain using an example what *serialization* is in Java. Your answer should address issues of serializable objects, deserialization, and custom serialization in detail.

12 MARKS

Q4:

(a) Show using a code example how a thread may be created (and started) using an application class that implements the `Runnable` interface.

Include a mechanism in the `Runnable` class to allow it to be shutdown gracefully (i.e., without needing to call the `stop()` method).

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

10 MARKS

(b) Outline the design and code implementation of the Java class for an object that will be used as a buffer to hold a `String` object. The contents of the `String` may be written 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 exactly once and there may be multiple producer and consumer threads executing (and attempting to access the buffer) concurrently.

15 MARKS

Q5:

(a) Explain using examples the difference between Data Sink Streams and Processing Streams.

5 MARKS

(b) Write a Java class called `RAFQ` that uses a `RandomAccessFile` to store diary entries of the format [date][diary entry]. For example, “2022-02-17”“Today is windy!”, where the quotes illustrate an individual UTF string. Your program should include the following two methods for writing a diary entry to and reading a diary entry from the file, respectively:

```
public void writeDiaryEntry(LocalDate timestamp, String diaryEntry)

public String getDiaryEntry(LocalDate timestamp)
```

An example of how the class is intended to be used is as follows. This example would print “Today is dry” to the console:

```
RAFQ raf = new RAFQ("mydiary.txt");
LocalDate timestamp = LocalDate.now();
raf.writeDiaryEntry(timestamp, "Today is sunny");

LocalDate anothertimestamp = LocalDate.parse("2022-01-03");
raf.writeDiaryEntry(anothertimestamp, "Today is dry");

LocalDate andanothertimestamp = LocalDate.parse("2022-04-12");
raf.writeDiaryEntry(andanothertimestamp, "Today is windy!");

System.out.println(raf.getDiaryEntry(anothertimestamp));
```

20 MARKS

END

NUI Galway
OÉ Gaillimh

Semester 1 Examinations 2021-2022

Course Instance Code(s)	3BCT, 3BP
Exam(s)	Third Year Computer Science & Information Technology Third Year Electronic and Computer Engineering
Module Code(s)	CT326
Module(s)	Programming III
Paper No.	1

External Examiner(s) Dr Ramona Trestian
Internal Examiner(s) Professor M. Madden
*Dr Adrian Clear

Instructions: Answer any 4 questions. All questions will be marked equally.

Duration 2 hours
No. of Pages 5
Discipline(s) Computer Science
Course Co-ordinator(s) Dr Adrian Clear

Requirements:

Release in Exam Venue	Yes [X]	No [ ]
MCQ Answersheet	Yes [ ]	No [X]
Handout	None	
Statistical/ Log Tables	None	
Cambridge Tables	None	
Graph Paper	None	
Log Graph Paper	None	
Other Materials	None	
Graphic material in colour	Yes [ ] No [X]	

PTO

Q1: Write a Java class called `Book` that has the following class attributes: `String title`, `Author author`, `LocalDate published`, `String isbn`

i. Make it possible to serialize `Book` objects. You should assume that the `Author` class is not serializable. However, you are required to include the `Author` data when serialising `Book` objects. Therefore, your class should indicate that the author attribute is to be ignored as part of default serialisation, and you must implement a suitable `writeObject()` method to perform custom serialisation. It should write all attributes of the Book class to the `ObjectOutputStream` passed to the `writeObject()` method. It should also write the individual attributes of the `author` attribute, which can be obtained from the following methods in the `Author` class:

```
public String getFirstName()
public String getLastName()
public LocalDate getDateOfBirth()
```

7 MARKS

ii. The `Book` class should also include a corresponding `readObject()` method to perform custom deserialization, i.e., it should read all of the `Book` class attributes in the default manner except for the `author` attribute. It should read the relevant strings (first name, last name) and `LocalDate` (date of birth) from the `ObjectInputStream` passed to the `readObject()` method, instantiate an `Author` object, and set it to the `author` class attribute. You can assume an `Author` constructor is available that takes three parameters, one for each of the data read from the stream.

8 MARKS

iii. Write a Java program that creates a list of three `Book` objects. The program should then write the list of `Book` objects to a file using object serialisation. The name of the file should be passed in at the command line to the `main()` method. 5 MARKS

iv. Write another Java program to de-serialise a list of `Book` objects from a file created using serialisation. The name of the file should be passed in at the command line to the `main()` method.

5 MARKS

Q2:

(a) Describe the relationship between a nested inner class and its enclosing class. What is a static nested class and how does it compare to a nested inner class?

5 MARKS

(b) Write a class called `Bank` that maintains a data structure of bank customers of type `Customer`, and their `Account` object. Call this data structure `customers`. You can assume the existence of a `Customer` class. The Bank class should include a nested inner class called `Account`, which has an account number (an `int`) and an account balance (a `double`). Include an `addCustomer` method in the `Bank` class that takes a `Customer` object as a parameter, creates an account for them, and stores the customer and their account in the data structure. The `Bank` class has the responsibility for generating unique account numbers for accounts.

Briefly describe and provide a justification for the data structure you chose for `customers`.

12 MARKS

(c) Assuming the Customer object consists of a First Name (`String`), Last Name (`String`), and a PPS number (`int`), write an appropriate `hashCode()` method for the class based on these three values.

8 MARKS

Q3:

(a) Describe the `Set` interface in the Java Collections Framework. What is its relationship to the `Collection` interface? What are the main characteristics of a `Set` collection? List and briefly describe two classes in the Java Collections Framework that implement the `Set` interface, outlining the difference between them.

6 MARKS

(b) Explain fully the purpose and operation of the following code idiom:

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

6 MARKS

(c) Write a `Plant` class that includes an ID number, a `genus` name and a `species` name as class attributes. The `Plant` class should implement the

`Comparable` interface to define the natural order for these objects such that the genus is compared first and then the species.

Write a Java program that uses an `ArrayList` to store a collection of Plant objects and then sort the list based on natural order.

Also, write the code for a `Comparator` class i.e., a class that implements the `Comparator` interface, that can be used to compare two `Plant` objects based only on their ID number.

Finally, use the version of the `Collections.sort()` method that allows you to pass your own Comparator object to re-sort the list of `Plant` objects.

13 MARKS

Q4:

(a) The original Lemmings video game is a puzzle game where a player can assign skills to Lemming characters in order to help them navigate through a level. Implement an Enum in Java called `Lemming` which enumerates the different Lemming skill types from the original Lemmings video game. Include in the enum the speed of each lemming carrying out their respective skill, as follows, Climber=5, Floater=3, Bomber=0, Blocker=0, Builder=2, Basher=4, Miner=3, Digger=1, and a `boolean` indicating whether or not they are destructive to their environment. Bombers, Bashers, Miners, and Diggers are destructive; the others are not. Provide a suitable `toString()` method to print information about the enumerated types.

12 MARKS

(b) Write a Java program to provide a GUI for configuring and starting the game of Lemmings. Use suitable Swing components to allow the player to set the following properties and perform the following operations:

1) Start the game.
2) Choose the game resolution from a list (640 x 480; 1280 x 720; 1920 x 1080; 2560 x 1440)
3) Enter a level code (e.g., CAJJMDLJCL)
4) Game difficulty selection buttons (Fun; Tricky; Taxing; Mayhem)
5) Exit the game.

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 have chosen above, write the code to construct the component, add the component to a container, and then set up simple event handling for the component. The event handlers only need to print out a message to indicate that they have been called.

13 MARKS

Q5:

(a) Show using a code example how a thread may be created (and started) using an application class that implements the `Runnable` interface.

Include a mechanism in the `Runnable` class to allow it to be shutdown gracefully (i.e., without needing to call the `stop()` method).

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

10 MARKS

(b) Outline the design and code implementation of the Java class for an object that will be used as a buffer to hold a `String` object. The contents of the `String` may be written 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 exactly once and there may be multiple producer and consumer threads executing (and attempting to access the buffer) concurrently.

15 MARKS

END

NUI Galway
OÉ Gaillimh

Semester 1 Examinations 2019 / 2020

Exam Code(s)	3BCT, 3BP, 3BLE
Exam(s)	Third Year Computer Science & Information Technology Third Year Electronic and Computer Engineering Third Year Electrical and Electronic Engineering
Module Code(s)	CT326
Module(s)	Programming III
Paper No.	1
External Examiner(s)	Dr. J. Howe
Internal Examiner(s)	Prof. M. Madden *Dr. D. Chambers
Instructions:	Answer any 4 questions. All questions carry equal marks.
Duration	2 hrs
No. of Pages	4
Discipline(s)	Computer Science
Course Co-Ordinator	Dr. D. Chambers

Requirements:

Release in Exam Venue	Yes	X	No	
MCQ Answersheet	Yes		No	X
Handout	None			
Statistical/ Log Tables	None			
Cambridge Tables	None			
Graph Paper	None			
Log Graph Paper	None			
Other Materials	None			
Graphic material in colour	Yes		No	X

PTO

1.a: Implement an Enum in Java which enumerates the months of the year. Include in the enum the relevant position of the month in the calender i.e. Jan = 1, Feb = 2, etc and the number of days in the month i.e. Jan = 31, Feb = 28, etc. You do not need to provide support for leap years in the implementation. Provide a suitable toString() method to print information about the enumerated types.
12 MARKS

b: The JDK contains two general-purpose List implementations i.e. *ArrayList* and *LinkedList*. Why is *ArrayList* generally the best performing implementation? Describe the circumstances under which *LinkedList* might offer better performance. Describe three of the polymorphic algorithms provided in the Java Collections framework. In relation to these algorithms, explain fully the purpose and operation of the following code idiom:

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

13 MARKS

2.a: Describe the general structure and purpose of the IO Streams classes provided in the Java programming environment. Also briefly describe the mechanism to support random file access in Java? 5 MARKS

b: Write a Java application that inputs a date as a string in the form 17-07-2010. 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. 5 MARKS

c: Develop a simple GUI-based Java program that may be used to control a lighting system. Use suitable Swing components to allow the lights controller to perform the following functions:

- Switch the light system on or off.
- Choose a light intensity level from a list - Level 1, 2, 3 or 4.
- Include Mode of Operation buttons - Manual or Timed.
- A button that, when pressed, will display the current status (i.e. the currently selected settings) of the lights in a text field.

