UNIVERSITY COLLEGE LONDON

University of London

EXAMINATION FOR INTERNAL STUDENTS

For The Following Qualifications:-

B.Sc. *M.Sci.*

Information Studs. G4: Programming 1

COURSE CODE	:	**INSTG004**
UNIT VALUE	:	**0.50**
DATE	:	**08-MAY-06**
TIME	:	**14.30**
TIME ALLOWED	:	**2 Hours**

06-C0848-3-50

TURN OVER

Answer **ANY THREE** questions.

Note to candidates:

Throughout this exam paper you should use the *Scanner* class, as described and used in the course, for user input in programs, and you may assume that an appropriate import statement for this class has been included when necessary at the start of each program file. In particular you may make use of the following statements (where *variable* is an identifier of a variable of the appropriate type):

```
Scanner scan = new Scanner(System.in);

variable = scan.nextInt();

variable = scan.nextBoolean();

variable = scan.next().charAt(0); // to input a single char

variable = scan.nextDouble();

variable = scan.nextFloat();

variable = scan.next();            // to input a string

variable = scan.nextLine();        // to input a string
                                   // with spaces
```

PAPER CONTINUED ON NEXT PAGE

1. a) Briefly explain what are meant by *compilation errors*, *runtime errors* and *logical errors* in computer programming.

[3 marks]

b) What is meant by a *Boolean expression* in Java? What truth values (`true` or `false`) do the following expressions evaluate to?

```
!(3 <= 8 || 5 < 9) && 7 > 6

!(3 <= 8) || (5 < 9 && 7 > 6)
```

[5 marks]

c) Give an example of a Java *import declaration* and briefly explain its function. [3 marks]

d) To take the degree module "BB01" in a certain university, students must have first passed three other modules, "AA01", "AA02" and "AA03". In addition they must have gained a "distinction" in at least one of these modules. Students pass a module if they score 40% or more, and gain a distinction if they score 60% or more. Complete the following program with an appropriate sequence of "if" and "if ... else ... " statements so that it outputs either the message "input error", or the message "you may take module BB01", or the message "you cannot take module BB01". You should not add any more user input statements to the program.

```
import java.util.Scanner;
public class ModuleBB01checker
{
    public static void main(String[] args)
    {
        final int PASS_MARK = 40, DISTINCTION_MARK = 60;
        int aa01Mark, aa02Mark, aa03Mark;
        boolean inputError = false, passedAllThree = false;
        boolean distinctionInOne = false;
        Scanner scan = new Scanner(System.in);

        System.out.print("Enter your mark for AA01: ");
        aa01Mark = scan.nextInt();
        System.out.print("Enter your mark for AA02: ");
        aa02Mark = scan.nextInt();
        System.out.print("Enter your mark for AA03: ");
        aa03Mark = scan.nextInt();

    add an appropriate sequence of "if" and "if ... else ..." statements here

    }
}
```

[9 marks]

[Total 20 marks]

PAPER CONTINUED ON NEXT PAGE

2. a) Briefly explain what is meant by the *flow of control* in a Java program. [3 marks]

b) The following program fragment outputs the rows of a units/price table for a certain product:

```
for (int count = 1; count <= MAX_NO_OF_UNITS; count++)
{
  System.out.print("\t\t" + count + "\t\t");
  System.out.println((count * UNIT_PRICE) - DISCOUNT);
}
```

Rewrite this fragment so that it outputs the same rows, but in reverse order. [2 marks]

c) A certain London bus company offers bus tours of the city for £11 per person in the summer, and £5 per person in the winter. In the summer they also offer a discount scheme so that for every four people in a group, one of those people goes free (so that, for example, a group of nine people pays only £77). In the winter they offer a one-off discount of £3 for group sizes of five or more. A program is being written that will produce a table of prices per group size. The program should be able to produce output as in the following example :

```
Enter minimum group size: 3
Enter maximum group size: 6

GROUP           WINTER           SUMMER
3               15               33
4               20               33
5               22               44
6               27               55
```

The main method of the program begins a follows:

```
public static void main(String[] args)
{
  final int SUMMER_PRICE = 11, WINTER_PRICE = 5;
  final int SUMMER_FREE_SIZE = 4, WINTER_DISCOUNT = 3;
  final int WINTER_MIN_DISCOUNT_SIZE = 5;
  Scanner scan = new Scanner(System.in);
  System.out.print("Enter minimum group size: ");
  int minimum = scan.nextInt();
  System.out.print("Enter maximum group size: ");
  int maximum = scan.nextInt();
  System.out.println("\nGROUP\tWINTER\tSUMMER");
```

