MidTerm 28 March 2018, questions and answers

Software Design (City University of Hong Kong)

香港城市大學
City University
of Hong Kong

Department of Computer Science
City University of Hong Kong

Sample Mid-Term Assessment (Solution)

CS3342 Software Design

Time Allowed: 90 Minutes

Student Name: ______________________________

Student ID: ________________________________

Tutorial Class: ______________________________

Instructions:

1. This is a ***CLOSED-BOOK*** examination.
2. Answer ***ALL*** questions.
3. Write down ***ALL*** of your answers in the spaces provided.
4. *Cheating will result in immediate disqualification and may be subject to further disciplinary actions.*
5. The maximum mark is 60.

Q1 **10**	Q2a **5**	Q2b **5**	Q2c **10**	Q2d **10**	Q3 **20**	TOTAL **60**

Question 1 – Software Engineering in General (10 Marks):

1. Software is a product and can be manufactured using the same technologies used for other engineering artifacts.
   a. True
   b. False
2. Software deteriorates rather than wears out because
   a. Software suffers from exposure to hostile environments
   b. Defects are more likely to arise after software has been used often
   c. Multiple change requests introduce errors in component interactions
   d. Software spare parts become harder to order
3. Most software continues to be custom built because
   a. Component reuse is common in the software world.
   b. Reusable components are too expensive to use.
   c. Software is easier to build without using someone else components.
   d. Off-the-shelf software components are unavailable in many application domains.
4. Which of the following is not generally considered a player in the software process?
   a. customers
   b. end-users
   c. project managers
   d. sales people
5. The linear sequential model of software development is:
   a. A reasonable approach when requirements are well defined.
   b. A good approach when a working program is required quickly.
   c. The best approach to use for projects with large development teams.
   d. An old-fashioned model that cannot be used in a modern context.
6. The linear sequential model of software development is also known as the:
   a. Fountain model
   b. Spiral model
   c. Waterfall model
   d. None of the above
7. The work products produced during requirement elicitation will vary depending on the
   a. size of the budget
   b. size of the product being built
   c. software process being used
   d. stakeholder's needs
8. It is relatively common for different customers to propose conflicting requirements, each arguing that his or her version is the right one.
   a. True
   b. False
9. Reusable software components must be
   a. catalogued for easy reference.
   b. standardized for easy application.
   c. validated for easy integration.
   d. all of the above.
10. An advantage of scenarios is that they
    a. demonstrate the behavior of the product in a way that is comprehensible to the user.
    b. can be understood by the users and therefore the users play active roles throughout the requirement gathering process.
    c. play an important role in system analysis.
    d. may be depicted in a number of ways e.g. lists of actions, storyboards.
    e. All of the above.

Please complete the following table; please provide a single best answer from choices.

MC Question	Your Best Answer (1 Mark each)
1	**b**
2	**c**
3	**d**
4	**d**
5	**a**
6	**c**
7	**b**
8	**a**
9	**d**
10	**e**

Question 2

Please study the following code listings for class **Bank** and class **Client.**

```
import java.util.ArrayList;

public class Bank
{
    private double _interest;
    private String _name;
    private ArrayList<String> _records = new ArrayList<>();

    public Bank(String name, double interest) {
        _name = name;
        _interest = interest;
    }

    public void setInterest(double interest) {
        _interest = interest;
    }
    public double getInterest() {
        return _interest;
    }
public boolean applyLoan(Client client, double amount) {
        String record = new String ("Client " + client.getName() +
                                    " applys loan: " + amount);
        records.add(record);
        return true;
    }
}

public class Client
{
    private String _name;
    private double _acceptedInterest;
    private final int RETRY_TIMES = 1000;

    public Client(String name, double acptInterest) {
        _name = name;
        _acceptedInterest = acptInterest;
    }
    public String getName() {
        return _name;
    }
    public void tryApply(Bank bank, double amount) throws Exception {
        boolean success = false;

        for (int i = 0; i < RETRY_TIMES; i++) {
            double currentInterest = bank.getInterest();
            if (currentInterest > _acceptedInterest) {
                System.out.println("Current interest is too high. Retry after 5
seconds.");
                Thread.sleep(5000); // wait 5000ms, i.e. 5 seconds
            } else {
                success = true;
                break;
            }
        }

        if (success) {
            bank.applyLoan(this, amount);
        }
    }
}
```

