NANYANG TECHNOLOGICAL UNIVERSITY

SEMESTER 2 EXAMINATION 2014-2015

CE2002/CZ2002 – OBJECT ORIENTED DESIGN & PROGRAMMING

Apr/May 2015 Time Allowed: 2 hours

INSTRUCTIONS

1. This paper contains 4 questions and comprises 8 pages.
2. Answer **ALL** questions.
3. This is a closed-book examination.
4. All questions carry equal marks.
5. APPENDIX A shows a class hierarchy referenced by Question 1 and Question 2.
6. APPENDIX B shows the Class Diagram referenced by Question 3.
7. APPENDIX C shows the Sequence Diagram referenced by Question 3.

---

1. (a) The Java statement "`ClassX object1 = new ClassX();`" creates an object. Explain the **TWO** steps involved in this statement for creating the object and what happens in the memory for each step. If there is another statement "`ClassX object2 = object1;`" following the previous statement, explain what happens in the memory.

(5 marks)

(b) Explain why Java **DOES NOT** support multiple inheritance. Elaborate Java's alternative to multiple inheritance, with an example of Java code.

(4 marks)

Note: Question No. 1 continues on Page 2

(c) List **THREE** types of methods in Java that **CANNOT** have dynamic binding, and explain the reason for each type of methods.

(6 marks)

(d) Study the class hierarchy diagram in Appendix A (page 6).

Explain the outcome, the class's method called or used (if any) and the type of casting for **each line** of the following codes during **compile-time** and **runtime**.

(i)
```
ClassA a = new ClassE();
a.print(100, 100);
a.print("hi");
ClassB b = (ClassB)a;
b.print("hi");
```

(5 marks)

(ii)
```
InterfaceA a = new ClassF();
ClassC c = (ClassC)a;
ClassG g = (ClassG)a;
c.print(100);
g.print(100);
```

(5 marks)

2. Study the class hierarchy diagram in Appendix A (page 6).

(a) Write the code for the following CLASSES:

- `ClassA` whose `print(x: int, y: int)` method prints out each of the two integer parameters.
- `ClassB` whose `print(s: String)` method prints out the entire string parameter, and `print(x: int, y: int)` method prints out the sum of the two integer parameters.
- `ClassD` whose `print(s: String)` method prints out the first character of the string parameter, and `print(x: int, y: int)` method prints out the subtraction of the two integer parameters.
- `ClassE` whose `print(x: int, y: int)` method prints out the average of the two integer parameters.

(10 marks)

Note: Question No. 2 continues on Page 3

(b) Write an application class `ClassApp` to have **THREE** `printInt` methods. All the three methods take two integers as parameters. The first `printInt` method prints out the sum of the two integers. The second `printInt` method prints out the subtraction of the two integers. The third `printInt` method prints out the average of the two integers. Note that the `printInt` methods **MUST** use `print(x: int, y: int)` method in `ClassB`, `ClassD` and `ClassE` for the printing.

Also write the `main` function in the application class to demonstrate **static binding** of the `printInt` methods to print out the sum, subtraction and average of two integers, respectively.

Note: all the above `printInt` methods return `void`.

(5 marks)

(c) To demonstrate **dynamic binding** of the `printInt` method, write another version of the application class `ClassApp` to have only one `printInt` method in the application class, regardless of which subclass of `ClassA` passed as argument of the `printInt` method.

(5 marks)

(d) If the `print(x: int, y: int)` methods in `ClassA`, `ClassB`, `ClassD` and `ClassE` are all implemented as **static** methods, explain whether the output of the **TWO** versions of the application class `ClassApp` in (b) and (c) will be affected. If the output is affected, elaborate what the changed output should be.

(5 marks)

3. (a) The UML **Class Diagram** in Appendix B (page 7) shows the relationships between the `IComparable` interface, the `Sale` class and the `SaleItem` class. Additional relevant details are provided below.

The `items` array in the `Sale` class stores `SaleItem` objects and the `numOfItem` stores the number of `SaleItem` objects currently in the `items` array. The `Sale` class has a constructor which will initialize the attributes `numOfItem` and the `items` array, and a destructor which will clean up the relevant attributes. The `addItem` function will create

Note: Question No. 3 continues on Page 4

ATTENTION: The Singapore Copyright Act applies to the use of this document. Nanyang Technological University Library

CE2002/CZ2002

the `SaleItem` object and add to the `items` array if `numOfItem` does not exceed the size of the `items` array. It returns `true` if the `SaleItem` object is added and returns `false` otherwise. The `getTotal` function will add the prices of all `SaleItem` objects and return the total value. The `operator==` function is overloaded to compare if two `Sale` objects have the same total values. It returns `true` if the total values are equal and returns `false` otherwise.

(i) Write the C++ code for the `IComparable` interface.

