SIM UNIVERSITY

ICT131

Examination – July Semester 2016

Introductory Programming and Object Oriented Concepts Using Java

Tuesday, 15 November 2016 **1:00 pm – 3:00 pm**

Time allowed: 2 hours

INSTRUCTIONS TO STUDENTS:

1. This examination contains **FOUR (4)** questions and comprises **EIGHT (8)** printed pages (including cover page and appendix A).
2. You must answer **ALL** questions.
3. This is a Closed Book examination.
4. All answers must be written in the answer book.
5. Appendix A contains some Java API which you might need.

At the end of the examination
Please ensure that you have written your examination number on each answer book used.

Failure to do so will mean that your work cannot be identified.

If you have used more than one answer book, please tie them together with the string provided.

THE UNIVERSITY RESERVES THE RIGHT NOT TO MARK YOUR SCRIPT IF YOU FAIL TO FOLLOW THESE INSTRUCTIONS.

Answer all questions. (Total 100 marks)

Question 1

(a) Figure Q1(a) shows a line that passes through 2 points: (x_1, y_1) and (x_2, y_2).

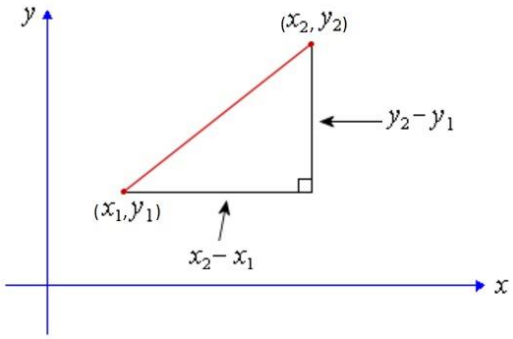

Figure Q1(a)

Assuming that $y_1 \neq y_2$, then the gradient of a line is given by

$$gradient = \frac{(x_2 - x_1)}{(y_2 - y_1)}$$

The intercept of a line is given by

$$y\text{–}intercept = y_1 - x_1(gradient)$$

Use the Java-supplied classes and cite the Java documentation to write a Java program that

- accepts four input arguments from the command line and converts them to whole numbers x_1, y_1, x_2 *and* y_2,
- calculates the *gradient* and y-*intercept* with the given formulae,
- uses `printf` to display the line equation in the form:

$$y = (gradient)x + y\text{–}intercept$$

Each number is displayed with 2 decimal places.

Shown below is an example output when the inputs of x_1, y_1, x_2 and y_2 are `"1","5","2"` and `"1"` respectively:

```
y = -0.25x + 5.25
```

Note: You MUST NOT hardcode the values for x_1, y_1, x_2 and y_2 in the formulae.

(11 marks)

(b) Examine the program in Figure Q1(b) and outline the output of the program.

```
 1 public class Q1b
 2 {
 3    public static void main (String [] args)
 4    {
 5       int [] x = {0, 1, 2};
 6       int y = 1;
 7       int z = incr(y, x[1]);
 8       x[y] = z;
 9       System.out.println("a) y = " + y);
10       System.out.println("a) z = " + z);
11       print(x);
12       incr(x);
13       print(x);
14    }
15
16    public static int incr (int i, int j)
17    {
18       int k = ++i + j;
19       System.out.println("b) i = " + i);
20       System.out.println("b) j = " + j);
21       System.out.println("b) k = " + k);
22       return k;
23    }
24
25    public static void incr (int [] x)
26    {
27       int j = 5;
28       for (int i = 0; i < x.length; i++)
29       {
30          x[i] = j--;
31          System.out.println("c) j = " + j);
32       }
33    }
34
35    public static void print (int [] x)
36    {
37       for (int i = 0; i < x.length; i++)
38       {
39          System.out.println("d) x[" + i + "] = "
40                                    + x[i]);
41       }
42    }
43 }
```

Figure Q1(b)

(14 marks)

Question 2

Outline the control structures of structured programming. For each of the following scenarios, scenario A, B and C, do the following:-

- **without repeating your choice for the other scenarios, specify with reasons**, the most appropriate control structure(s) to use from the following list:

  `if` `if-else` `switch` `for` `while` `do-while`

  Note that if a scenario requires more than one control structures, these control structures must not be used again in the other scenarios.

- **Develop a program segment** to show how the chosen structure(s) is/are applied to the scenario.

- **If there is any need to read user input,** assume that a `Scanner` object has been created and is referenced with the variable `sc`.

(a) Scenario A:

Write a program segment that allows a user to enter one or more whole numbers, each number representing an exam score. The user terminates the data entry by entering `-1` for an exam score. The program segment then displays the average exam score.

(9 marks)

(b) Scenario B:

Write a program segment to display the number of days in a given month. For the user input "Jan", "Mar", "May", "Jul", "Aug", "Oct" or "Dec", the program segment displays `31 days`. For the input "Apr", "Jun", "Sep" or "Nov", the program segment displays `30 days`. For the input "Feb", the program segment displays `28 or 29 days`. For any other input, the program segment displays `invalid month`. Assume that the input value is already stored in the variable `monthName`.

(8 marks)

(c) Scenario C:

A valid test score is a decimal number between 1 and 100, inclusive of 1 and 100. Assume that a test score has been read into the variable `score`, write a program segment to validate the test score. If the test score is not within that range, display `The test score is invalid` and request for another input until the input for the test score is valid.

(8 marks)

Question 3

Develop a Java program using the control structures of structured programming on an array data structure for an application that manages a contact list of size 30. The contact list is implemented using

- one array `name` to record the names of people or companies and
- one array `telephone` to record their telephone numbers.

