GX2324

OLLSCOIL NA hÉIREANN, GAILLIMH

THE NATIONAL UNIVERSITY OF IRELAND, GALWAY

SUMMER EXAMINATIONS 2000/2001

Masters Degree in Information Technology

Masters Degree in Engineering Science

CT509: PROGRAMMING

Prof. D. Bell

Prof. G. Lyons

Dr. M. Madden

Time allowed: **three** hours

Answer any **four** questions

All questions carry **equal** marks

---

1. (a) Discuss, with examples, the use of inheritance and composition in object-oriented programming. (9)

(b) Java has six different relational operators. List them, and show how each is used. (3)

(c) Write a Java program that repeatedly asks the user to enter an integer number, and keeps track of the number of even and odd numbers entered. It must stop when the user enters 0 (zero), and display the total count of even numbers and the total count of odd numbers. The program must include the following methods:

(1) A method called **isEven**, that takes an **int** parameter and returns **true** if the int is even, or **false** if it is not.
Note: if a number is even, when it is divided by 2 the remainder is zero.

(2) A method called **getIntegerFromUser**, that takes no parameters and returns an int. The method must use an input dialog ask the user to enter a number, and return the value entered by the user.

(3) A **main** method, with the rest of the functionality required for the program to work. (13)

2. (a) What is the purpose of Java's **BufferedReader** and **BufferedWriter** classes? Demonstrate with code fragments how objects of these two classes are created. (6)

(b) Explain how **try, catch, finally** blocks are used in Java to handle exceptions. (7)

(c) Write a Java program that asks the user for the name of a file, and then counts the number of occurrences of the word "the" in the file. To achieve this, your program should use a stream tokenizer. Each time a token is read, if the token is a word the program should test whether it equals "the", ignoring case. (You can use the method **equalsIgnoreCase**, which is a member of the **String** class, to do this.) After the end of the file is reached, the program should display the number of occurrences.

Note: You do not need to verify the existence of the file before opening it, but you must include code to handle input/output exceptions and display an appropriate message if such an exception occurs. (12)

3. (a) You are part of a team employed by Athenry Aircon Automation to develop software for their new air conditioning system, **AAircon**. The system includes a cooler that can be either on or off, and a heater that can be set to 0 (off) or a number in the range 1 to 5 (minimum to maximum heat).

Write the code for a Java class called **AAircon**, with appropriate member variables for keeping track of its current heater and cooler settings. Include a constructor for the class, which sets both the heater and the cooler to off. The public interface of the class must include methods **turnOffCooler**, **turnOnCooler**, **turnOffHeater** (which sets the heat to 0), **turnOnHeater** (which sets the heat to 3, provided the heat is currently off), **turnUpHeater** (which increases the heat by 1, provided it is on and not already at the maximum), **turnDownHeater** (which decreases the heat by 1, provided it is on and not already at the minimum). Each of these methods must change the member variables as appropriate. Since the cooler and heater cannot both be operated simultaneously, the method to turn on the cooler should automatically turn off the heater, and *vice versa*.

Note: You do not need to display any messages or have other output from any of these methods. (12)

(b) The company also has plans for a luxury model, the **AAirconDelux**. This model will have an automatic mode, which will operate the cooler and heater automatically, based on the air temperature and the desired temperature.

Based on the **AAircon** class, write a class called **AAirconDelux**, which has member variables to keep track of whether automatic mode is on or off, and what the desired temperature is (a floating point number in the range 10-40). Include a constructor for the class, and additional public methods called **turnOnAuto** and **turnOffAuto**. The method **turnOnAuto** must have a single parameter, to specify the desired temperature. (6)

(c) For the **AAirconDelux** class, write a method called **control**, which has no parameters and no return value. (This method will be called from another part of the software system, which is being developed separately.) It does nothing if a unit is not in automatic mode. Otherwise, it compares the desired temperature with the air temperature (you can assume that a method called **getAirTemperature**, which has no parameters returns a floating point number, exists.) If they are the same, the method turns off both the heater and the cooler. If the desired temperature is lower, it turns on the cooler. If the desired temperature is higher, it turns on the heater. (7)

4. Discuss in detail **any four** of the following five Java topics, using code examples for illustration:

(1) Type-wrapper classes

(2) Constructors and finalizers

(3) Persistence and object serialization

(4) The Java Virtual Machine

(5) Polymorphism (25)

5. (a) Explain each of the following Java programming terms, using short code samples to illustrate your answer:

(1) Recursion

(2) Method overloading

(3) Keywords **break** and **continue** (12)

(b) Java has two main constructs for performing iteration. For each of these, explain its format and give an example of a situation where it would be the most appropriate option. (6)

(c) Java has three repetition constructs. For each of these, explain its format and give an example of a situation where it would be the most appropriate option. (7)

6. (a) Moycullen Manufacturing has commissioned your company to develop a software system for keeping track of their products. As part of this, your responsibility is to write a program to assist them with data entry.

Write a Java class called **Product** with private members for storing the product name (a text string), the product ID (an integer number), the unit price (a floating-point number) and the number of units sold (an integer number). The class should have three methods in its public interface:

(1) **askUserForData**, which uses message boxes to ask the user for the name, ID, unit price and units sold

(2) **getSalesAmount**, which multiplies the unit price by the number of units and returns a floating-point number

(3) **display**, which displays the name, ID, unit price, units sold and sales amount in a message dialog. (11)

(b) Write a second class containing the main method for the program. This method should have an ArrayList called **allProducts**, which will hold Product objects. It should build up a list of products, with the following procedure:

(1) Create a new Product object and call its **askUserForData** method

(2) Add the new product to the **allProducts** list

(3) Ask the user whether they want to enter details for another product

(4) If they do, repeat the procedure. (7)

(c) Add code to the main method to calculate and display the total sales for all products. This code must work by retrieving each Product object from the **allProducts** list in turn, calling the **getSalesAmount** for each individual product, and adding it to a running total. When the total sales for all products have been added up, the total sales should be displayed in a message box. (7)