

Systems Analysis and Design

Instructor : Huang, Chuen-Min

Teamwork2 ver.1

Group 8

ID	Name
B10423023	Ryan
A10523048	Tony
B10423022	William
B10523006	Peggy
B10523013	Yvonne
B10523017	Peggy
B10523028	James
B10523037	Yee
B10523056	Sandy

Date 2018/5/29

1. Please explain the Law of Demeter (LoD) by using of your project.

(1) to itself

```
4 String deadline;
5 String[] content = new String[500];
6 static double amount;
7 static double balance;
8 public void validation(String Deadline, String[] Content, double Amount){
9     /*illegal words validate*/
10    /*If there are illegal words inside*/
11    /*set it a mark*/
12    deadline = Deadline;
13    content = Content;
14    amount = Amount;
15    account acc = new account();
16    getB(acc);
17    mark ma = new mark();
18    GUI msg = new GUI();
19    while (validatebalance()){
20        msg.displayMsg("##insufficient balance.##");
21        System.out.printf("amount:");
22        Amount = keyboard.nextDouble();
23        amount = Amount;
24    }
25    if(validate10000()){
26        msg.displayMsg("##amount exceed NT$10000.##");
27        ma.setMark();
28    }
29    if(ma.getResult()){
30        staff sta = new staff();
```

```
public static boolean validatebalance() {
    if(balance >= amount){
        return false;
    }else{
        return true;
    }
}

public static boolean validate10000(){
    if(amount > 10000){
        return true;
    }else{
        return false;
    }
}
```

(2) to objects contained in attributes of itself or a superclass

```
public class account {  
    double balance = 20000;  
    /*Suppose there are 20,000 balances in the account*/  
    public void account(){  
    }  
    public void added(double add){  
        balance = balance + add;  
    }  
    public void minus(double m){  
        balance = balance - m;  
    }  
    public double getBalance(){  
        return balance;  
    }  
}
```

(3) to an object that is passed as a parameter to the method

```
public void getB (account acc) {  
    balance=acc.getBalance();  
}
```

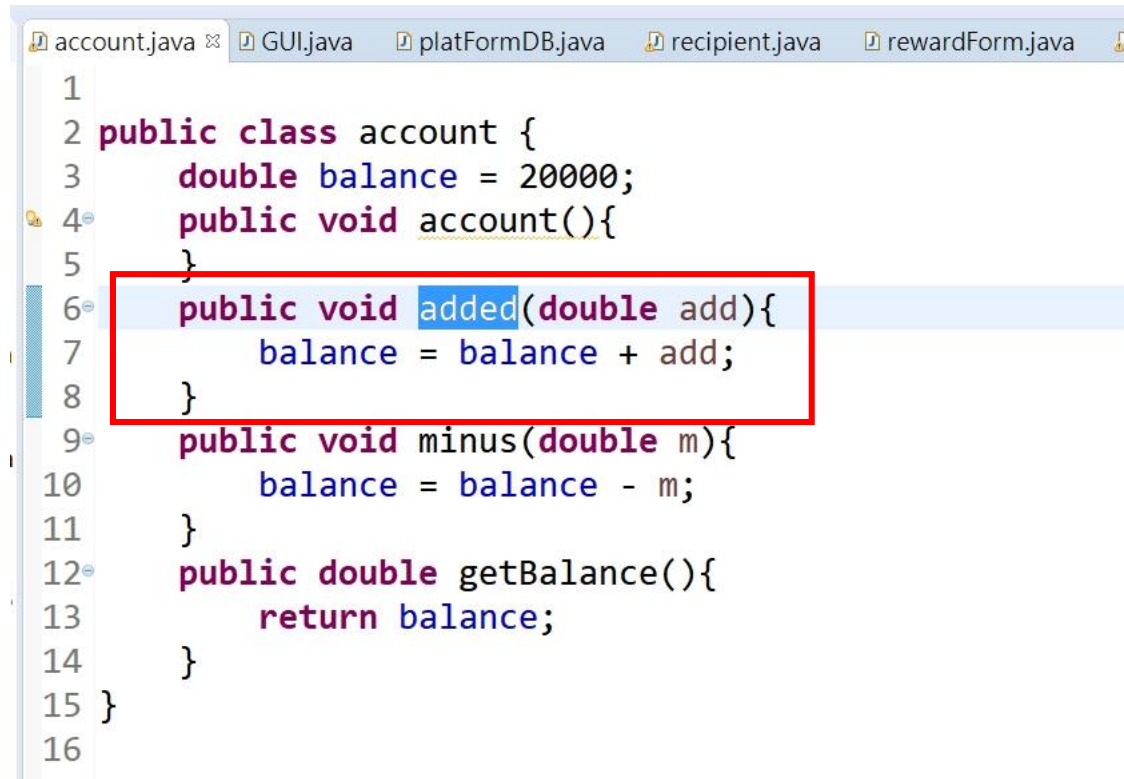
(4) to an object that is created by the method

```
public void create() {  
    rewardForm rew = new rewardForm();  
    rew.enter();  
}  
public void edit(){  
    /*edit feature*/  
}  
public void report(){  
    /*report feature*/  
}
```