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: 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 String object 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 only needs to include source code for the server side application. 12 MARKS

b: Write another Java application with the same functionality as outlined above, in part a of this question, but this time using Datagram type sockets. Hint: you can use ByteArrayOutputStream and ByteArrayInputStream to populate and read the array associated with the DatagramPacket object. This application does not need to implement a reliable data transfer protocol. The answer only needs to include source code for the server side application. 13 MARKS

4. Develop a simple Java based payroll system that can calculate the weekly pay due for different categories of employees. The system should be implemented using the following design guidelines:

a: Implement an *abstract* base class called Employee that is used to hold basic information about an employee e.g. name, address, etc. This class should also define an *abstract* method called earnings() that returns the weekly pay for each type of employee. The class should include a suitable constructor and accessor methods to retrieve information about the employee. Include the private instance variable joinDate in class Employee to represent when they joined the company. Use the java.util.Date class for this variable. 7 MARKS

b: Implement a class called Manager, derived from Employee. A manager is paid a fixed weekly salary. The class should include a suitable constructor and should also implement the earnings() method. 5 MARKS

c: Implement a class called HourlyWorker, derived from Employee. An hourly worker is paid a fixed wage per hour, so in any given week they will be paid for the number of hours worked in the past week. The class should include a constructor and implement the earnings() method. 5 MARKS

d: Write a short driver program that creates an array of Employee variables to store references to the various employee objects. In a loop, calculate the payroll for each Employee, and add a €100.00 bonus to the person's payroll if they joined the company over 5 years ago. 8 MARKS

5.a: Discuss briefly the differences between a process and a thread. What is the best way to stop executing threads, assuming they still haven't finished all their work? 5 MARKS

b: Write a Java animation applet that uses a thread to continuously scroll a message string across the screen from right to left. 8 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. 12 MARKS

NUI Galway
OÉ Gaillimh

Autumn Examinations 2019

Exam Code(s)	3BCT, 3BP, 3BLE
Exam(s)	Third Year Computer Science & Information Technology Third Year Electronic and Computer Engineering Third Year Electrical and Electronic Engineering
Module Code(s)	CT326
Module(s)	Programming III
Paper No.	1
External Examiner(s)	Dr. J. Howe
Internal Examiner(s)	Prof. M. Madden *Dr. D. Chambers
Instructions:	Answer any 4 questions. All questions carry equal marks.
Duration	2 hrs
No. of Pages	4
Discipline(s)	Information Technology
Course Co-Ordinator	Dr. D. Chambers
Requirements	None

1. Develop a simple Java based payroll system that can calculate the weekly pay due for different categories of employees. The system should be implemented using the following design guidelines:

   a: Implement an *abstract* base class called Employee that is used to hold and access basic information about an employee e.g. name, address, etc. This class should also define an *abstract* method called earnings() that returns the weekly pay for each type of employee. The class should include a suitable constructor and accessor methods to retrieve information about the employee. 7 MARKS

   b: Implement a class called Manager, derived from Employee. A manager is paid a fixed weekly salary. The class should include a suitable constructor and should also implement the earnings() method. 6 MARKS

   c: Implement a class called HourlyWorker, derived from Employee. An hourly worker is paid a fixed wage per hour, so in any given week they will be paid for the number of hours worked in the past week. The class should include a constructor and implement the earnings() method. 6 MARKS

   d: Write a short driver program that creates an object for each of the employee sub-classes, it then calls the earnings() method for each object and displays the results. 6 MARKS

2.a: Implement an enum in Java which enumerates the days of the week. Assuming the week starts on Sunday, include in the enum the relevant position of the day in the week i.e. Sun = 1, Mon = 2, etc and whether the day is a normally a working day i.e. Sun = false, Mon = true, etc. Provide a suitable toString() method to print information about the enumerated types. 12 MARKS

   b: Write a simple Student class that includes an ID number, a first name and a last name as class attributes. The Student class should implement the Comparable interface to define the natural order for these objects such that the last name is compared first and then the first name. Write a Java program that uses an ArrayList to store a collection of Student objects and then sort the list based on natural order. Also, write the code for a Comparator class i.e. a class that implements the Comparator interface, that can be used to compare two Student objects based only on their ID number. Finally, use the version of the Collections.sort() method that allows you to pass your own Comparator object to re-sort the list of Employee objects. 13 MARKS

3. Write a Java class called **Rational** for performing arithmetic with fractions.

Use integer variables to represent the **private** instance variables of the class - the **numerator** and the **denominator**. Provide a constructor method that enables an object of this class to be initialised when it is declared e.g. the fraction 2/3 would be stored in the object as 2 in the **numerator** and 3 in the **denominator**.
4 MARKS

Provide **public** methods for each of the following:

(a) Addition of two **Rational** numbers. The result should be stored in the target object e.g. if **r1** and **r2** are objects of type **Rational**, calling **r1.add(r2)** would add the value of **r2** to **r1** and then store the new value in **r1**. (Hint: adding 2/3 and 3/4 gives the result 17/12). 7 MARKS

(b) Multiplication of two **Rational** numbers. In the same way as for (a), the result should be stored in the target object. (Hint: multiplying 2/3 and 3/4 gives the result 6/12). 7 MARKS

(c) Printing **Rational** numbers in the form **a/b**, where **a** is the **numerator** and **b** is the **denominator**. 3 MARKS

Finally, write a suitable driver program that could be used to test your class.
4 MARKS

4.a: 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 String object 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 only needs to include source code for the server side application.
12 MARKS

b: Write another Java application with the same functionality as outlined above, in part a of this question, but this time using Datagram type sockets. Hint: you can use ByteArrayOutputStream and ByteArrayInputStream to populate and read the array associated with the DatagramPacket object. This application does not need to implement a reliable data transfer protocol. The answer only needs to include source code for the server side application. 13 MARKS

5.a: Evaluate the following code sample (in terms of good design practice). The code is for a simple home heating system. The system is turned on if the current temperature falls below some minimum value. It's then turned off again when it goes above the maximum value. The class is instantiated and started i.e. it runs in its own thread of execution.

```
class Thermostat extends Thread
{
        private final int THERMOMETER = 0x10;
        private final int HEATER = 0xf7;
        private final int HEATER_ON = 0x1;
        private final int HEATER_OFF = 0;
        public void run()
        {
            while (true)
            {
                while (read(THERMOMETER) > min)
                    sleep(100);
                write(HEATER, HEATER_ON);
                while (read(THERMOMETER) < max)
                    sleep(100);
                write(HEATER, HEATER_OFF);
            }
        }
}
```

What is wrong with this code? Suggest a better design approach based on using the dependency inversion principle. 10 MARKS

b: 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.
15 MARKS

NUI Galway
OÉ Gaillimh

Semester 1 Examinations 2018 / 2019

Exam Code(s)	3BCT, 3BP, 3BLE
Exam(s)	Third Year Computer Science & Information Technology Third Year Electronic and Computer Engineering Third Year Electrical and Electronic Engineering
Module Code(s)	CT326
Module(s)	Programming III
Paper No.	1
External Examiner(s)	Dr. J. Howe
Internal Examiner(s)	Prof. M. Madden *Dr. D. Chambers
Instructions:	Answer any 4 questions. All questions carry equal marks.
Duration	2 hrs
No. of Pages	4
Department(s)	Information Technology
Requirements	None

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

```
import java.io.*;
public class Account {
        protected HolderDetails holder;
        protected List<Transaction> transactions;
        protected float balance;

        // Add a suitable constructor here

        // Add methods to make deposits / withdrawals

        // 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 / withdrawals and a method to print out a transaction summary for a given time interval. Include appropriate user defined Exceptions as required.
8 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.
8 MARKS

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

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

2.a: Describe the general structure and purpose of the IO Streams classes provided in the Java programming environment. Also describe the mechanism that supports random file access in Java. 5 MARKS

b: Write a Java application that inputs a date as a string in the form 17-07-2018 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 Student class that includes an id number, a name, and course details and a suitable constructor method. Then write a Java program that uses an ArrayList to store a collection of Student 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 Student 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 Student objects. 14 MARKS

3.a: What types of Sockets are supported in the Java networking package? Which type of Socket would you recommend for a VOIP type application and a File Transfer type application? 3 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 String object 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 only needs to include source code for the server side application. 10 MARKS

c: Write another Java application with the same functionality as outlined above, in part b of this question, but this time using Datagram type sockets. Hint: you can use ByteArrayOutputStream and ByteArrayInputStream to populate and read the array associated with the DatagramPacket object. This application does not need to implement a reliable data transfer protocol. The answer only needs to include source code for the server side application. 12 MARKS

4.a: What is the best way to stop executing threads (assuming they still have not finished their work)? Show using a code example how a thread may be created (and started) using an application class that 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

b: 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 exactly once and there may be multiple producer and consumer threads executing (and attempting to access the buffer) concurrently.
15 MARKS

5: Suppose that you've written a program that displays two messages, as follows:

```
public class NotI18N {
    static public void main(String[] args) {
        System.out.println("Hello.");
        System.out.println("How are you?");
     }
}
```

You then decide that this program needs to display the same or similar messages for people living in France and Spain. Outline the steps needed to properly internationalise this program i.e. the hardcoded English language messages should be removed and replaced with a more flexible mechanism that will facilitate additional language support in the future. 10 MARKS

b: Write 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 have chosen above, write the code to construct the component, add the component to a container and then set up 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

NUI Galway
OÉ Gaillimh

Autumn Examinations 2018

Exam Code(s)	3BCT, 3BP, 3BLE
Exam(s)	Third Year Computer Science & Information Technology Third Year Electronic and Computer Engineering Third Year Electrical and Electronic Engineering
Module Code(s)	CT326
Module(s)	Programming III
Paper No.	1
External Examiner(s)	Dr. J. Howe
Internal Examiner(s)	Prof. M. Madden *Dr. D. Chambers
Instructions:	Answer any 4 questions. All questions carry equal marks.
Duration	2 hrs
No. of Pages	4
Department(s)	Information Technology
Requirements	None

1. Develop a simple Java based payroll system that can calculate the weekly pay due for different categories of employees. The system should be implemented using the following design guidelines:

   a: Implement an *abstract* base class called Employee that is used to hold and access basic information about an employee e.g. name, address, etc. This class should also define an *abstract* method called earnings() that returns the weekly pay for each type of employee. The class should include a suitable constructor and accessor methods to retrieve information about the employee. 7 MARKS

   b: Implement a class called Manager, derived from Employee. A manager is paid a fixed weekly salary. The class should include a suitable constructor and should also implement the earnings() method. 6 MARKS

   c: Implement a class called HourlyWorker, derived from Employee. An hourly worker is paid a fixed wage per hour, so in any given week they will be paid for the number of hours worked in the past week. The class should include a constructor and implement the earnings() method. 6 MARKS

   d: Write a short driver program that creates an object for each of the employee sub-classes, it then calls the earnings() method for each object and displays the results. 6 MARKS

2.a: Implement an enum in Java which enumerates the days of the week. Assuming the week starts on Sunday, include in the enum the relevant position of the day in the week i.e. Sun = 1, Mon = 2, etc and whether the day is a normally a working day i.e. Sun = false, Mon = true, etc. Provide a suitable toString() method to print information about the enumerated types. 12 MARKS

   b: 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. 13 MARKS

3. Write a Java class called **Rational** for performing arithmetic with fractions.

Use integer variables to represent the **private** instance variables of the class - the **numerator** and the **denominator**. Provide a constructor method that enables an object of this class to be initialised when it is declared e.g. the fraction 2/3 would be stored in the object as 2 in the **numerator** and 3 in the **denominator**.
4 MARKS

Provide **public** methods for each of the following:

(a) Addition of two **Rational** numbers. The result should be stored in the target object e.g. if **r1** and **r2** are objects of type **Rational**, calling **r1.add(r2)** would add the value of **r2** to **r1** and then store the new value in **r1**. (Hint: adding 2/3 and 3/4 gives the result 17/12). 7 MARKS

(b) Multiplication of two **Rational** numbers. In the same way as for (a), the result should be stored in the target object. (Hint: multiplying 2/3 and 3/4 gives the result 6/12). 7 MARKS

(c) Printing **Rational** numbers in the form **a/b**, where **a** is the **numerator** and **b** is the **denominator**. 3 MARKS

Finally, write a suitable driver program that could be used to test your class.
4 MARKS

4.a: 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 String object 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 only needs to include source code for the server side application.
12 MARKS

b: Write another Java application with the same functionality as outlined above, in part a of this question, but this time using Datagram type sockets. Hint: you can use ByteArrayOutputStream and ByteArrayInputStream to populate and read the array associated with the DatagramPacket object. This application does not need to implement a reliable data transfer protocol. The answer only needs to include source code for the server side application. 13 MARKS

5.a: Evaluate the following code sample (in terms of good design practice). The code is for a simple home heating system. The system is turned on if the current temperature falls below some minimum vlaue. It's then turned off again when it goes above the maximum value. The class is instantiated and started i.e. it runs in its own thread of execution.