Use a "for" or "while" loop to complete the main method of the program so that it outputs tables correctly. For this part of the question you may assume that the user enters a whole number between 1 and 5 after the first prompt and a whole number between 6 and 10 for the second prompt, so that it is not necessary to include any input error checking in the program. [6 marks]

d) Using "while" loops, modify the given program fragment in part (c) so that it performs input error checking. The error checking should ensure that the table of prices has a last entry of 50 or less group size, and that the table is no more than 10 lines long (excluding the table headings). For this part of the question you may assume only that the user always enters a (positive or negative) whole number at every prompt. [9 marks]

[Total 20 marks]

PAPER CONTINUED ON NEXT PAGE

3. a) Briefly explain the function of a *parameter* and the function of a *return statement* in a Java method declaration.

[7 marks]

b) A type of fan-assisted oven that can have its temperature set and its fan turned off and on is to be represented as a Java class. The oven must always operate within the following safety constraints: (i) the temperature must never be less than 5 or more than 230 (degrees Celsius), and (ii) if the fan is turned off the temperature must not be set to more than 180. Write a class definition for the oven. Your definition should be such that, when tested with the following program,

```
import java.util.Scanner;
public class OvenTestProg
{
    public static void main(String[] args)
    {
        Scanner scan = new Scanner(System.in);
        Oven anOven = new Oven();
        System.out.println(anOven.toString());
        char selection = 'a';
        while (selection != 'q')
        {
            System.out.print("\nType: ");
            System.out.println("\tf - to turn on fan");
            System.out.println("\to - to turn off fan");
            System.out.println("\tt - to set temperature");
            System.out.println("\tq - to quit\n");
            System.out.print("\tEnter selection: ");
            selection = scan.next().charAt(0);
            switch (selection)
            {
                case 'f':
                    anOven.turnOnFan();
                    System.out.println(anOven.toString());
                    break;
                case 'o':
                    anOven.turnOffFan();
                    System.out.println(anOven.toString());
                    break;
                case 't':
                    System.out.print("\tEnter new temperature: ");
                    int temperature = scan.nextInt();
                    anOven.setTemperature(temperature);
                    System.out.println(anOven.toString());
                    break;
            }
        }
        System.out.println("\tPROGRAM ENDED\n");
    }
}
```

the example input/output on the following page can be reproduced.

P.T.O.

QUESTION 3 CONTINUED ON NEXT PAGE

```
                Status: Temperature 5,  Fan off

        Type:   f - to turn on fan
                o - to turn off fan
                t - to set temperature
                q - to quit

                Enter selection: f
                Status: Temperature 5,  Fan on

        Type:   f - to turn on fan
                o - to turn off fan
                t - to set temperature
                q - to quit

                Enter selection: t
                Enter new temperature: 200
                Status: Temperature 200,  Fan on

        Type:   f - to turn on fan
                o - to turn off fan
                t - to set temperature
                q - to quit

                Enter selection: o
                Status: Temperature 180,  Fan off

        Type:   f - to turn on fan
                o - to turn off fan
                t - to set temperature
                q - to quit

                Enter selection: t
                Enter new temperature: 200
                Status: Temperature 180,  Fan off

        Type:   f - to turn on fan
                o - to turn off fan
                t - to set temperature
                q - to quit

                Enter selection: q
                PROGRAM ENDED
```

(The numbers and letters in bold are user inputs.)

[13 marks]

[Total 20 marks]

PAPER CONTINUED ON NEXT PAGE

4. a) Briefly explain what is meant by *method decomposition* in the context of Java programming. [3 marks]

b) Briefly explain the difference between *primitive data variables* and *object reference variables* in the context of Java programming. [4 marks]

c) A 1 litre can of a certain type of paint can cover 3 square meters of wall. The following program calculates how many cans of paint will be needed to cover a wall of a height and width given by the user:

```
import java.util.Scanner;
public class PaintCalc
{
    public static final int COVERAGE = 3, MAX_METERS = 20;

    public static void main(String[] args)
    {
        int height = 0, width = 0;
        Scanner scan = new Scanner(System.in);

        while (height < 1 || height > MAX_METERS)
        {
            System.out.print("How high is the wall in meters? ");
            height = scan.nextInt();
            if (height < 1 || height > MAX_METERS)
                System.out.print("ERROR! - ");
        }
        while (width < 1 || width > MAX_METERS)
        {
            System.out.print("How wide is the wall in meters? ");
            width = scan.nextInt();
            if (width < 1 || width > MAX_METERS)
                System.out.print("ERROR! - ");
        }
        int area = height * width;
        System.out.print("Number of cans needed: ");
        if (area % COVERAGE == 0)
            System.out.println(area / COVERAGE);
        else
            System.out.println((area / COVERAGE) + 1);
    }
}
```