Question 2a - Roles of Variables (5 Marks)

Table 1. Roles for Variables in Software Programs

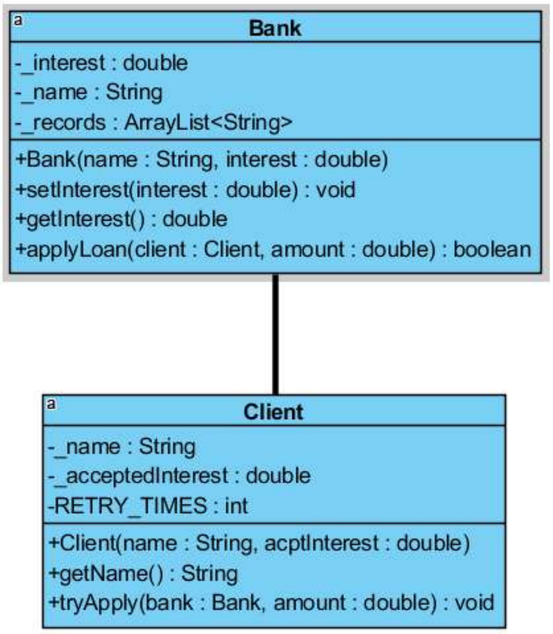
Role	**Description**
Constant/ Fixed value	A variable which is initialized without any calculation and whose value does not change thereafter.
Stepper	A variable stepping through values that can be predicted as soon as the succession starts.
Most-recent holder	A variable holding the latest value encountered in going through a succession of values.
Gatherer	A variable accumulating the effect of individual values in going through a succession of values.
Transformation	A variable that always gets its new value from the same calculation from value(s) of other variable(s).
One-way flag	A two-valued variable that cannot get its initial value once its value has been changed.
Temporary	A variable holding some value for a very short time only.
Organizer	A data structure, which is only used for rearranging its data and object elements after initialization.

Based on the code listings of class **AccountController**, correctly classify the role of each variable in the table below. You may use Table 1 for your reference. (5 Marks)

Variable	**Role of Variable**
RETRY_TIMES	Constant/Fixed value
acptInterest	Constant/Fixed value
i	Stepper
currentInterest	Most-recent holder
success	One-way flag

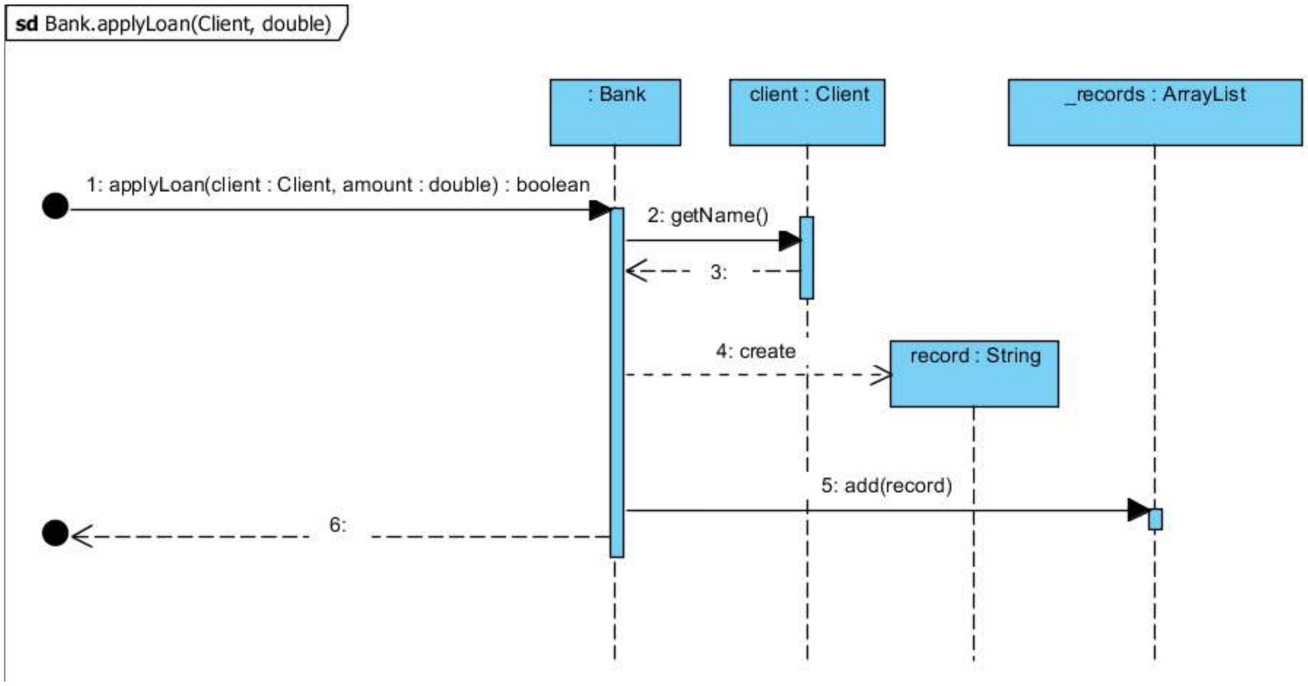
Question 2b – Class Diagram (5 Marks)