2. There are six (or seven) types of interaction coupling, each falling on different parts of a good-to-bad continuum. Choose three pieces of your project to describe what types of the coupling they belong to.

(1)Data:

The calling method passes a variable to the called method.



```
1
2 public class account {
3     double balance = 20000;
4     public void account(){
5     }
6     public void added(double add){
7         balance = balance + add;
8     }
9     public void minus(double m){
10        balance = balance - m;
11    }
12    public double getBalance(){
13        return balance;
14    }
15 }
16
```

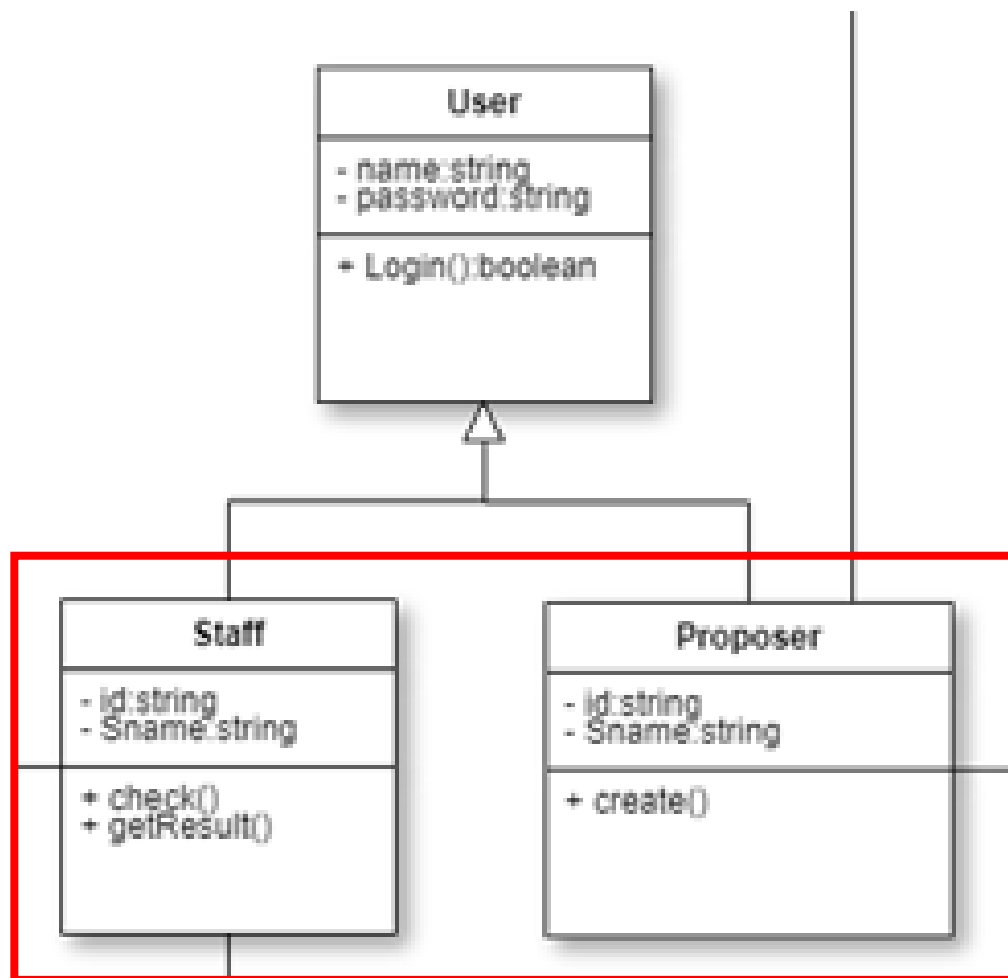
(2) Stamp:

The calling method passes a composite variable to the called method.

```
import java.util.Scanner;
public class validation {
    Scanner keyboard = new Scanner(System.in);
    String deadline;
    String[] content = new String[500];
    static double amount;
    static double balance;
    public void validation(String Deadline, String[] Content, double Amount){
        /*illegal words validate*/
        /*If there are illegal words inside*/
        /*set it a mark*/
        deadline = Deadline;
        content = Content;
        amount = Amount;
        account acc = new account();
        getB(acc);
        mark ma = new mark();
        GUI msg = new GUI();
        while (validatebalance()){
            msg.displayMsg("##insufficient balance.##");
            System.out.printf("amount:");
            Amount = keyboard.nextDouble();
            amount = Amount;
        }
    }
}
```

```
public void getB (account acc) {
    balance=acc.getBalance();
}
```

(3) No Direct Coupling: Staff do not relate to Proposer, they do not call one another.



```
if(identity == 1){
    proposer pro = new proposer();
    pro.proposer();
}
else if(identity == 2){
    recipient rec = new recipient();
    rec.recipient();
}
else if(identity == 3){
    staff sta = new staff();
    sta.staff();
}
```

3. There are seven types of method cohesion, choose three pieces of your project to describe what types of the cohesion they belong to.

(1) Sequential:

After finishing the log in, decide what the user's identity.

```
public class user {
    String name;
    int password;
    public void log_in(){
        Scanner keyboard = new Scanner(System.in);
        int identity = 0, choose = 0;
        int ans = 0;
        System.out.println("-----identity-----");
        System.out.println("1.proposer");
        System.out.println("2.recipient");
        System.out.println("3.staff");
        System.out.printf("Choose the identity you are:");
        identity = keyboard.nextInt();
        System.out.println("-----Log In-----");
        System.out.printf("Name:");
        name = keyboard.next();
        System.out.printf("Password:");
        password = keyboard.nextInt();
        /*Verify account exists?*/
        if(identity == 1){
            proposer pro = new proposer();
            pro.proposer();
        }
        else if(identity == 2){
            recipient rec = new recipient();
            rec.recipient();
        }
        else if(identity == 3){
            staff sta = new staff();
            sta.staff();
        }
        else {
            System.out.println("Log in again");
            log_in();
        }
    }
}
```

(2) Functional:

One method only do one thing.

```
public class account {  
    double balance = 20000;  
    /*Suppose there are 20,000 balances in the account*/  
    public void account(){  
    }  
    public void added(double add) {  
        balance = balance + add;  
    }  
    public void minus(double m) {  
        balance = balance - m;  
    }  
    public double getBalance() {  
        return balance;  
    }  
}
```


(3)Logical:

The method supports multiple related functions, but the choice of the specific function is chosen based on a control variable.

```
account.java GUI.java platFormDB.java recipient.java rewardForm.java user.java staff.java valid
1 import java.util.Scanner;
2 public class validation {
3     Scanner keyboard = new Scanner(System.in);
4     String deadline;
5     String[] content = new String[500];
6     static double amount;
7     static double balance;
8     public void validation(String Deadline, String[] Content, double Amount){
9         /*illegal words validate*/
10        /*If there are illegal words inside*/
11        /*set it a mark*/
12        deadline = Deadline;
13        content = Content;
14        amount = Amount;
15        account acc = new account();
16        getB(acc);
17        mark ma = new mark();
18        GUI msg = new GUI();
19        while (validatebalance()){
20            msg.displayMsg("##insufficient balance.##");
21            System.out.printf("amount:");
22            Amount = keyboard.nextDouble();
23            amount = Amount;
24        }
25        if(validate10000()){
26            msg.displayMsg("##amount exceed NT$10000.##");
27            ma.setMark();
28        }
29        if(ma.getResult()){
30            staff sta = new staff();
31            System.out.println("##give to staff for validate##");
32            sta.check(Deadline, Content, Amount);
33        }else{
34            System.out.println("##the case has been sent out.##");
35            acc.minus(Amount);
36            platFormDB rDB = new platFormDB();
37            rDB.update(Deadline, Content, Amount);
38        }
39    }
40 }
```

4. Connascence generalized the ideas of cohesion and coupling, use three pieces of your project to describe what types of the connascence they belong to.

