NATIONAL UNIVERSITY OF SINGAPORE

**SCHOOL OF COMPUTING**

**SEMESTER II (1999-2000)**
**EXAMINATION FOR**

# CS2103/CS2103S SOFTWARE DESIGN METHODOLOGY

25 April 2000 Time Allowed: 2 Hours

---

**INSTRUCTIONS TO CANDIDATES**

1. This examination paper consists of **FIVE (5)** questions and comprises **NINE** (9) printed pages.
2. Answer **all** questions.
3. Make sure that your answers are succinct and to the point.
4. Write your answers in the **blank spaces** in this answer book only.
5. Write your answers in ink. You may use pencil for drawing diagrams only.
6. This is an **OPEN BOOK** examination.
7. Please fill in your **Matriculation Number** below. Also write your matriculation number on the top right hand corner of every page.

**Matriculation Number:**

| | | | | | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|---|---|
| | | -- | | | | | -- | | -- | | |

---

| For Examiner's Use Only | |
|---|---|
| | Marks |
| Question 1 | |
| Question 2 | |
| Question 3 | |
| Question 4 | |
| Question 5 | |
| TOTAL: 50 marks | |

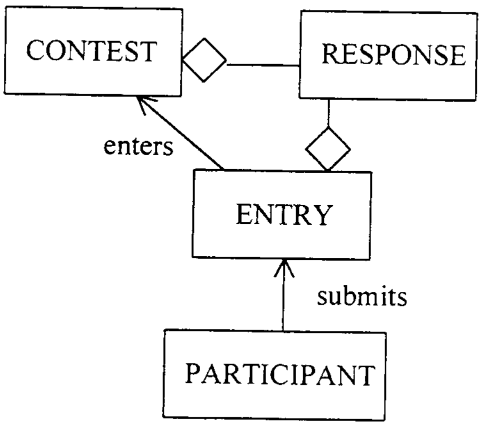
**Question 1 [OOA] (12 marks)**

a. (6 marks) A school has many students and offers many subjects. Each student must take at least 6 but at most 9 subjects, and each subject has a maximum enrolment of 100 students. Certain subject combinations are not allowed: for example, students who take Physics may not take Literature and Geography.

Design a class diagram based on UML notation that allows the above specification to be adequately captured.

b. Ah Beng took CS2103 last semester. Now, he is working on an industrial project. He has been tasked to build a system to administer online contests. There are multiple contests, each has a list of correct responses. A participant may submit to multiple contests but not more than one entry per contest. Each entry contains exactly one guess response.

After digging through his CS2103 notes, Ah Beng came up with the following model:



(i) (2 marks) Complete the above diagram by adding multiplicities.

(ii) (2 marks) Ah Beng's boss suggests that ENTRY is redundant. He suggests a simpler model in which PARTICIPANT is directly connected to CONTEST and RESPONSE. How should Ah Beng react to this suggestion?

(iii) (2 marks) Ah Beng's boss further adds the requirement that each contest has a prize, and if several participants guess a correct response, they share the prize. Expand the above class diagram to capture this information.

**Question 2 [OOD, Implementation] (14 marks)**

a. Examine the following 2 cases of defining a new class via **inheritance**.

For each case:
i. (2 marks each) identify one design issue arising from the form of inheritance
ii. (2 marks each) suggest an alternative design that will resolve the issue.

Case 1

```
class Vector{
        :
   public int size()
   // returns the number of items in the vector

   public void addElement(Object item)
   // adds an item to the end of the vector

   public void removeElementAt(int i)
   // removes the ith item from the vector
        :
}

class Stack extends Vector{
        :
   public Object push(Object item)
   // calls addElement to push an item onto the stack

   public Object pop()
   // calls removeElementAt(size()-1) to pop an item
        :
}
```

Case 2

```
class Employee{
// Invariant: k = (this.setSalary(k)).getSalary()

   protected int salary;
               :
   public int getSalary()
   // returns salary

   public Employee setSalary(int k)
   // sets salary to k, and returns this
               :
}

class Manager extends Employee{
   private int allowance;
         :
   public int getSalary()
   // returns salary+allowance
         :
}
```

b. (6 marks) A job is defined as follows. Each job has a status which is either *pending*, *assigned*, *ongoing* or *completed*. Associated with each status is an action `update()` which is triggered when the job is set to the respective status.