Based on the code listings of class **Bank** and class **Client**, draw a complete class diagram. Correctly show all the attributes.


a
Bank
-_interest : double
-_name : String
-_records : ArrayList<String>
+Bank(name : String, interest : double)
+setInterest(interest : double) : void
+getInterest() : double
+applyLoan(client : Client, amount : double) : boolean
a
Client
-_name : String
-_acceptedInterest : double
-RETRY_TIMES : int
+Client(name : String, acptInterest : double)
+getName() : String
+tryApply(bank : Bank, amount : double) : void

Question 2c – Sequence Diagram (10 Marks)

Based on the code listings of function **Bank.applyLoan**, complete the following Sequence Diagram with full details.



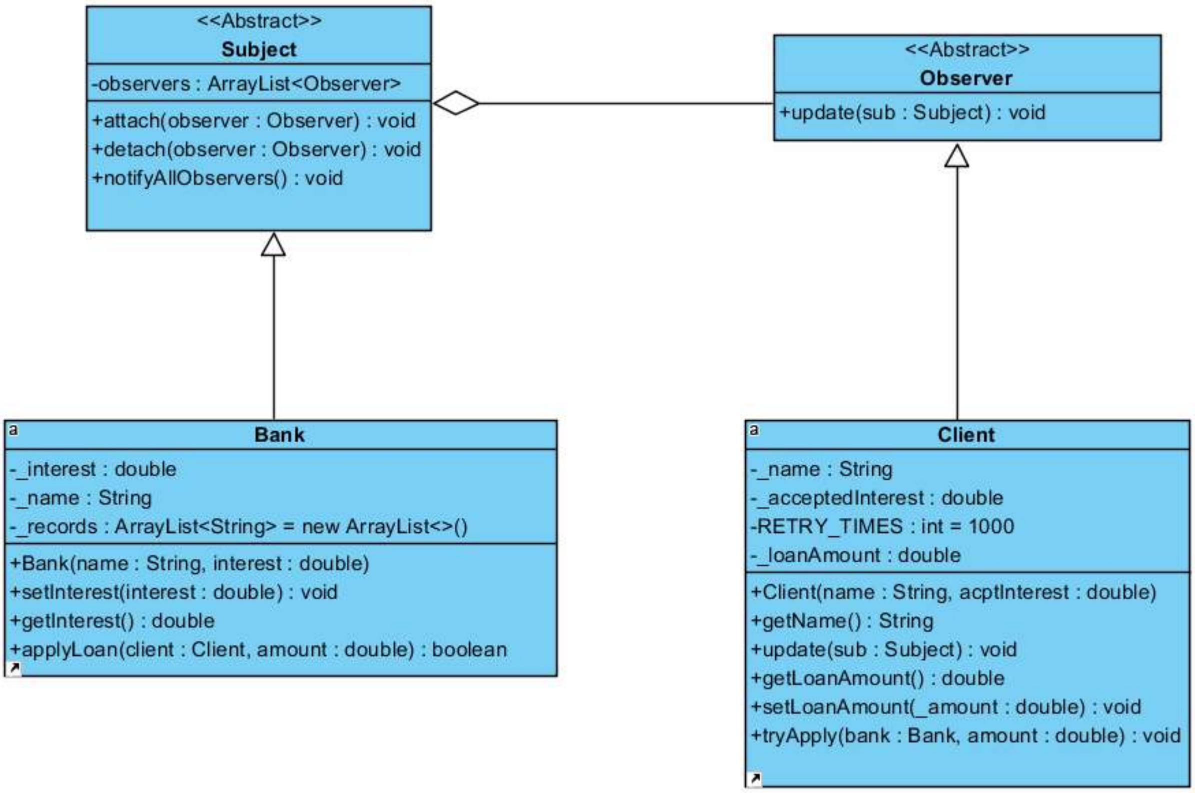
Question 2d – Design Principles and Patterns (10 Marks)