```
class Thermostat extends Thread
{
        private final int THERMOMETER = 0x10;
        private final int HEATER = 0xf7;
        private final int HEATER_ON = 0x1;
        private final int HEATER_OFF = 0;
        public void run()
        {
            while (true)
            {
                while (read(THERMOMETER) > min)
                        sleep(100);
                write(HEATER, HEATER_ON);
                while (read(THERMOMETER) < max)
                        sleep(100);
                write(HEATER, HEATER_OFF);
            }
        }
}
```

What is wrong with this code? Suggest a better design approach based on using the dependency inversion principle. 10 MARKS

b: 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.
15 MARKS

NUI Galway
OÉ Gaillimh

Semester 1 Examinations 2017 / 2018

Exam Code(s)	3BCT, 3BP, 3BLE
Exam(s)	Third Year Computer Science & Information Technology Third Year Electronic and Computer Engineering Third Year Electrical and Electronic Engineering
Module Code(s)	CT326
Module(s)	Programming III
Paper No.	1
External Examiner(s)	Dr. J. Howe
Internal Examiner(s)	Dr. M. Schukat *Dr. D. Chambers
Instructions:	Answer any 4 questions. All questions carry equal marks.
Duration	2 hrs
No. of Pages	4
Department(s)	Information Technology
Requirements	None

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

```
import java.io.*;
public class Account implements Comparable<Account>,
Serializable {
     protected int accnum;
     protected HolderDetails holder;
     protected List<Transaction>  transactions;
     protected float balance;

     // Add a suitable constructor here

     // Add methods to make deposits / withdrawals

     // Method to print out account transaction summary

     // Add suitable attribute accessor methods

     // Add method to implement the Comparable interface
}
```

a: Complete the implementation of the Account class, providing a suitable constructor, attribute accessor methods, methods for making a deposit or withdrawal, a method to print out a transaction summary related to a range of dates and an implementation method for the Comparable interface.
7 MARKS

b: Provide implementations for the HolderDetails class and the Transaction class. The HolderDetails class is used to store personal details about the account holder. The Transaction class contains details about past transactions including the type of transaction, the amount and the Date. 7 MARKS

c: Define and implement a new class, called CurrentAccount, derived from Account, that allows withdrawals to proceed up to some overdraft limit. Note that the base Account class shown has no overdraft facility.
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: Implement an enum in Java which enumerates the days of the week. Assuming the week starts on Sunday, include in the enum the relevant position of the day in the week i.e. Sun = 1, Mon = 2, etc and whether the day is a normally a working day i.e. Sun = false, Mon = true, etc. Provide a suitable toString() method to print information about the enumerated types. 12 MARKS

b: 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. 13 MARKS

3.a: 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 String object 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 only needs to include source code for the server side application. 12 MARKS

b: Write another Java application with the same functionality as outlined above, in part a of this question, but this time using Datagram type sockets. Hint: you can use ByteArrayOutputStream and ByteArrayInputStream to populate and read the array associated with the DatagramPacket object. This application does not need to implement a reliable data transfer protocol. The answer only needs to include source code for the server side application. 13 MARKS

4.a: Discuss briefly the differences between a process and a thread. What is the best way to stop executing threads, assuming they still have not finished their work?
5 MARKS

b: Write a Java animation applet that uses a thread to continuously scroll a text message across the screen from right to left. The message itself and the rate at which the text scrolls can be passed to the applet as HTML based parameters.
10 MARKS

c: Outline the design and give the full source code for a Java class that will be used as a thread safe 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 exactly once and there may be multiple producer and consumer threads executing (and attempting to access the buffer) concurrently.
10 MARKS

5: Assume that a Sports Club at the University 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 registration status i.e. are they fully paid up members of the club. Each member of the club should also have a unique membership id number, this number is automatically assigned when the member object is created. The Member class should implement the Comparable interface and use the membership id number to define the natural order for these objects.
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. Member objects should be sorted by their id number as they are added to the collection. SportsClub should include methods for adding new members, removing members and returning a list of current members. 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. The program should also be able to save the full application state on exit, using Object Serialization, and load up the saved state at startup.
5 MARKS

NUI Galway
OÉ Gaillimh

Autumn Examinations 2017

Exam Code(s)	3BCT, 3BP, 3BLE
Exam(s)	Third Year Computer Science & Information Technology Third Year Electronic and Computer Engineering Third Year Electrical and Electronic Engineering
Module Code(s)	CT326
Module(s)	Programming III
Paper No.	1
External Examiner(s)	Dr. J. Power
Internal Examiner(s)	Dr. M. Schukat *Dr. D. Chambers
Instructions:	Answer any 4 questions. All questions carry equal marks.
Duration	2 hrs
No. of Pages	4
Department(s)	Information Technology
Requirements	None

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

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

        // Add a suitable constructor here

        // Add methods to make deposits / withdrawals

        // 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 / withdrawals and a method to print out a transaction summary for a given time interval. 8 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. 8 MARKS

c: Define and implement a new class, called CurrentAccount, derived from Account, that allows withdrawals to proceed up to some overdraft limit. 6 MARKS

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

2.a: Create a standalone Java application that will count and sum up the number of lines in the text file passed as an argument on the command line. The program should create a **FileReader** object and pass this in the constructor of a **LineNumberReader** object to handle the file reading required.

The **LineNumberReader** class has two useful methods (that could be used):

public String readLine() throws IOException; This method reads a line of text. It returns a String containing the contents of the line, not including any line-termination characters, or null if the end of the stream has been reached.

public int getLineNumber(); This method returns the current line number.

10 MARKS

b: 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: What types of Sockets are supported in the Java networking package? Which type of Socket would you recommend for a VOIP type application? Would the same type of socket also be the most suitable for a File Transfer type application?

5 MARKS

b: Write a Java application that uses Stream type sockets to exchange Java Objects using object serialisation. The client side should connect to the server and send it a String object. The server should print out the String and respond to the client with a text based response encapsulated in another String Object. The client should receive the String Object from the server and print out this response.

10 MARKS

c: Write another Java application with the same functionality as outlined above, in part a of this question, but this time using Datagram type sockets. Hint: you can use ByteArrayOutputStream and ByteArrayInputStream to populate and read the array associated with the DatagramPacket object. This application does not need to implement a reliable data transfer protocol. The answer only needs to include source code for the server side application. 10 MARKS

4.a: Discuss briefly the differences between a process and a thread. What is the best way to stop executing threads (assuming they still haven't finished their work)?
5 MARKS

b: Write a JAVA animation applet that uses a thread to continuously scroll a text message across the screen from right to left. The message itself and the rate at which the text scrolls can be passed to the applet as HTML based parameters.
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 exactly once and there may be multiple producer and consumer threads executing (and attempting to access the buffer) concurrently.
10 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 constructor method(s) that enable objects of this class to be fully initialized.
5 MARKS

b: Provide a **public** method to add two **Complex** numbers: the real part of the right operand is added to the real part of the left operand, the imaginary part of the right operand is added to the imaginary part of the left operand. 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. 7 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 7 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 and write a short driver program to test your class. 6 MARKS

NUI Galway
OÉ Gaillimh

Semester 1 Examinations 2016 / 2017

Exam Code(s)	3BCT, 3BP, 3BLE
Exam(s)	Third Year Computer Science & Information Technology Third Year Electronic and Computer Engineering Third Year Electrical and Electronic Engineering
Module Code(s)	CT326
Module(s)	Programming III
Paper No.	1
External Examiner(s)	Dr. J. Power
Internal Examiner(s)	Dr. J. Duggan *Dr. D. Chambers
Instructions:	Answer any 4 questions. All questions carry equal marks.
Duration	2 hrs
No. of Pages	4
Department(s)	Information Technology
Requirements	None

1. Develop a simple Java based payroll system that can calculate the weekly pay due for different categories of employees. The system should be implemented using the following design guidelines:

   a: Implement an *abstract* base class called Employee that is used to hold basic information about an employee e.g. name, address, etc. This class should also define an *abstract* method called earnings() that returns the weekly pay for each type of employee. The class should include a suitable constructor and accessor methods to retrieve information about the employee. Include the private instance variable joinDate in class Employee to represent when they joined the company. Use the java.util.Date class for this variable. 7 MARKS

   b: Implement a class called Manager, derived from Employee. A manager is paid a fixed weekly salary. The class should include a suitable constructor and should also implement the earnings() method. 5 MARKS

   c: Implement a class called HourlyWorker, derived from Employee. An hourly worker is paid a fixed wage per hour, so in any given week they will be paid for the number of hours worked in the past week. The class should include a constructor and implement the earnings() method. 5 MARKS

   d: Write a short driver program that creates an array of Employee variables to store references to the various employee objects. In a loop, calculate the payroll for each Employee, and add a €100.00 bonus to the person's payroll if they joined the company over 5 years ago. 8 MARKS

2.a: Implement an Enum in Java which enumerates the months of the year. Include in the enum the relevant position of the month in the calender i.e. Jan = 1, Feb = 2, etc and the number of days in the month i.e. Jan = 31, Feb = 28, etc. You do not need to provide support for leap years in the implementation. Provide a suitable toString() method to print information about the enumerated types. 12 MARKS

   b: The JDK contains two general-purpose List implementations i.e. *ArrayList* and *LinkedList*. Why is *ArrayList* generally the best performing implementation? Describe the circumstances under which *LinkedList* might offer better performance. Describe the polymorphic algorithms provided in the JAVA Collections framework. In relation to these algorithms, explain fully the purpose and operaton of the following code idiom:

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

13 MARKS

3.a: Write a Java class called Rational for performing arithmetic with positive rational numbers. A positive rational no. is a number that can be expressed in the form of p/q, where p,q are positive integers with q !=0 and p and q have no common divisor. Make the class implement the comparable interface so that each object should be able to compare itself to another object of the same type based on ascending order. 5 MARKS

b: Write a Java program that uses an ArrayList to store a collection of 10 of the Rational objects defined in part a. Sort the ArrayList of Rational objects according to the Natural Order defined in the Comparable interface of the Rational class itself using the sort method provided in the collections framework and then finally print out the list. 8 MARKS

c: Sort the ArrayList of Rational objects again using a separate Comparator class (a class that implements java.util.Comparator) that sorts them in descending order i.e. the opposite of the Natural Order defined in the Rational class itself. 7 MARKS

d: Modify the program to use the built-in Collections.binarySearch() method to check if a particular fraction, inputted through the console, is contained within the list. Print out your results to the console. 5 MARKS

4.a: 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 String object 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 only needs to include source code for the server side application. 12 MARKS

b: Write another Java application with the same functionality as outlined above, in part a of this question, but this time using Datagram type sockets. Hint: you can use ByteArrayOutputStream and ByteArrayInputStream to populate and read the array associated with the DatagramPacket object. This application does not need to implement a reliable data transfer protocol. The answer only needs to include source code for the server side application. 13 MARKS

5.a: Evaluate the following code sample (in terms of good design practice). The code is for a simple home heating system. The system is turned on if the current temperature falls below some minimum value. It's then turned off again when it goes above the maximum value. The class is instantiated and started i.e. it runs in its own thread of execution.