(1)Type or Class: “enter” use the attribute “content” which is declared as “String[]”, change its type then it might cause “content” can’t save data.

```
public class rewardForm {
    String deadline;
    String[] content = new String[500];
    double amount;
    Scanner keyboard = new Scanner(System.in);
    public void enter() {
        System.out.println("-----Create Reward Case-----");
        System.out.printf("deadline:");
        deadline = keyboard.next();
        System.out.printf("content(input -1 to stop):");
        for(int i=0;i<500;i++){
            content[i]=keyboard.next();
            if(content[i].indexOf("-1") != -1){
                break;
            }
        }
        System.out.printf("amount:");
        amount = keyboard.nextDouble();
        validation val = new validation();
        val.validation(deadline, content, amount);
    }
    public void editRewardForm() {
        /*edit features*/
    }
}
```

(2) Position: “CaculateBalance”:

If the code A and B swapped the position , the result will be wrong.

```
public class account {
    double balance = 20000;
    /*Suppose there are 20,000 balances in the account*/
    public void account(){
    }
    public void added(double add) {
        balance = balance + add;
    }
    public void minus(double m) {
        balance = balance - m;
    }
    public double getBalance() {
        return balance;
    }
}
```

A

B

(3)Name: If the name of the attribute “balance” changes, the method “getBalance” will have to change.

```
public class account {  
    double balance = 20000;  
    /*Suppose there are 20,000 balances in the account*/  
    public void account(){  
    }  
    public void added(double add){  
        balance = balance + add;  
    }  
    public void minus(double m){  
        balance = balance - m;  
    }  
    public double getBalance(){  
        return balance;  
    }  
}
```

5. Use one class from your project that can create a set of invariants and add them to the CRC card or the class diagram.

Front:

Class Name: RewardForm	ID: 4	Type: Concrete, Domain
Description: The Proposer can use RewardForm to create Reward		Use Case: 1
Responsibilities enter createMsg appendmsg	Collaborators PlatFormDB Validation Proposer Mark	

Back:

Attributes: content (1..1) (string) amount (1..1) (double) deadline (1..1) (date)
Relationships: Generalization: Aggregation: Other Associations: PlatFormDB{1..1} Validation{1..*} Proposer{1..1} Mark{0..*}

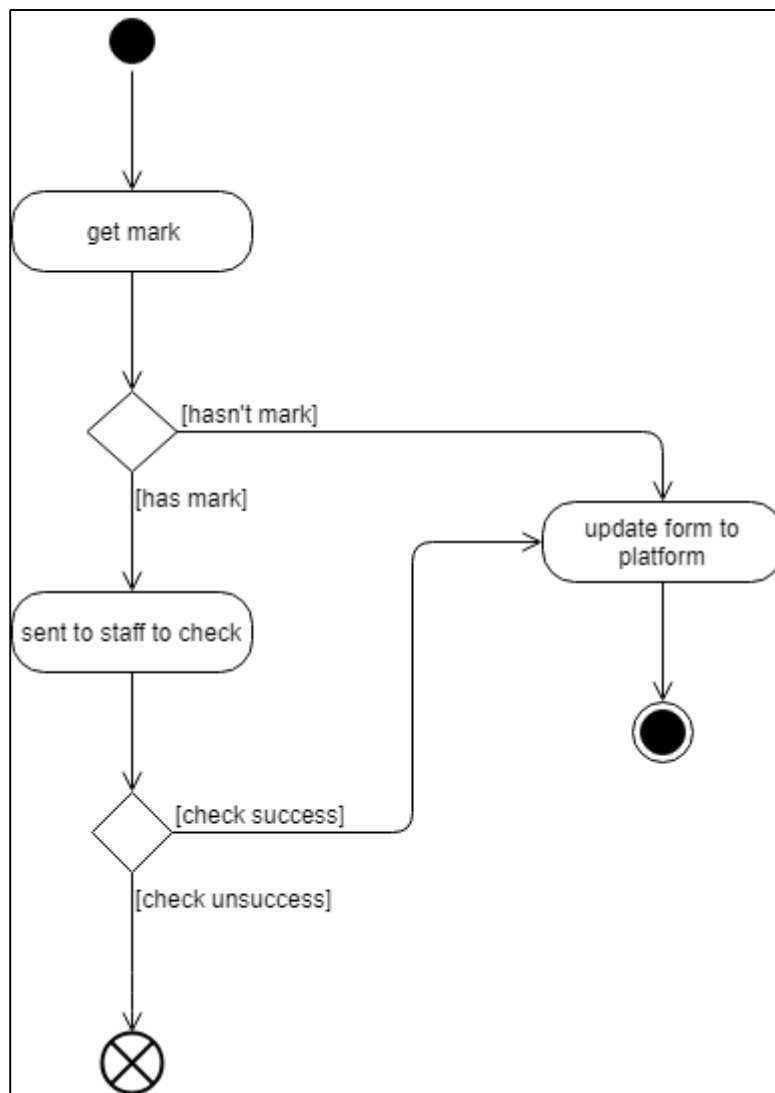
6. Use a method of a class from your project that can create a contract and describe its algorithm specification. Specify the pre- or post- condition and use both Structured English and an activity diagram to specify the algorithm.