Write a complete Java program with the following methods:

(a) A method `addContact` to add a contact into the list.

- The method accepts the two arrays and an integer `count` which is the number of data already in the arrays.
- If the count of data is already the maximum number of data that can be stored in each array, the method prints the error message: `No more contact can be added!` Otherwise, the method prompts the user for a name and a telephone number, records them in the appropriate arrays and increments `count` as one more data is recorded.
- This method returns the value of `count`.

(8 marks)

(b) A method `updateContact` to update a telephone number from the list.

- The method accepts the two arrays, an integer `count` which is the number of data already recorded in the arrays, and a name whose telephone number is to be changed.
- If the provided name is a name of an existing contact, the method prompts the user for a telephone number, updates the corresponding telephone number and returns true. If the update is unsuccessful, then method returns false.

(8 marks)

(c) A method `displayMenu` that displays the following menu:

```
Menu
1. Add New Contact
2. Update Existing Contact
0. Exit
```

(2 marks)

(d) A method `main` which does the following:

- declare and create two arrays `name` and `telephone` that can record a maximum of 30 names and 30 of telephone numbers respectively
- declare and initialise `count` which is the number of data already recorded in the arrays to zero,
- repeatedly display a menu to allow the user to select an option and call the appropriate method to perform the task according to his selection until the user chooses to exit the program. For option 2, prompt for the name of the contact to update the contact, and print the outcome of the update.

(7 marks)

Question 4

Friends@Sushi, a sushi bar, is tracking its sales to fine-tune its pricing. You are tasked to write programs to develop a prototype for Friends@Sushi using object-oriented programming approach. Details of a food item include the following: name e.g., Ikura, price e.g., $3.95, orders to date for this year e.g., 183 orders, and whether it is also part of a set meal e.g., true if it can be purchased as part of a set meal and false if otherwise.

Implement a Java class that models a food item. This class should have the following:

(a) Suitable instance variables for each piece of information given above.

(4 marks)

(b) A constructor with the appropriate actions on the instance variables. The default value for orders to date is 0 and the default value is the food item cannot be purchased as part of a set meal.

(5 marks)

(c) The get and set methods for the instance variables for price, and only the get method for whether a food item is also part of a set meal.

(3 marks)

(d) A `changePurchasedAsPartOfMeal ()` method which changes whether the food item can be purchased as part of a set meal. If currently it cannot be purchased as part of a set meal, then change it to being able to be purchased as part of a set meal, and vice versa.

(2 marks)

(e) An `order()` method that accepts the order quantity to add to the orders to date. If the order quantity is negative, do not add to the orders to date but return false. Otherwise, add and return true.

(4 marks)

(f) A `toString()` method that returns a string containing the values of the instance variables with descriptions.

(2 marks)

(g) Write a test program to exercise the class written in parts (a) to (f). The test program should perform the following:

- Create 2 food items with data as shown in the Figure Q4 and **hold the objects in a single suitable data structure.**

Name	**Price**	**Orders to Date**	**Also Part Of Set**
Ikura	**$4.95**	**0**	**No**
Aburi	**$3.95**	**0**	**No**

Figure Q4

- Apply the `order()` method to make 3 orders of the first food item.
- Apply the `changePartOfMeal()` method on the second food item to make it part of a set meal.

(5 marks)

Appendix A

Class Double

Modifier and Type	Method and Description
`Double`	`doubleValue()` Returns the `double` value of this `Double` object.
`static double`	`parseDouble(String s)` Returns a new `double` initialized to the value represented by the specified `String`, as performed by the `valueOf` method of class `Double`.

Class Integer

Modifier and Type	Method and Description
`Int`	`intValue()` Returns the value of this `Integer` as an `int`.
`static int`	`parseInt(String s)` Parses the string argument as a signed decimal integer.
`static Integer`	`valueOf(int i)` Returns an `Integer` instance representing the specified `int` value.

Class Scanner

Modifier and Type	Method and Description
`Boolean`	`hasNextBoolean()` Returns true if the next token in this scanner's input can be interpreted as a boolean value using a case insensitive pattern created from the string "true|false".
`Boolean`	`hasNextDouble()` Returns true if the next token in this scanner's input can be interpreted as a double value using the `nextDouble()` method.
`Boolean`	`hasNextInt()` Returns true if the next token in this scanner's input can be interpreted as an int value in the default radix using the `nextInt()` method.
`String`	`next()` Finds and returns the next complete token from this scanner.
`boolean`	`nextBoolean()` Scans the next token of the input into a boolean value and returns that value.

Double	nextDouble() Scans the next token of the input as adouble.
Int	nextInt() Scans the next token of the input as an int.
String	nextLine() Advances this scanner past the current line and returns the input that was skipped.

Class String

Modifier and Type	Method and Description
Char	charAt(int index) Returns the char value at the specified index.
Int	compareTo(String anotherString) Compares two strings lexicographically.
Int	compareToIgnoreCase(String str) Compares two strings lexicographically, ignoring case differences.
String	concat(String str) Concatenates the specified string to the end of this string.
Int	indexOf(String str) Returns the index within this string of the first occurrence of the specified substring.
boolean	equals(Object anObject) Compares this string to the specified object.
boolean	equalsIgnoreCase (String anotherString) Compares this String to another String, ignoring case considerations.
Int	indexOf(String str, int fromIndex) Returns the index within this string of the first occurrence of the specified substring, starting at the specified index.
Int	lastIndexOf(String str) Returns the index within this string of the last occurrence of the specified substring
Int	length() Returns the length of this string.

----- END OF PAPER -----