(3 marks)

(ii) Write the C++ code for the `Sale` class. You should include all necessary error checking in your code.

(12 marks)

(b) The UML **Sequence Diagram** in Appendix C (page 8) shows the objects' interaction of a scenario flow in a particular application. Using the details depicted in the diagram, write the preliminary JAVA code and methods for the class `TestA`. You may make appropriate assumptions on the method parameters, return types and return value(s) if they are not stated in the diagram.

(10 marks)

4. Study the following description of a Travel Management System (TMS):

TMS is a web-based system that maintains a centralized repository of all related information. TMS allows users to easily access the relevant information and plan their travel arrangements.

To access TMS, users must register with the system by providing their user identity, password, name, email address and passport identification. After registration, users can access TMS to create their travel package by selecting the itinerary of the travel, the hotel accommodations and the required transportations.

For each itinerary, information such as arrival and departure dates, transportation details, accommodation details and pick-up points is to be provided.

Note: Question No. 4 continues on Page 5

The transportation details provide the information on the ticket fare, date and time of boarding and arrival, and the airline used. The accommodation details provide the information on the hotel, the room type, room rate, and check-in, check-out date and time.

After a user has confirmed the travel package, the total price of the package and the duration of travel (start and end date) are displayed together with the user's relevant details.

(a) You are being tasked to identify the entity classes needed to build the TMS based on the description above.

Show your design in a **Class Diagram**. Your **Class Diagram** should show clearly the relationship between classes, relevant attributes (at least TWO) and, multiplicities, association names, if any. You need not show the class methods.

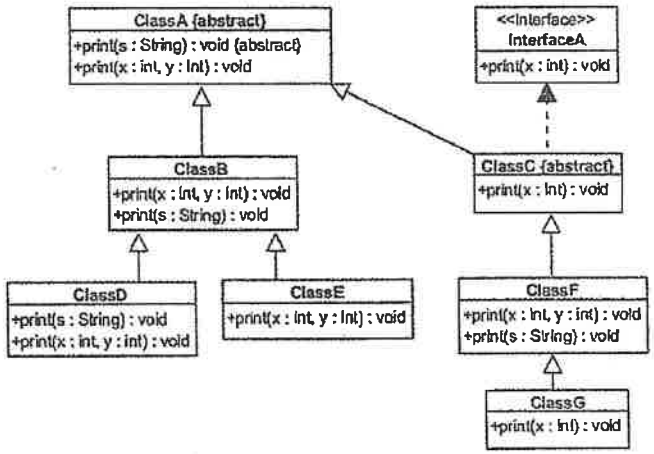
(15 marks)

(b) To calculate the total price of the package, the system will aggregate all the costs of the selected services and increase it by a certain percentage. In future, other services, like local tours, will be added as part of the travel package.

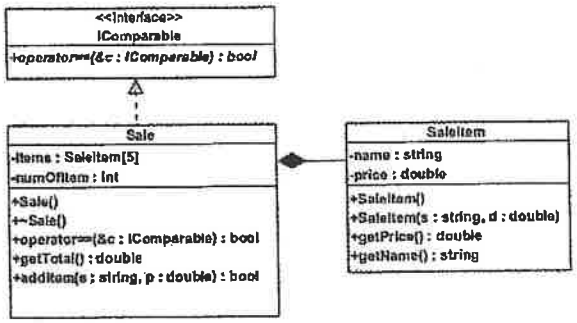
By using the design principles you have learned, suggest how you would improve the design of the **Class Diagram** in **Q4(a)** to ensure extensibility and maintainability of TMS.

(10 marks)

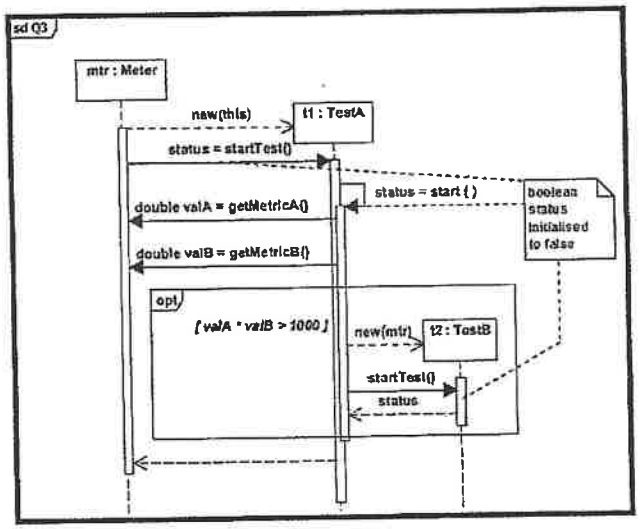
APPENDIX A : Class Hierarchy



APPENDIX B : Class Diagram



APPENDIX C : Sequence Diagram



END OF PAPER