Contract :

Method Name: update	Class Name: platFormDB	ID: 6
Client(Consumers): RewardForm		
Associated Use Cases: Create Reward Case		
Description of Responsibilities: The form that have mark but check successful and unmarked form should be update.		
Arguments Received: deadline : String content : String[] amount : double		
Type of Value Returned: void		
Pre-Conditions: Arguments not Null.		
Post-Conditions: Reward is updated successful.		

Method Name: update	Class Name: platFormDB	ID: 6
Contract ID: 5	Programmer: team 8	Date Due: 5/25
Programming Language: Java		
Triggers/Events: After all verification successful		
Arguments Received: Data Type:	Note:	
deadline : String content : String[] amount : double		
Messages Sent & Arguments Passed: ClassName.MethodName:	Data Type:	Notes:
Arguments Returned: Data Type:	Notes:	
void		
Algorithm Specification: IF the form has mark sent to staff to check Else update to the platform		
Misc. Notes:		

Activity diagram



7. Please evaluate any piece of your project in terms of cohesion, coupling, and connascence perspective.

Interaction Coupling:Data

We use this method of account to passes a variable to the called method, that is Interaction coupling which type is data, and in the coupling evaluation is good.

Interaction Coupling:Stamp

We use this method of validation to passes a variable to the called method, it can use the part of the object to do the method. that is Interaction coupling which type is data, and in the coupling evaluation is even worse than Data but it looks good overall.

Method Cohesion: Functional

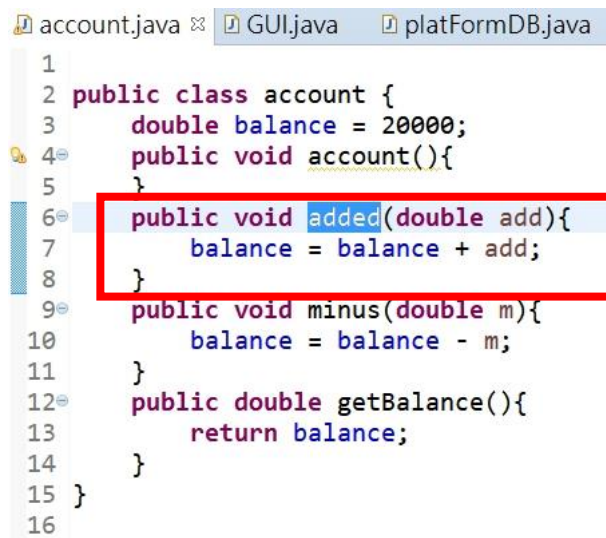
In the method of account, each method will only focus on the tasks that it has designed. This evaluation of the cohesion is not bad.

Method Cohesion: Logical

In the method of validation, the method of the called can verify whether the amount is insufficient and whether the form needs to be marked, which is entirely determined by the message transmitted by the validation method. This evaluation of the cohesion is not good.