```
class Thermostat extends Thread
{
        private final int THERMOMETER = 0x10;
        private final int HEATER = 0xf7;
        private final int HEATER_ON = 0x1;
        private final int HEATER_OFF = 0;
        public void run()
        {
            while (true)
            {
                while (read(THERMOMETER) > min)
                        sleep(100);
                write(HEATER, HEATER_ON);
                while (read(THERMOMETER) < max)
                        sleep(100);
                write(HEATER, HEATER_OFF);
            }
        }
}
```

What is wrong with this code? Suggest a better design approach based on using the dependency inversion principle. 10 MARKS

b: 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 exactly once and there may be multiple producer and consumer threads executing (and attempting to access the buffer) concurrently.
15 MARKS

NUI Galway
OÉ Gaillimh

Autumn Examinations 2016

Exam Code(s)	3BCT, 3BP, 3BLE
Exam(s)	Third Year Computer Science & Information Technology Third Year Electronic and Computer Engineering Third Year Electrical and Electronic Engineering
Module Code(s)	CT326
Module(s)	Programming III
Paper No.	1
External Examiner(s)	Dr. J. Power
Internal Examiner(s)	Prof. G. Lyons Dr. M. Madden *Dr. D. Chambers
Instructions:	Answer any 4 questions. All questions carry equal marks.
Duration	2 hrs
No. of Pages	4
Department(s)	Information Technology
Requirements	None

1. Develop a simple Java based payroll system that can calculate the weekly pay due for different categories of employees. The system should be implemented using the following design guidelines:

   a: Implement an *abstract* base class called Employee that is used to hold and access basic information about an employee e.g. name, address, etc. This class should also define an *abstract* method called earnings() that returns the weekly pay for each type of employee. The class should include a suitable constructor and accessor methods to retrieve information about the employee. 7 MARKS

   b: Implement a class called Manager, derived from Employee. A manager is paid a fixed weekly salary. The class should include a suitable constructor and should also implement the earnings() method. 6 MARKS

   c: Implement a class called HourlyWorker, derived from Employee. An hourly worker is paid a fixed wage per hour, so in any given week they will be paid for the number of hours worked in the past week. The class should include a constructor and implement the earnings() method. 6 MARKS

   d: Write a short driver program that creates an object for each of the employee sub-classes, it then calls the earnings() method for each object and displays the results. 6 MARKS

2.a: Create a standalone Java application that will count and sum up the number of lines in the text file passed as an argument on the command line. The program should create a **FileReader** object and pass this in the constructor of a **LineNumberReader** object to handle the file reading required.

   The **LineNumberReader** class has two useful methods (that could be used):

   public String readLine() throws IOException; This method reads a line of text. It returns a String containing the contents of the line, not including any line-termination characters, or null if the end of the stream has been reached.

   public int getLineNumber(); This method returns the current line number. 10 MARKS

   b: 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. Write a Java class called **Rational** for performing arithmetic with fractions.

Use integer variables to represent the **private** instance variables of the class - the **numerator** and the **denominator**. Provide a constructor method that enables an object of this class to be initialised when it is declared e.g. the fraction 2/3 would be stored in the object as 2 in the **numerator** and 3 in the **denominator**.
4 MARKS

Provide **public** methods for each of the following:

(a) Addition of two **Rational** numbers. The result should be stored in the target object e.g. if **r1** and **r2** are objects of type **Rational**, calling **r1.add(r2)** would add the value of **r2** to **r1** and then store the new value in **r1**. (Hint: adding 2/3 and 3/4 gives the result 17/12). 7 MARKS

(b) Multiplication of two **Rational** numbers. In the same way as for (a), the result should be stored in the target object. (Hint: multiplying 2/3 and 3/4 gives the result 6/12). 7 MARKS

(c) Printing **Rational** numbers in the form **a/b**, where **a** is the **numerator** and **b** is the **denominator**. 3 MARKS

Finally, write a suitable driver program that could be used to test your class.
4 MARKS

4.a: 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 String object 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 only needs to include source code for the server side application.
12 MARKS

b: Write another Java application with the same functionality as outlined above, in part a of this question, but this time using Datagram type sockets. Hint: you can use ByteArrayOutputStream and ByteArrayInputStream to populate and read the array associated with the DatagramPacket object. This application does not need to implement a reliable data transfer protocol. The answer only needs to include source code for the server side application. 13 MARKS

5.a: Evaluate the following code sample (in terms of good design practice). The code is for a simple home heating system. The system is turned on if the current temperature falls below some minimum vlaue. It's then turned off again when it goes above the maximum value. The class is instantiated and started i.e. it runs in its own thread of execution.

```
class Thermostat extends Thread
{
        private final int THERMOMETER = 0x10;
        private final int HEATER = 0xf7;
        private final int HEATER_ON = 0x1;
        private final int HEATER_OFF = 0;
        public void run()
        {
            while (true)
            {
                while (read(THERMOMETER) > min)
                        sleep(100);
                write(HEATER, HEATER_ON);
                while (read(THERMOMETER) < max)
                        sleep(100);
                write(HEATER, HEATER_OFF);
            }
        }
}
```

What is wrong with this code? Suggest a better design approach based on using the dependency inversion principle. 10 MARKS

b: 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.
15 MARKS

Semester 1 Examinations 2015 / 2016

Exam Code(s)	3BCT, 3BP, 3BLE
Exam(s)	Third Year Computer Science & Information Technology Third Year Electronic and Computer Engineering Third Year Electrical and Electronic Engineering
Module Code(s)	CT326
Module(s)	Programming III
Paper No.	1
External Examiner(s)	Dr. J. Power
Internal Examiner(s)	Prof. G. Lyons Dr. M. Madden *Dr. D. Chambers
Instructions:	Answer any 4 questions. All questions carry equal marks.
Duration	2 hrs
No. of Pages	4
Department(s)	Information Technology
Requirements	None

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

```
import java.io.*;
public class Account implements Comparable<Account>,
Serializable {
     protected int accnum;
     protected HolderDetails holder;
     protected List<Transaction> transactions;
     protected float balance;

     // Add a suitable constructor here

     // Add methods to make deposits / withdrawals

     // Method to print out account transaction summary

     // Add suitable attribute accessor methods

     // Add method to implement the Comparable interface
}
```

a: Complete the implementation of the Account class, providing a suitable constructor, attribute accessor methods, methods for making a deposit or withdrawal, a method to print out a transaction summary related to a range of Dates. Also provide an implementation method for the Comparable interface that bases the natural order of these objects on their accnum attribute.
7 MARKS

b: Provide implementations for the HolderDetails class and the Transaction class. The HolderDetails class is used to store personal details about the account holder. The Transaction class contains details about past transactions including the type of transaction, the amount and the Date. 7 MARKS

c: Define and implement a new class, called CurrentAccount, derived from Account, that allows withdrawals to proceed up to some overdraft limit. Note that the base Account class shown has no overdraft facility.
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: Write the Java code for a Vehicle class that implements an accessor method called getEngineSize() that returns an int representing the engine size for that vehicle e.g. 1895. Then write a Java program that uses an ArrayList to store a collection of Vehicle 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 Vehicle objects based on their engine size. Finally, use the version of the Collections.sort() method that allows you to pass your own Comparator object to sort the list of Vehicle objects. 12 MARKS

b: The JDK contains two general-purpose List implementations i.e. *ArrayList* and *LinkedList*. Which of these classes is generally considered to be the best performing implementation and why is this the case? Outline the circumstances under which each of these classes might offer better performance than the other. Describe the polymorphic algorithms provided in the JAVA Collections framework. In relation to these algorithms, explain fully the purpose and operaton of the following code idiom:

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

13 MARKS

3.a: 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 an Integer object to the server, the server then responds with a single String object. 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, the client side source code is not required. 13 MARKS

b: Write another Java application with the same functionality as outlined in part a of this question, but this time using Datagram type sockets. Hint: you can use ByteArrayOutputStream and ByteArrayInputStream to populate and read the array associated with the DatagramPacket object. This application does not need to implement a reliable data transfer protocol. In this case, only the source code for the client application is required, the server side source code is not required. 12 MARKS

4.a: Discuss briefly the differences between a process and a thread. What is the best way to stop executing threads (assuming they still haven't finished their work)? 5 MARKS

b: Show, using a simple code example, how a thread may be created (and started) using an application class that 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. 8 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 exactly once and there may be multiple producer and consumer threads executing (and attempting to access the buffer) concurrently. 12 MARKS

5.a: Suppose that you've written a program that displays two messages, as follows:

```
public class NotI18N {
    static public void main(String[] args) {
        System.out.println("Hello.");
        System.out.println("How are you?");
     }
}
```

You then decide that this program needs to display the same or similar messages for people living in France and Spain. Outline the steps needed to properly internationalise this program i.e. the hardcoded English language messages should be removed and replaced with a more flexible mechanism that will facilitate additional language support in the future. 10 MARKS

b: Write 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 1200 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 set up 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

NUI Galway
OÉ Gaillimh

Autumn Examinations 2015

Exam Code(s)	3BCT, 3BP1
Exam(s)	Third Year Computer Science & Information Technology Third Year Electronic and Computer Engineering
Module Code(s)	CT326
Module(s)	Programming III
Paper No.	1
External Examiner(s)	Dr. J. Power
Internal Examiner(s)	Prof. G. Lyons Dr. M. Madden *Dr. D. Chambers
Instructions:	Answer any 4 questions. All questions carry equal marks.
Duration	2 hrs
No. of Pages	4
Department(s)	Information Technology
Requirements	None

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

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

        // Add a suitable constructor here

        // Add methods to make deposits / withdrawals

        // 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 / withdrawals and a method to print out a transaction summary for a given time interval. 8 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. 8 MARKS

c: Define and implement a new class, called CurrentAccount, derived from Account, that allows withdrawls to proceed up to some overdraft limit. 6 MARKS

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

2.a: Create a standalone Java application that will count and sum up the number of lines in the text file passed as an argument on the command line. The program should create a **FileReader** object and pass this in the constructor of a **LineNumberReader** object to handle the file reading required.

The **LineNumberReader** class has two useful methods (that could be used):

public String readLine() throws IOException; This method reads a line of text. It returns a String containing the contents of the line, not including any line-termination characters, or null if the end of the stream has been reached.

public int getLineNumber(); This method returns the current line number.

10 MARKS

b: 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: What types of Sockets are supported in the Java networking package? Which type of Socket would you recommend for a VOIP type application and a File Transfer type application? 5 MARKS

b: Write a Java application that uses Stream type sockets to exchange Java Objects using object serialisation. The client side should connect to the server and send it a String object. The server should print out the String and respond to the client with a text based response encapsulated in another String Object. The client should receive the String Object from the server and print out this response. The answer only needs to include source code for the server side application.

10 MARKS

c: Write another Java application with the same functionality as outlined in part b of this question, but this time using Datagram type sockets. Hint: you can use ByteArrayOutputStream and ByteArrayInputStream to populate and read the array associated with the DatagramPacket object. The answer only needs to include source code for the server side application.

10 MARKS

4.a: Discuss briefly the differences between a process and a thread. What is the best way to stop executing threads (assuming they still haven't finished their work)? 5 MARKS

b: What is meant by the term *deadlock*? Using the example of the *Dining Philosophers Problem* (covered in class), discuss how deadlock might occur in this case and propose a solution to overcome the problem. 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 exactly once and there may be multiple producer and consumer threads executing (and attempting to access the buffer) concurrently. 10 MARKS

5. Write a Java class called **Rational** for performing arithmetic with fractions. Use integer variables to represent the **private** instance variables of the class - the **numerator** and the **denominator**. Provide a constructor method that enables an object of this class to be initialised when it is declared e.g. the fraction 2/3 would be stored in the object as 2 in the **numerator** and 3 in the **denominator**. 3 MARKS

Provide **public** methods for each of the following:

(a) Addition of two **Rational** numbers. The result should be stored in the target object e.g. if **r1** and **r2** are objects of type **Rational**, calling **r1.add(r2)** would add the value of **r2** to **r1** and then store the new value in **r1**. (Hint: adding 2/3 and 3/4 gives the result 17/12). 4 MARKS

(b) Subtraction of two **Rational** numbers. In the same way as for (a), the result should be stored in the target object. (Hint: subtracting 2/3 from 3/4 gives the result 1/12). 4 MARKS

(c) Multiplication of two **Rational** numbers. In the same way as for (a), the result should be stored in the target object. (Hint: multiplying 2/3 and 3/4 gives the result 6/12). 4 MARKS

(d) Division of two **Rational** numbers. In the same way as for (a), the result should be stored in the target object. (Hint: dividing 2/3 by 3/4 gives the result 8/9). 4 MARKS

(e) Printing **Rational** numbers in the form **a/b**, where **a** is the **numerator** and **b** is the **denominator**. 3 MARKS

Finally, write a suitable driver program that could be used to test your class. 3 MARKS

NUI Galway
OÉ Gaillimh

Spring Examinations 2014 / 2015

Exam Code(s)	3BCT, 3BP1
Exam(s)	Third Year Computer Science & Information Technology Third Year Electronic and Computer Engineering
Module Code(s)	CT326
Module(s)	Programming III
Paper No.	1
External Examiner(s)	Dr. J. Power
Internal Examiner(s)	Prof. G. Lyons Dr. M. Madden *Dr. D. Chambers
Instructions:	Answer any 4 questions. All questions carry equal marks.
Duration	2 hrs
No. of Pages	4
Department(s)	Information Technology
Requirements	None

1.a Write a Java program that uses an ArrayList to store a collection of Student 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 Student objects based on their course code. The Student class implements an accessor method called getCourseCode() that returns a String representing the course code for that student e.g. 3BCT. Finally, use the version of the Collections.sort() method that allows you to pass your own Comparator object to sort the list of Student objects. 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 1200 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 have chosen above, write the code to construct the component, add the component to a container and then set up 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

2.a: 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 string object 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. 12 MARKS

b: Write another Java application with the same functionality as outlined in part a of this question, but this time using Datagram type sockets. Hint: you can use ByteArrayOutputStream and ByteArrayInputStream to populate and read the array associated with the DatagramPacket object. This application does not need to implement a reliable data transfer protocol. 13 MARKS

3.a: What is the best way to stop executing threads (assuming they still have not finished their work)? Show using a code example how a thread may be created (and started) using an application class that 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

b: 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 written 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. 15 MARKS

4.a: 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

b: Suppose that you have written a program that displays two messages, as follows:

```
public class q4b {
   static public void main(String[] args) {
      System.out.println("Hello.");
      System.out.println("How are you?");
} }
```

You then decide that this program needs to display the same or similar messages for people living in France and Germany. Outline the steps needed to properly internationalise this program i.e. the hardcoded English language messages should be removed and replaced with a more flexible mechanism that will facilitate additional language support in the future. 7 MARKS

c: Using suitable source code samples, outline the steps required to use a Database within a Java program using the JDBC API. How are transactions supported using this mechamisn? What type of Metadata may be retrieved from a Database using JDBC? 12 MARKS

5: Assume that a Sports Club at the University 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 registration status i.e. are they fully paid up members of the club. Each member of the club should also have a unique membership id number, this number is automatically assigned when the member object is created. The Member class should implement the Comparable interface and use the membership id number to define the natural order for these objects.
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. Member objects should be sorted by their id number as they are added to the collection. 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 name or 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

NUI Galway
OÉ Gaillimh

Autumn Examinations 2014

Exam Code(s)	3BCT, 3BP1
Exam(s)	Third Year Computer Science & Information Technology Third Year Electronic and Computer Engineering
Module Code(s)	CT326
Module(s)	Programming III
Paper No.	1
External Examiner(s)	Dr. J. Power
Internal Examiner(s)	Prof. G. Lyons Dr. M. Madden *Dr. D. Chambers
Instructions:	Answer any 4 questions. All questions carry equal marks.
Duration	2 hrs
No. of Pages	5
Department(s)	Information Technology
Requirements	None

1. Develop a simple Java based payroll system that can calculate the weekly pay due for different categories of employees. The system should be implemented using the following design guidelines:

a: Implement an *abstract* base class called Employee that is used to hold and access basic information about an employee e.g. name, address, etc. This class should also define an *abstract* method called earnings() that returns the weekly pay for each type of employee. The class should include a suitable constructor and accessor methods to retrieve information about the employee. 5 MARKS

b: Implement a class called Manager, derived from Employee. A manager is paid a fixed weekly salary. The class should include a suitable constructor and should also implement the earnings() method. 5 MARKS

c: Implement a class called HourlyWorker, derived from Employee. An hourly worker is paid a fixed wage per hour, so in any given week they will be paind for the number of hours worked in the past week. The class should include a constructor and implement the earnings() method. 5 MARKS

d: Implement a class called CommissionWorker, derived from Employee. A commission worker is paid a base salary per week and an additional bonus based on the number of items sold during the past week. The class should include a constructor and earnings() method. 5 MARKS

e: Write a short driver program that creates an object for each of the employee sub-classes, it then calls the earnings() method for each object and displays the results. 5 MARKS

2.a: Create a standalone Java application that will count and sum up the number of lines in the text file passed as an argument on the command line. The program should create a **FileReader** object and pass this in the constructor of a **LineNumberReader** object to handle the file reading required.

The **LineNumberReader** class has two useful methods (that could be used):

public String readLine() throws IOException; This method reads a line of text. It returns a String containing the contents of the line, not including any line-termination characters, or null if the end of the stream has been reached.

public int getLineNumber(); This method returns the current line number.

10 MARKS

b: 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: What types of Sockets are supported in the Java networking package? Which type of Socket would you recommend for a VOIP type application and a File Transfer type application? 5 MARKS

b: Write a Java application that uses Stream type sockets to exchange Java Objects using object serialisation. The client side should connect to the server and send it a String object. The server should print out the String and respond to the client with a text based response encapsulated in another String Object. The client should receive the String Object from the server and print out this response.

10 MARKS

c: Write another Java application with the same functionality as outlined in part b of this question, but this time using Datagram type sockets. Hint: you can use ByteArrayOutputStream and ByteArrayInputStream to populate and read the array associated with the DatagramPacket object. 10 MARKS

4.a: Discuss briefly the differences between a process and a thread. What is the best way to stop executing threads (assuming they still haven't finished their work)?
5 MARKS

b: Write a JAVA animation applet that uses a thread to continuously scroll a text message across the screen from right to left. The message itself and the rate at which the text scrolls can be passed to the applet as HTML based parameters.
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 exactly once and there may be multiple producer and consumer threads executing (and attempting to access the buffer) concurrently.
10 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 constructor method(s) that enable objects of this class to be fully initialized.
5 MARKS

b: Provide a **public** method to add two **Complex** numbers: the real part of the right operand is added to the real part of the left operand, the imaginary part of the right operand is added to the imaginary part of the left operand. 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. 7 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 7 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 and write a short driver program to test your class. 6 MARKS

6. Using Java Remote Method Invocation, write the Java code for a remote compute server that could be used to remotely execute arbitrary Task objects. The server allows clients to submit Task objects, that is objects that implement the Task interface, for remote execution on the server and are then returned the result as a Java object. The following Java interfaces / classes should be provided:

    - *Compute* - this remote interface should provide a method to upload Task objects to the server and to then run the task and return the result back to the client when execution is complete. 4 MARKS

    - *Task* - this interface should define an arbitrary task object that may be passed as a parameter to the compute server. 4 MARKS

    - *MathTask* – this class provides an implementation of the Task interface and is used to perform some calculation that returns an Integer object. The calculation itself can be just some simple arithmetic e.g. add two numbers. 6 MARKS

    - *ComputeServer* - this class should provide an implementation of the Compute interface as well as the code required to initialise the server and make the remote object locatable for clients in the RMI registry. The server runtime should be protected so that objects uploaded to the server can not cause any harm. 6 MARKS

    - *ComputeClient* – this should provide a simple client program that creates a MathTask object and submits it to the server for remote execution and then displays the result. 5 MARKS

    The design of the system should make it possible for new Task classes to be easily added to the system in the future, making the system very flexible. The design should use Java RMI and Object Serialisation to submit Task objects and to return the result back to the client.

NUI Galway
OÉ Gaillimh

Spring Examinations 2013 / 2014

Exam Code(s)	3BCT, 3BP1
Exam(s)	Third Year Computer Science & Information Technology Third Year Electronic and Computer Engineering
Module Code(s)	CT326
Module(s)	Programming III
Paper No.	1
External Examiner(s)	Dr. J. Power
Internal Examiner(s)	Prof. G. Lyons Dr. M. Madden *Dr. D. Chambers
Instructions:	Answer any 4 questions. All questions carry equal marks.
Duration	2 hrs
No. of Pages	5
Department(s)	Information Technology
Requirements	None

1.a: Write an **abstract** Java class called **Account** to hold information about bank accounts. The **Account** class should contain suitable **private** instance variables to represent the account number, the balance of the account, the name of the account holder and the address of the holder. It should also contain a **List** of transaction objects for that account. Provide a constructor method that enables an object of this class to be fully initialised when it is declared. Also provide other **public** methods to facilitate simple transactions on the account object and to retrieve the various account attributes (e.g. the balance, account holder details, transactions). Provide the implementation for a suitable class that contains details about past transactions. The Transaction class will need suitable **private** instance variables to represent the type of tansaction, the amount, the date / time, and the balance after the transaction was applied. You can use the **java.util.Date** class to represent the date / time within the Transaction class. The list of transactions in the Account will be stored in the order the transactions were made. 10 MARKS

b: Write another Java class called **SavingsAccount** that inherits the **Account** class. This derived class should contain an additional **private** instance variable called **annualInterestRate** and an additional method called **addMonthlyInterest** that calculates the monthly interest and adds it to the account balance. The interest should be based on the closing balance each day for the past month. The closing balance can be obtained by iterating through the list of Transaction objects for the past month and then multiplying the closing account balance, as stored in the final Transaction object for each day, by **annualInterestRate** and then divide this amount by 365; this daily interest should then be added to the account balance. Provide a suitable constructor method that enables an object of this class to be fully initialised when it is declared. You might find the following code sample useful in using and comparing date objects :

```
Calendar cal = Calendar.getInstance();
cal.setTime(someDate); // someDate is a Date object
int day = cal.get(Calendar.DAY_OF_MONTH);
```

10 MARKS

c: Finally, write a driver program to test class **SavingsAccount**. Instantiate two **SavingsAccount** objects called **saver1** and **saver2**, with initial account balances of €2000 and €3000, respectively. This program should set **annualInterestRate** to 4%, then calculate the monthly interest and print the new balances for each of the savers. 5 MARKS

2.a Write a Java program that uses an ArrayList to store a collection of Vehicle 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 Vehicle objects based on their engine size. The Vehicle class should implement an accessor method called getEngineSize() that returns an int representing the engine size for that vehicle e.g. 1895. Finally, use the version of the Collections.sort() method that allows you to pass your own Comparator object to sort the list of Vehicle objects.
12 MARKS

b: The JDK contains two general-purpose List implementations i.e. *ArrayList* and *LinkedList*. Why is *ArrayList* generally the best performing implementation? Describe the cricumstances under which *LinkedList* might offer better performance. Describe the polymorphic algorithms provided in the JAVA Collections framework. In relation to these algorithms, explain fully the purpose and operaton of the following code idiom:

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

13 MARKS

3. Write a Java class called **Rational** for performing arithmetic with fractions.

Use integer variables to represent the **private** instance variables of the class - the **numerator** and the **denominator**. Provide a constructor method that enables an object of this class to be initialised when it is declared e.g. the fraction 2/3 would be stored in the object as 2 in the **numerator** and 3 in the **denominator**.
4 MARKS

Provide **public** methods for each of the following:

(a) Addition of two **Rational** numbers. The result should be stored in the target object e.g. if **r1** and **r2** are objects of type **Rational**, calling **r1.add(r2)** would add the value of **r2** to **r1** and then store the new value in **r1**. (Hint: adding 2/3 and 3/4 gives the result 17/12). 7 MARKS

(b) Multiplication of two **Rational** numbers. In the same way as for (a), the result should be stored in the target object. (Hint: multiplying 2/3 and 3/4 gives the result 6/12). 7 MARKS

(c) Printing **Rational** numbers in the form **a/b**, where **a** is the **numerator** and **b** is the **denominator**. 3 MARKS

Finally, write a suitable driver program that could be used to test your class.
4 MARKS

4.a: 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 String object 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. 12 MARKS

b: Write another Java application with the same functionality as outlined above, in part a of this question, but this time using Datagram type sockets. Hint: you can use ByteArrayOutputStream and ByteArrayInputStream to populate and read the array associated with the DatagramPacket object. This application does not need to implement a reliable data transfer protocol. 13 MARKS

5. Using Java Remote Method Invocation, write the Java code for a remote compute server that could be used to remotely execute arbitrary Task objects. The server allows clients to submit Task objects, that is objects that implement the Task interface, for remote execution on the server and are then returned the result as a Java object. The following Java interfaces / classes should be provided:

- *Compute* - this remote interface should provide a method to upload Task objects to the server and to then run the task and return the result back to the client when execution is complete. 4 MARKS

- *Task* - this interface should define an arbitrary task object that may be passed as a parameter to the compute server. 4 MARKS

- *MathTask* – this class provides an implementation of the Task interface and is used to perform some calculation that returns an Integer object. The calculation itself can be just some simple arithmetic e.g. add two numbers. 6 MARKS

- *ComputeServer* - this class should provide an implementation of the Compute interface as well as the code required to initialise the server and make the remote object locatable for clients in the RMI registry. The server runtime should be protected so that objects uploaded to the server can not cause any harm. 6 MARKS

- *ComputeClient* – this should provide a simple client program that creates a MathTask object and submits it to the server for remote execution and then displays the result. 5 MARKS

The design of the system should make it possible for new Task classes to be easily added to the system in the future, making the system very flexible. The design should use Java RMI and Object Serialisation to submit Task objects and to return the result back to the client.

6.a: Evaluate the following code sample (in terms of good design practice). The code is for a simple home heating system. The system is turned on if the current temperature falls below some minimum value. It's then turned off again when it goes above the maximum value. The class is instantiated and started i.e. it runs in its own thread of execution.