The current design of the bank system uses an active approach. All the clients keep checking the interest rate by requesting the Bank server every 5 seconds, which is inefficient.

Now we need to refine the bank system to make the clients could be informed when the interest is changed. How do you modify the design accordingly?

Which design pattern do you think it is most suitable? ______Observer Pattern_______ (**2** Marks)

Like your solution in 2b, draw a modified/extended class diagram in here. (Showing only affected parts shall be sufficient to illustrate the concept) (**8** Marks)



<<Abstract>>
Subject
-observers : ArrayList<Observer>
+attach(observer : Observer) : void
+detach(observer : Observer) : void
+notifyAllObservers() : void
<<Abstract>>
Observer
+update(sub : Subject) : void
Bank
-_interest : double
-_name : String
-_records : ArrayList<String> = new ArrayList<>()
+Bank(name : String, interest : double)
+setInterest(interest : double) : void
+getInterest() : double
+applyLoan(client : Client, amount : double) : boolean
Client
-_name : String
-_acceptedInterest : double
-RETRY_TIMES : int = 1000
-_loanAmount : double
+Client(name : String, acptInterest : double)
+getName() : String
+update(sub : Subject) : void
+getLoanAmount() : double
+setLoanAmount(_amount : double) : void
+tryApply(bank : Bank, amount : double) : void

Question 3 – Software Requirements (20 Marks)

Case Study: ***English Learning System***

The goal of the application is to teach both local and non-local children to recognize simple English words and speak in English. The idea of teaching method is by means of storytelling.

Around 100 stories have been prepared in the form of flash animation files, and each short story has been ranked in one of three difficulty levels, namely basic, intermediate, and advanced. All types of users, including administrators, need to login to the website before they operate on the English Learning System.

Before the start of a study term, Tom the system administrator of the English Learning System needs to prepare the Weekly Programs. For each of the difficulty levels, Tom will assign a story every week. It is the responsibility of the child's parent to register a child to the appropriate difficulty level according to their child's English proficiency.

During the semester, the child can watch the story of the week. Then the system will allocate the pre-assigned exercises of the story to each child who has watched the story. These exercises can be in any types, such as true-or-false, multiple choices, fill-in-the-blanks or game-based questions. The system will allow the child to complete the exercises and then record the exercise scores, and then notifies their parent about the scores by sending a message.