Using **polymorphism**, give a detailed design (PDL notation) of the `Job` class in terms of the `setStatus` method, which sets the job to a given status, and perform the corresponding `update()` action. (*Hint*: recall the Shape class discussed in lecture.)

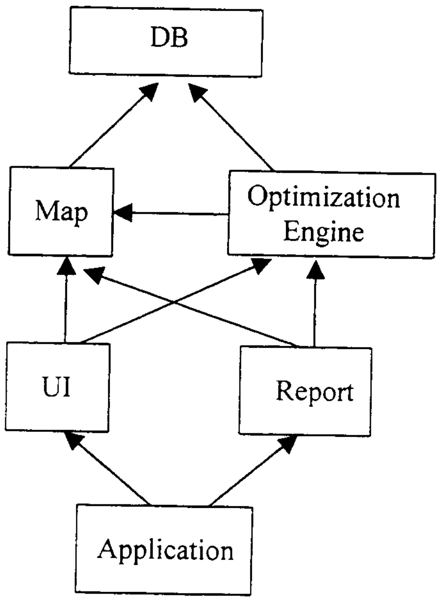
**Question 3 [Models, Exception Handling] (10 marks)**

Ah Hwei has also taken CS2103. She would like to implement a new ADT (abstract data type) called the **Map**. Complete the following specification by adding appropriate pre-conditions, post-conditions, invariants, and error-handling mechanism (i.e. `throws`). You may assume there is no illegal or null pointer arguments.

```
class Map {
// A Map is a set of key-value pairs, where the key and value are
// strings. Each key can occur at most once in the map.

   private Capacity;

   public Map(int N)
   // Description: Constructs an empty map with a capacity of N.

   public int size()
   // Description: Returns the number of pairs in the map.
   // Pre-condition:  NIL.
   // Post-condition: NIL.

   // Invariant:  ______________________________________________

   public boolean isMember(String k)
   // Description: Returns true iff k is a key in the map.
   // Pre-condition:  NIL.
   // Post-condition: NIL.
   // Invariant:  NIL.

   public void add(String k,String v) throws ____________________
   // Description:  Inserts the pair (k,v) into the map.

   // Pre-condition:  __________________________________________

                ____________________________________________________

   // Post-condition:  _________________________________________
   // Invariant: NIL.

   public String lookup(String k) throws _______________________
   // Description: Returns the value associated with k.

   // Pre-condition:____________________________________________
   // Post-condition: NIL.
   // Invariant: NIL.

   public void delete(String k) throws _________________________
   // Description:  Removes the pair (k,v) from the map.

   // Pre-condition:  __________________________________________

   // Post-condition:___________________________________________
   // Invariant:  NIL.
}
```

**Question 4 [Integration] (8 marks)**

Consider the following client-object relation diagram.



a. (2 marks) List all different bottom-up integration plans (e.g. <Map,DB,...>).

b. (2 marks) In top-down integration testing, what set(s) of modules (if any) can be tested concurrently?

c. (2 marks) Suppose the implementation of the **Map** changes. What modules (if any) might have to be changed?

d. (2 marks) Suppose subsequently a new module **X** needs to be added to the system. Explain briefly what you must do to integrate this module into the system.

**Question 5 [Testing] (6 marks)**

a. (3 marks) Boundary value analysis (BVA) has been associated with Black-box Testing. Discuss briefly how BVA might also be used in White-box Testing, and give an example to illustrate your idea.

b. (3 marks) Suppose you are given a system represented as a state diagram (such as the elevator controller). Briefly discuss how you may design test cases for the system.

END OF PAPER