```
class Thermostat extends Thread
{
        private final int THERMOMETER = 0x10;
        private final int HEATER = 0xf7;
        private final int HEATER_ON = 0x1;
        private final int HEATER_OFF = 0;
        public void run()
        {
            while (true)
            {
                while (read(THERMOMETER) > min)
                        sleep(100);
                write(HEATER, HEATER_ON);
                while (read(THERMOMETER) < max)
                        sleep(100);
                write(HEATER, HEATER_OFF);
            }
        }
}
```

What is wrong with this code? Suggest a better design approach based on using the dependency inversion principle. 10 MARKS

b: 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 exactly once and there may be multiple producer and consumer threads executing (and attempting to access the buffer) concurrently.
15 MARKS

NUI Galway
OÉ Gaillimh

Autumn Examinations 2013

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. M. O'Boyle
Internal Examiner(s)	Prof. G. Lyons Dr. M. Madden *Dr. D. Chambers
Instructions:	Answer any 4 questions. All questions carry equal marks.
Duration	3 hrs
No. of Pages	5
Department(s)	Information Technology
Requirements	None

1. Develop a simple Java based payroll system that can calculate the weekly pay due for different categories of employees. The system should be implemented using the following design guidelines:

a: Implement an *abstract* base class called Employee that is used to hold and access basic information about an employee e.g. name, address, etc. This class should also define an *abstract* method called earnings() that returns the weekly pay for each type of employee. The class should include a suitable constructor and accessor methods to retrieve information about the employee. 5 MARKS

b: Implement a class called Manager, derived from Employee. A manager is paid a fixed weekly salary. The class should include a suitable constructor and should also implement the earnings() method. 5 MARKS

c: Implement a class called HourlyWorker, derived from Employee. An hourly worker is paid a fixed wager per hour, so in any given week they will be paind for the number of hours worked in the past week. The class should include a constructor and implement the earnings() method. 5 MARKS

d: Implement a class called CommissionWorker, derived from Employee. A commission worker is paid a base salary per week and an additional bonus based on the number of items sold during the past week. The class should include a constructor and earnings() method. 5 MARKS

e: Write a short driver program that creates an object for each of the employee sub-classes, it then calls the earnings() method for each object and displays the results. 5 MARKS

2.a: Describe the general structure and purpose of the IO Streams classes provided in the Java programming environment. What Java Class is used to support random file access? 4 MARKS

b: Write a Java application that inputs a date as a string in the form 23-03-2013 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: What types of Sockets are supported in the Java networking package? Which type of Socket would you recommend for a VOIP type application and a File Transfer type application? 5 MARKS

b: Write a Java application that uses Stream type sockets to exchange Java Objects using object serialisation. The client side should connect to the server and send it a String object. The server should print out the String and respond to the client with a text based response encapsulated in another String Object. The client should receive the String Object from the server and print out this response. 10 MARKS

c: Write another Java application with the same functionality as outlined above, in part b of this question, but this time using Datagram type sockets. Hint: you can use ByteArrayOutputStream and ByteArrayInputStream to populate and read the array associated with the DatagramPacket object. 10 MARKS

4.a: Discuss briefly the differences between a process and a thread. What is the best way to stop executing threads be stopped (assuming they still haven't finished their work)? 5 MARKS

b: Write a JAVA animation applet that uses a thread to continuously scroll a text message across the screen from right to left. The message itself and the rate at which the text scrolls can be passed to the applet as HTML based parameters. 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

5. Assume that a Sports Club at the University 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 student details. The class should include methods for updating member details and querying their registration status i.e. are they fully paid up members of the club. 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 name or 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

6. Using Java Remote Method Invocation, write the Java code for a remote compute server that could be used to remotely execute arbitrary Task objects. The server allows clients to submit Task objects, that is objects that implement the Task interface, for remote execution on the server and are then returned the result as a Java object. The following Java interfaces / classes should be provided:

    - *Compute* - this remote interface should provide a method to upload Task objects to the server and to then run the task and return the result back to the client when execution is complete. 4 MARKS

    - *Task* - this interface should define an arbitrary task object that may be passed as a parameter to the compute server. 4 MARKS

    - *MathTask* – this class provides an implementation of the Task interface and is used to perform some calculation that returns an Integer object. The calculation itself can be just some simple arithmetic e.g. add two numbers. 6 MARKS

    - *ComputeServer* - this class should provide an implementation of the Compute interface as well as the code required to initialise the server and make the remote object locatable for clients in the RMI registry. The server runtime should be protected so that objects uploaded to the server can not cause any harm. 6 MARKS

    - *ComputeClient* – this should provide a simple client program that creates a MathTask object and submits it to the server for remote execution and then displays the result. 5 MARKS

    The design of the system should make it possible for new Task classes to be easily added to the system in the future, making the system very flexible. The design should use Java RMI and Object Serialisation to submit Task objects and to return the result back to the client.

NUI Galway
OÉ Gaillimh

Spring Examinations 2012 / 2013

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. M. O'Boyle
Internal Examiner(s)	Prof. G. Lyons Dr. M. Madden *Dr. D. Chambers
Instructions:	Answer any 4 questions. All questions carry equal marks.
Duration	3 hrs
No. of Pages	5
Department(s)	Information Technology
Requirements	None

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? Do all the methods in an **abstract** superclass have to be declared as **abstract**? 5 MARKS

c: Consider the inheritance hierarchy of **Figure** 1 below. For each class, indicate the 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 including constructors and other methods required. The base Account class should be declared as an abstract class and details of all transactions (debits and credits) should be stored in an ArrayList of Transaction objects i.e. define a class called Transaction suitable for this purpose. 15 MARKS

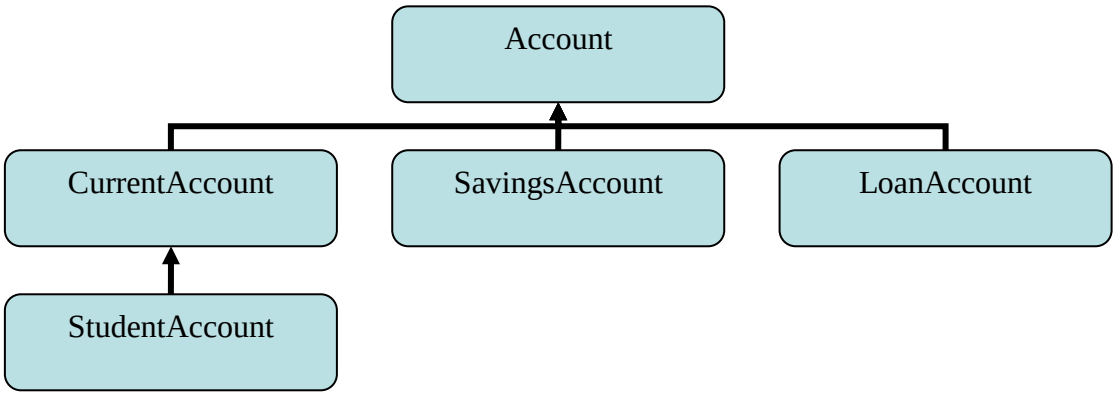


Figure 1 - Inheritance Hierarchy for Bank Accounts

2.a: Create a standalone Java application that will count and sum up the number of lines in the text file passed as an argument on the command line. The program should create a **FileReader** object and pass this in the constructor of a **LineNumberReader** object to handle the file reading required.

The **LineNumberReader** class has two useful methods (that could be used):

public String readLine() throws IOException; This method reads a line of text. It returns a String containing the contents of the line, not including any line-termination characters, or null if the end of the stream has been reached.

public int getLineNumber(); This method returns the current line number.

10 MARKS

b: 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: What types of Sockets are supported in the Java networking package? Which type of Socket would you recommend for a VOIP type application and a File Transfer type application? 5 MARKS

b: Write a Java application that uses Stream type sockets to exchange Java Objects using object serialisation. The client side should connect to the server and send it a String object. The server should print out the String and respond to the client with a text based response encapsulated in another String Object. The client should receive the String Object from the server and print out this response.

10 MARKS

c: Write another Java application with the same functionality as outlined above, in part b of this question, but this time using Datagram type sockets. Hint: you can use ByteArrayOutputStream and ByteArrayInputStream to populate and read the array associated with the DatagramPacket object. 10 MARKS

4.a: Discuss briefly the differences between a process and a thread. What is the best way to stop executing threads be stopped (assuming they still haven't finished their work)? 5 MARKS

b: Write a JAVA animation applet that uses a thread to continuously scroll a text message across the screen from right to left. The message itself and the rate at which the text scrolls can be passed to the applet as HTML based parameters. 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

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 constructor method(s) that enable object of this class to be fully initialized. 5 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. 7 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 7 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 and write a short driver program to test your class. 6 MARKS

6. Using Java Remote Method Invocation, write the Java code for a remote compute server that could be used to remotely execute arbitrary Task objects. The server allows clients to submit Task objects, that is objects that implement the Task interface, for remote execution on the server and are then returned the result as a Java object. The following Java interfaces / classes should be provided:

    - *Compute* - this remote interface should provide a method to upload Task objects to the server and to then run the task and return the result back to the client when execution is complete. 4 MARKS

    - *Task* - this interface should define an arbitrary task object that may be passed as a parameter to the compute server. 4 MARKS

    - *MathTask* – this class provides an implementation of the Task interface and is used to perform some calculation that returns an Integer object. The calculation itself can be just some simple arithmetic e.g. add two numbers. 6 MARKS

    - *ComputeServer* - this class should provide an implementation of the Compute interface as well as the code required to initialise the server and make the remote object locatable for clients in the RMI registry. The server runtime should be protected so that objects uploaded to the server can not cause any harm. 6 MARKS

    - *ComputeClient* – this should provide a simple client program that creates a MathTask object and submits it to the server for remote execution and then displays the result. 5 MARKS

    The design of the system should make it possible for new Task classes to be easily added to the system in the future, making the system very flexible. The design should use Java RMI and Object Serialisation to submit Task objects and to return the result back to the client.

NUI Galway
OÉ Gaillimh

Autumn Examinations 2012

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. M. O'Boyle
Internal Examiner(s)	Prof. G. Lyons Dr. M. Madden *Dr. D. Chambers
Instructions:	Answer any 4 questions. All questions carry equal marks.
Duration	3 hrs
No. of Pages	5
Department(s)	Information Technology
Requirements	None

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. 15 MARKS

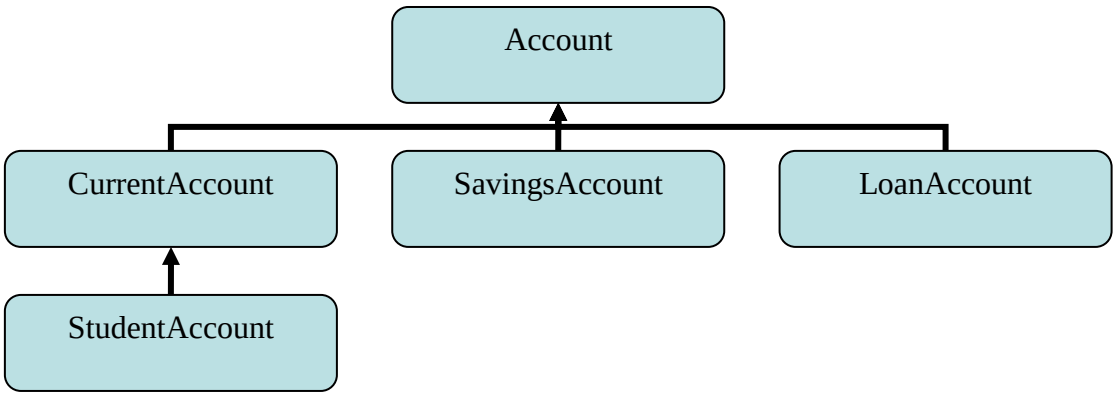


Figure 1 - Inheritance Hierarchy for Bank Accounts

2.a: Describe the general structure and purpose of the IO Streams classes provided in the Java programming environment. What Java Class is used to support random file access? 4 MARKS

b: Write a Java application that inputs a date as a string in the form 17/02/2010 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: Describe briefly the purpose and operation of the following code idiom:

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

5 MARKS

b: Create a standalone Java application that will count and sum up the number of lines in the text file passed as an argument on the command line. The program should create a **FileReader** object and pass this in the constructor of a **LineNumberReader** object to handle the file reading required.

The **LineNumberReader** class has two useful methods (that could be used):

public String readLine() throws IOException; This method reads a line of text. It returns a String containing the contents of the line, not including any line-termination characters, or null if the end of the stream has been reached.

public int getLineNumber(); This method returns the current line number.

10 MARKS

c: Evaluate the following code sample (in terms of good design practice). The code is for a simple home heating system. The system is turned on if the current temperature falls below some minimum vlaue. It's then turned off again when it goes above the maximum value. The class is instantiated and started i.e. it runs in its own thread of execution.