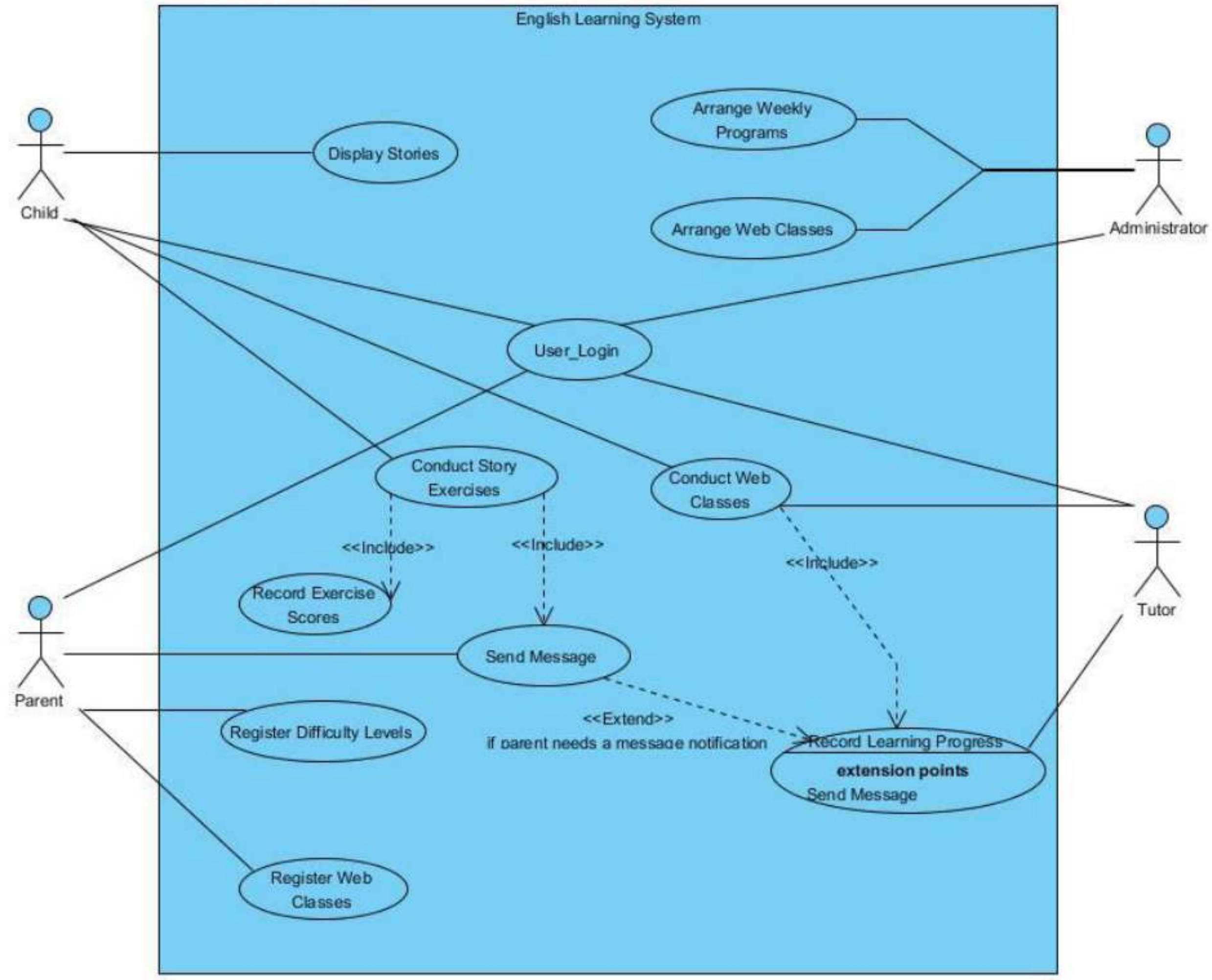
The child may take web-classes, which will be held every week, and their parent can register a web-class for their child. The web-classes are Tutor-led online trainings, in which tutors may verbally answer child's questions, and guide them to improve their oral skills in English, and then the Tutor will document the child's learning progress. If parents need a message notification about their child's learning progress of the taken web classes, then the system will also send a message to their parent. Tom is responsible to arrange the web-classes as well.

Based on the description of the English Learning above, the following shows the use cases and actors in the system.

Use Case	**Actor(s)**
Display Stories	**Child**
User Login	**Child/Parent/Administrator/Tutor**
Arrange Weekly Program	**Administrator**
Arrange Web Classes	**Administrator**
Conduct Story Exercises	**Child**
Record Exercise Score	**Child**
Conduct Web Classes	**Child/Tutor**
Register Web Classes	**Parent**
Register Difficulty Levels	**Parent**
Send Message (for Scores, and Progress Report)	**Parent**
Record Learning Result	**Tutor**

Based on all the given information above:

Draw a complete **<u>Use Case Diagram</u>** of the English Learning System, you must utilize **<<include>>, <<extend>>** with correct arrows when possible, clearly indicating conditions for extension points if any. **(20 marks)**



-- END --