Rewrite this program using method decomposition, ensuring that the output remains exactly the same. The first two methods of the program should be exactly as follows:

```
public static void main(String[] args)
{
    int height = inputWithPrompt("How high is the wall in meters? ");
    int width = inputWithPrompt("How wide is the wall in meters? ");
    outputCansNeeded(height, width);
}

public static boolean measurementWrong(int aMeasurement)
{
    return (aMeasurement < 1 || aMeasurement > MAX_METERS);
}
```

Your answer should therefore include appropriate definitions for "`inputWithPrompt(...)`" and "`outputCansNeeded(...)`", and should make appropriate use of the method "`measurementWrong(...)`" above. [8 marks]

P.T.O.

QUESTION 4 CONTINUED ON NEXT PAGE

d) Briefly explain why the program given in part (c) cannot be decomposed with a main method as follows:

```
public static void main(String[] args)
{
    int height = 0, width = 0;
    inputMeasurements(height, width);
    outputCansNeeded(height, width);
}
```

[5 marks]

[Total 20 marks]

PAPER CONTINUED ON NEXT PAGE

5. This question is about a program for a mile long running race. It stores the last names of each runner and the time in seconds they took to complete the race. The race rules state that runners who fail to complete the race or who take longer than 20 minutes are recorded as having a completion time of 1200 seconds. The question and program assume the existence of a class "Runner" with the following public methods:

```
public Runner ()
public void setRunnerName (String aName)
public void setRunnerTime (int aNumber)
public String getRunnerName ()
public int getRunnerTime ()
```

The first of these methods is a constructor which sets an initial value of "none" for the runner's name and –1 for the race time (to indicate an "unused" Runner instance). The other methods are standard "set" and "get" methods for the two items of data associated with each Runner.

In the program, the runner information is stored in an array of Runner object instances called "`runnerList`" which is declared in the "main(...)" method. The main menu of the program gives the user two options (apart from quitting the program). The first option is to enter all the race details, and the second option is to display details of who came first, second and third. The first part of the program is as follows:

```
import java.util.Scanner;
public class RunnerProg
{
    public static final int MAX_NO_OF_RUNNERS = 50;
    public static final int MAX_POSS_TIME = 1200;

    public static void main(String[] args)
    {
        Scanner scan = new Scanner(System.in);
        Runner[] runnerList = new Runner[MAX_NO_OF_RUNNERS];
        for (int index = 0; index < runnerList.length; index++)
            runnerList[index] = new Runner();

        char option = 'z';
        while (option != 'q')
        {
            System.out.println("\nEnter:");
            System.out.println("   i - to input race details");
            System.out.println("   w - to display winners");
            System.out.println("   q - to quit the program");
            System.out.print("Type your option and press RETURN: ");
            option = scan.next().charAt(0);

            switch (option)
            {
                case 'i': inputDetails(runnerList);
                          break;

                case 'w': displayWinners(runnerList);
                          break;
            }
        }
    }
```

P.T.O.

QUESTION 5 CONTINUED ON NEXT PAGE

a) After carefully examining the preceding program fragment, write a suitable definition for the method "inputDetails(...)" (which is called from inside the switch statement in the "main(...)" method). You may assume that no two runners have the same last name. Invocation of this method via an input of "i" from the main user menu should cause the program to have input/output such as:

```
Please enter the total number of runners: 10

RUNNER NUMBER 1
Enter runner's last name: Jones
Enter race time for Jones: 456

RUNNER NUMBER 2
Enter runner's last name: Miller
Enter race time for Miller: 459

.............
.............
.............

RUNNER NUMBER 9
Enter runner's last name: Patel
Enter race time for Patel: 652

RUNNER NUMBER 10
Enter runner's last name: Smith
Enter race time for Smith: 359
```

(The names and numbers in bold are user inputs.)

The method you write should ensure that the race time entered for each runner is sensible.

[10 marks]

b) Write a definition for the method "displayWinners(...)" (which is also called from inside the switch statement in the "main(...)" method). For this part of the question you may assume that at least three runners took part in the race, that no two runners have the same last name, and that no two runners finished the race at the same time. Invocation of this method via an input of "w" from the main user menu should cause the program to have input/output such as:

```
First place: Smith
Second place: Jones
Third place: Miller
```

[10 marks]

[Total 20 marks]