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