```
class Thermostat extends Thread
{
        private final int THERMOMETER = 0x10;
        private final int HEATER = 0xf7;
       private final int HEATER_ON = 0x1;
       private final int HEATER_OFF = 0;
       public void run()
       {
            while (true)
            {
                 while (read(THERMOMETER) > min)
                        sleep(100);
                 write(HEATER, HEATER_ON);
                  while (read(THERMOMETER) < max)
                        sleep(100);
                  write(HEATER, HEATER_OFF);
            }
       }
}
```

What's wrong with this design? Suggest a better design approach based on using the dependency inversion principle. 10 MARKS

4.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: Write a JAVA animation applet that uses a thread to continuously scroll a text message across the screen from right to left. The message itself and the rate at which the text scrolls can be passed to the applet as HTML based parameters. 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

5.a: What types of Sockets are supported in the Java networking package? Which type of Socket would you recommend for a VOIP type application and a File Transfer type application? 5 MARKS

b: Write a Java application that uses Stream type sockets to exchange Java Objects using object serialisation. The client side should connect to the server and send it an Integer Object. The server should print out this value and respond to the client with a text based response encapsulated in a String Object. The client should receive the String Object from the server and print out this response. 10 MARKS

c: Write another Java application with the same functionality as outlined above, in part b of this question, but this time using Datagram type sockets. Hint: you can use ByteArrayOutputStream and ByteArrayInputStream to populate and read the array associated with the DatagramPacket object. 10 MARKS

6. Assume that a Sports Club at the University 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 student details. The class should include methods for updating member details and querying their registration status i.e. are they fully paid up members of the club. 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 name or 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

NUI Galway
OÉ Gaillimh

Spring Examinations 2011 / 2012

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. M. O'Boyle
Internal Examiner(s)	Prof. G. Lyons Dr. M. Madden Dr. D. Chambers
Instructions:	Answer any 4 questions. All questions carry equal marks.
Duration	3 hrs
No. of Pages	5
Department(s)	Information Technology
Requirements	None

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

```
import java.io.*;
public class Account implements Comparable<Account>,
Serializable {
     protected int accnum;
     protected HolderDetails holder;
     protected List<Transaction>  transactions;
     protected float balance;

     // Add a suitable constructor here

     // Add methods to make deposits / withdrawals

     // Method to print out account transaction summary

     // Add suitable attribute accessor methods

     // Add method to implement the Comparable interface
}
```

a: Complete the implementation of the Account class, providing a suitable constructor, attribute accessor methods, methods for making a deposit or withdrawal, a method to print out a transaction summary related to a range of Dates and an implementation method for the Comparable interface.
7 MARKS

b: Provide implementations for the HolderDetails class and the Transaction class. The HolderDetails class is used to store personal details about the account holder. The Transaction class contains details about past transactions including the type of transaction, the amount and the Date. 7 MARKS

c: Define and implement a new class, called CurrentAccount, derived from Account, that allows withdrawals to proceed up to some overdraft limit. Note that the base Account calss shown has no overdraft facility.
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. Also briefly describe the 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 17-07-2010 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 Student class that includes an id number, a name, and course details and a suitable constructor method. Then write a Java program that uses an ArrayList to store a collection of Student 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 Student 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 Student 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 exactly once and there may be multiple Producer and Consumer threads executing (and attempting to access the buffer) concurrently. 10 MARKS

4.a: Suppose that you've written a program that displays two messages, as follows:

```
public class NotI18N {
    static public void main(String[] args) {
        System.out.println("Hello.");
        System.out.println("How are you?");
     }
}
```

You then decide that this program needs to display the same or similar messages for people living in France and Spain. Outline the steps needed to properly internationalise this program i.e. the hardcoded English language messages should be removed and replaced with a more flexible mechanism that will facilitate additional language support in the future. 10 MARKS

b: Write 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 set up 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? Which type of Socket would you recommend for a VOIP type application and a File Transfer type application? 5 MARKS

b: Write a Java application that uses Stream type sockets to exchange Java Objects using object serialisation. The client side should connect to the server and send it an Integer Object. The server should print out this value and respond to the client with a text based response encapsulated in a String Object. The client should receive the String Object from the server and print out this response. 10 MARKS

c: Write another Java application with the same functionality as outlined above, in part b of this question, but this time using Datagram type sockets. Hint: you can use ByteArrayOutputStream and ByteArrayInputStream to populate and read the array associated with the DatagramPacket object. 10 MARKS

6. Assume that a Sports Club at the University 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 student details. The class should include methods for updating member details and querying their registration status i.e. are they fully paid up members of the club. 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 name or 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

Autumn Examinations 2011

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. M. O'Boyle
Internal Examiner(s)	Prof. G. Lyons Dr. J. Duggan Dr. D. Chambers

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

Duration	3 hrs
No. of Pages	5
Department(s)	Information Technology
Requirements	None

1: 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. 5 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. 6 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 6 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. 4 MARKS

e: Write a short driver program to test your class. 4 MARKS

2.a: Suppose that you've written a program that displays two messages, as follows:

```
public class NotI18N {
    static public void main(String[] args) {
        System.out.println("Hello.");
        System.out.println("How are you?");
} }
```

You then decide that this program needs to display the same or similar messages for people living in France and Germany. Outline the steps needed to properely internationalise this program i.e. the hardcoded English language messages should be removed and replaced with a more flexible mechanism that will facilitate additional language support in the future. 10 MARKS

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

(i) Application class implements the Runnable interface.
(ii) Application class extends the Thread class. 5 MARKS

b: What is meant by the term *deadlock*? Using the example of the *Dining Philosophers Problem* (covered in class), discuss how deadlock might occur in this case and propose a solution to overcome the problem. 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. Develop a simple Java based payroll system that can calculate the weekly pay due for different categories of employees. The system should be implemented using the following design guidelines:

a: Implement an *abstract* base class called Employee that is used to hold and access basic information about an employee e.g. name, address, etc. This class should also define an *abstract* method called earnings() that returns the weekly pay for each type of employee. The class should include a suitable constructor and accessor methods to retrieve information about the employee. 5 MARKS

b: Implement a class called Manager, derived from Employee. A manager is paid a fixed weekly salary. The class should include a suitable constructor and should also implement the earnings() method. 5 MARKS

c: Implement a class called HourlyWorker, derived from Employee. An hourly worker is paid a fixed wager per hour, so in any given week they will be paind for the number of hours worked in the past week. The class should include a constructor and implement the earnings() method. 5 MARKS

d: Implement a class called CommissionWorker, derived from Employee. A commission worker is paid a base salary per week and an additional bonus based on the number of items sold during the past week. The class should include a constructor and earnings() method. 5 MARKS

e: Write a short driver program that creates an object for each of the employee sub-classes, it then calls the earnings() method for each object and displays the results. 5 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 simple text messages i.e. one side sends a single String value to the other and it then receives another String value 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 single integer value to the server; the server then responds with another single integer value back to the Client. 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

Spring Examinations 2010 / 2011

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. M. O'Boyle
Internal Examiner(s)	Prof. G. Lyons Dr. J. Duggan Dr. D. Chambers

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

Duration	3 hrs
No. of Pages	5
Department(s)	Information Technology
Requirements	None

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. 15 MARKS

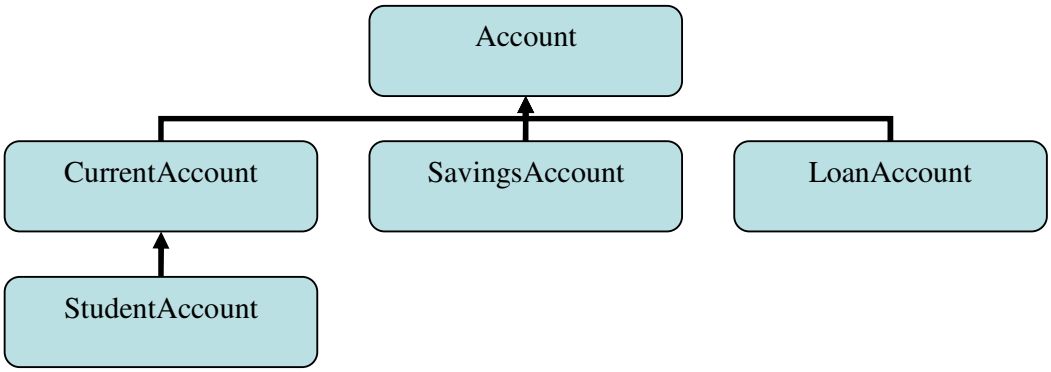


Figure 1 - Inheritance Hierarchy for Bank Accounts

2.a: Describe the general structure and purpose of the IO Streams classes provided in the Java programming environment. What Java Class is used to support random file access? 4 MARKS

b: Write a Java application that inputs a date as a string in the form 17/02/2010 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: Describe briefly the purpose and operation of the following code idiom:

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

5 MARKS

b: Create a standalone Java application that will count and sum up the number of lines in the text file passed as an argument on the command line. The program should create a **FileReader** object and pass this in the constructor of a **LineNumberReader** object to handle the file reading required.

The **LineNumberReader** class has two useful methods (that could be used):

public String readLine() throws IOException; This method reads a line of text. It returns a String containing the contents of the line, not including any line-termination characters, or null if the end of the stream has been reached.

public int getLineNumber(); This method returns the current line number.

10 MARKS

c: Evaluate the following code sample (in terms of good design practice). The code is for a simple home heating system. The system is turned on if the current temperature falls below some minimum vlaue. It's then turned off again when it goes above the maximum value. The class is instantiated and started i.e. it runs in its own thread of execution.

```
class Thermostat extends Thread
{
        private final int THERMOMETER = 0x10;
        private final int HEATER = 0xf7;
        private final int HEATER_ON = 0x1;
        private final int HEATER_OFF = 0;
        public void run()
        {
            while (true)
            {
                while (read(THERMOMETER) > min)
                        sleep(100);
                write(HEATER, HEATER_ON);
                while (read(THERMOMETER) < max)
                        sleep(100);
                write(HEATER, HEATER_OFF);
            }
        }
}
```

What's wrong with this design? Suggest a better design approach based on using the dependency inversion principle. 10 MARKS

4.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: Write a JAVA animation applet that uses a thread to continuously scroll a text message across the screen from right to left. The message itself and the rate at which the text scrolls can be passed to the applet as HTML based parameters.
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

5. 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.
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.
6 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. 8 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.
5 MARKS

6.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 simple text messages i.e. one side sends a single String value to the other and it then receives another String value 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 single integer value to the server; the server then responds with another single integer value back to the Client. 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

Autumn Examinations 2010

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. M. O'Boyle
Internal Examiner(s)	Dr. D. Chambers Dr. J. Duggan Prof. G. Lyons

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

Duration	3 hrs
No. of Pages	5
Department(s)	Information Technology
Requirements	None

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 17/07/2010 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: Suppose that you've written a program that displays two messages, as follows:

```
public class NotI18N {
    static public void main(String[] args) {
        System.out.println("Hello.");
        System.out.println("How are you?");
} }
```

You then decide that this program needs to display the same or similar messages for people living in France and Germany. Outline the steps needed to properely internationalise this program i.e. the hardcoded English language messages should be removed and replaced with a more flexible mechanism that will facilitate additional language support in the future. 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

NUI Galway
OÉ Gaillimh

Spring Examinations 2009 / 2010

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. M. O'Boyle
Internal Examiner(s)	Dr. D. Chambers Dr. J. Duggan

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

Duration	3 hrs
No. of Pages	5
Department(s)	Information Technology
Requirements	None

1. Develop a simple Java based payroll system that can calculate the weekly pay due for different categories of employees. The system should be implemented using the following design guidelines:

   a: Implement an *abstract* base class called Employee that is used to hold and access basic information about an employee e.g. name, address, etc. This class should also define an *abstract* method called earnings() that returns the weekly pay for each type of employee. The class should include a suitable constructor and accessor methods to retrieve information about the employee. 5 MARKS

   b: Implement a class called Manager, derived from Employee. A manager is paid a fixed weekly salary. The class should include a suitable constructor and should also implement the earnings() method. 5 MARKS

   c: Implement a class called HourlyWorker, derived from Employee. An hourly worker is paid a fixed wager per hour, so in any given week they will be paind for the number of hours worked in the past week. The class should include a constructor and implement the earnings() method. 5 MARKS

   d: Implement a class called CommissionWorker, derived from Employee. A commission worker is paid a base salary per week and an additional bonus based on the number of items sold during the past week. The class should include a constructor and earnings() method. 5 MARKS

   e: Write a short driver program that creates an object for each of the employee sub-classes, it then calls the earnings() method for each object and displays the results. 5 MARKS

2.a: Describe the general structure and purpose of the IO Streams classes provided in the Java programming environment. What Java Class is used to support random file access? 4 MARKS

b: Write a Java application that inputs a date as a string in the form 17/02/2010 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 Bank Account class that includes an account number, holder details, the balance and a suitable constructor method. Then write a Java program that uses an ArrayList to store a collection of Account 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 Account objects based on their balance. Finally, use the version of the Collections.sort() method that allows you to pass your own Comparator object to sort the list of Account 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: Suppose that you've written a program that displays two messages, as follows:

```
public class NotI18N {
   static public void main(String[] args) {
      System.out.println("Hello.");
      System.out.println("How are you?");
} }
```

You then decide that this program needs to display the same or similar messages for people living in France and Germany. Outline the steps needed to properely internationalise this program i.e. the hardcoded English language messages should be removed and replaced with a more flexible mechanism that will facilitate additional language support in the future. 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? 5 MARKS

b: Write a Java application that uses Stream type sockets to exchange Java Objects using object serialisation. The client side should connect to the server and send it an Integer Object. The server should print out this value and respond to the client with a text based response encapsulated in a String Object. The client should receive the String Object from the server and print out this response. 10 MARKS

c: Implement another Java application with the same functionality as outlined above, in part b of this question, but this time using Datagram type sockets. Hint: you can use ByteArrayOutputStream and ByteArrayInputStream to populate and read the array associated with the DatagramPacket object. 10 MARKS

6. Write a Java class called **Rational** for performing arithmetic with fractions.

Use integer variables to represent the **private** instance variables of the class - the **numerator**and the **denominator**. Provide a constructor method that enables an object of this class to be initialised when it is declared e.g. the fraction 2/3 would be stored in the object as 2 in the **numerator** and 3 in the **denominator**. Also provide a no-argument constructor with default values in case no initialisers are provided.
3 MARKS

Provide **public** methods for each of the following:

(a) Addition of two **Rational** numbers. The result should be stored in the target object e.g. if **r1** and **r2** are objects of type **Rational**, calling **r1.add(r2)** would add the value of **r2** to **r1** and then store the new value in **r1**. (Hint: adding 2/3 and 3/4 gives the result 17/12). 4 MARKS

(b) Subtraction of two **Rational** numbers. In the same way as for (a), the result should be stored in the target object. (Hint: subtracting 2/3 from 3/4 gives the result 1/12). 4 MARKS

(c) Multiplication of two **Rational** numbers. In the same way as for (a), the result should be stored in the target object. (Hint: multiplying 2/3 and 3/4 gives the result 6/12). 4 MARKS

(d) Division of two **Rational** numbers. In the same way as for (a), the result should be stored in the target object. (Hint: dividing 2/3 by 3/4 gives the result 8/9). 4 MARKS

(e) Printing **Rational** numbers in the form **a/b**, where **a** is the **numerator** and **b** is the **denominator**. 3 MARKS

Finally, write a suitable driver program that could be used to test your class.
3 MARKS

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

Autumn Examinations 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: 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. 5 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. 6 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 6 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. 4 MARKS

e: Write a short driver program to test your class. 4 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? 5 MARKS

b: What information is normally written out during object serialisation in the Java programming environment? Using a suitable example, describe how you can provide custom serialisation for your own classes. 10 MARKS

c: Write a Java applicaion that prompts the user to input their Name, Address, Date of Birth and Student ID number using either the standard input *System.in* or a GUI based input dialog - 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. 10 MARKS

3.a: Discuss briefly the differences between a process and a thread. Show (using simple code examples) how threads may be created (and started) using the following mechanisms:

(i) Application class implements the Runnable interface.

(ii) Application class extends the Thread class. 5 MARKS

b: What is meant by the term *deadlock*? Using the example of the *Dining Philosophers Problem* (covered in class), discuss how deadlock might occur in this case and propose a solution to overcome the problem. 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. Develop a simple Java based payroll system that can calculate the weekly pay due for different categories of employees. The system should be implemented using the following design guidelines:

a: Implement an *abstract* base class called Employee that is used to hold and access basic information about an employee e.g. name, address, etc. This class should also define an *abstract* method called earnings() that returns the weekly pay for each type of employee. The class should include a suitable constructor and accessor methods to retrieve information about the employee. 5 MARKS

b: Implement a class called Manager, derived from Employee. A manager is paid a fixed weekly salary. The class should include a suitable constructor and should also implement the earnings() method. 5 MARKS

c: Implement a class called HourlyWorker, derived from Employee. An hourly worker is paid a fixed wager per hour, so in any given week they will be paind for the number of hours worked in the past week. The class should include a constructor and implement the earnings() method. 5 MARKS

d: Implement a class called CommissionWorker, derived from Employee. A commission worker is paid a base salary per week and an additional bonus based on the number of items sold during the past week. The class should include a constructor and earnings() method. 5 MARKS

e: Write a short driver program that creates an object for each of the employee sub-classes, it then calls the earnings() method for each object and displays the results. 5 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

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

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

Autumn Examinations 2008

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 will be marked equally.

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

1. Write a Java class called **Rational** for performing arithmetic with fractions.

   Use integer variables to represent the **private** instance variables of the class - the **numerator**and the **denominator**. Provide a constructor method that enables an object of this class to be initialised when it is declared e.g. the fraction 2/3 would be stored in the object as 2 in the **numerator** and 3 in the **denominator**. Also provide a no-argument constructor with default values in case no initialisers are provided. Provide **public** methods for each of the following:

   i. Addition of two **Rational** numbers. The result should be stored in the target object e.g. if **r1** and **r2** are objects of type **Rational**, calling **r1.add(r2)** would add the value of **r2** to **r1** and then store the new value in **r1**. (Hint: adding 2/3 and 3/4 gives the result 17/12).

   ii. Subtraction of two **Rational** numbers. In the same way as for (a), the result should be stored in the target object. (Hint: subtracting 2/3 from 3/4 gives the result 1/12).

   iii. Multiplication of two **Rational** numbers. In the same way as for (a), the result should be stored in the target object. (Hint: multiplying 2/3 and 3/4 gives the result 6/12).

   iv. Division of two **Rational** numbers. In the same way as for (a), the result should be stored in the target object. (Hint: dividing 2/3 by 3/4 gives the result 8/9).

   v. Printing **Rational** numbers in the form **a/b**, where **a** is the **numerator** and **b** is the **denominator**.

   Finally, write a suitable driver program that could be used to test your class.
   25 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? 5 MARKS

b: What information is normally written out during object serialisation in the Java programming environment? Using a suitable example, describe how you can provide custom serialisation for your own classes. 10 MARKS

c: Write a Java applicaion that prompts the user to input their Name, Address, Date of Birth and Student ID number using either the standard input *System.in* or a GUI based input dialog - 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. 10 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: 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();
      }
```
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. 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

6.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? 5 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.
10 MARKS

c: Suppose that you've written a program that displays three messages, as follows:

```
public class NotI18N {

    static public void main(String[] args) {

        System.out.println("Hello.");
        System.out.println("How are you?");
        System.out.println("Goodbye.");
    }
}
```

You then decide that this program needs to display the same or similar messages for people living in France and Germany. Outline the steps needed to properely internationalise this program i.e. the hardcoded English language messages should be removed and replaced with a more flexible mechanism that will facilitate additional language support in the future. 10 MARKS

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

Spring Examinations, 2007/2008

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 will be marked equally.

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

1. Develop a simple Java based payroll system that can calculate the weekly pay due for different categories of employees. The system should be implemented using the following design guidelines:

a: Implement an *abstract* base class called Employee that is used to hold and access basic information about an employee e.g. name, address, etc. This class should also define an *abstract* method called earnings() that returns the weekly pay for each type of employee. The class should include a suitable constructor and accessor methods to retrieve information about the employee. 5 MARKS

b: Implement a class called Manager, derived from Employee. A manager is paid a fixed weekly salary. The class should include a suitable constructor and should also implement the earnings() method. 5 MARKS

c: Implement a class called HourlyWorker, derived from Employee. An hourly worker is paid a fixed wager per hour, so in any given week they will be paind for the number of hours worked in the past week. The class should include a constructor and implement the earnings() method. 5 MARKS

d: Implement a class called CommissionWorker, derived from Employee. A commission worker is paid a base salary per week and an additional bonus based on the number of items sold during the past week. The class should include a constructor and earnings() method. 5 MARKS

e: Write a short driver program that creates an object for each of the employee sub-classes, it then calls the earnings() method for each object and displays the results. 5 MARKS

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

b: Create a standalone Java application that will count and sum up the number of lines in the text file passed as an argument on the command line. The program should create a **FileReader** object and pass this in the constructor of a **LineNumberReader** object to handle the file reading required.
The **LineNumberReader** class has two useful methods (that could be used):

public String readLine() throws IOException; This method reads a line of text. It returns a String containing the contents of the line, not including any line-termination characters, or null if the end of the stream has been reached.

public int getLineNumber(); This method returns the current line number.
10 MARKS

c: Write a Java program that uses an ArrayList to store a collection of Integer 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 Integer objects. Then use the version of the Collections.sort() method that allows you to pass your own Comparator object to sort the list of Integer objects. 10 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: 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();
      }
```

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: 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. 5 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. 7 MARKS

(Question 5 is continued on the next page)

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 7 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: 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? 5 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.
10 MARKS

c: Suppose that you've written a program that displays three messages, as follows:

```
public class NotI18N {

    static public void main(String[] args) {

        System.out.println("Hello.");
        System.out.println("How are you?");
        System.out.println("Goodbye.");
    }
}
```

You then decide that this program needs to display the same or similar messages for people living in France and Germany. Outline the steps needed to properely internationalise this program i.e. the hardcoded English language messages should be removed and replaced with a more flexible mechanism that will facilitate additional language support in the future. 10 MARKS