Connascence: Name

If the name of the attribute “balance” changes, the method “getBalance” will have to change.



```
1
2 public class account {
3     double balance = 20000;
4     public void account(){
5     }
6     public void added(double add){
7         balance = balance + add;
8     }
9     public void minus(double m){
10        balance = balance - m;
11    }
12    public double getBalance(){
13        return balance;
14    }
15 }
16
```

```

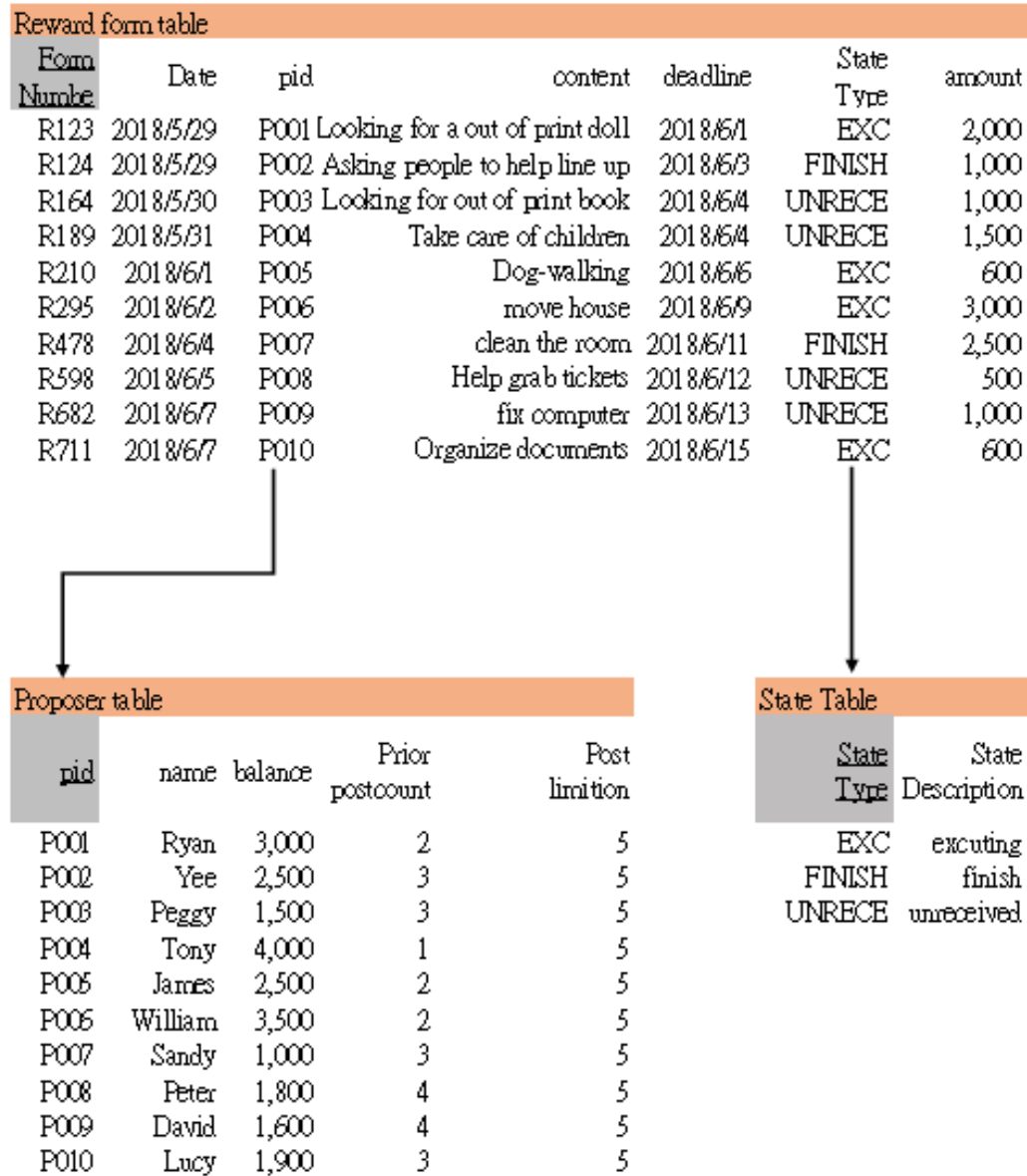
import java.util.Scanner;
public class validation {
    Scanner keyboard = new Scanner(System.in);
    String deadline;
    String[] content = new String[500];
    static double amount;
    static double balance;
    public void validation(String Deadline, String[] Content, double Amount){
        /*illegal words validate*/
        /*If there are illegal words inside*/
        /*set it a mark*/
        deadline = Deadline;
        content = Content;
        amount = Amount;
        account acc = new account();
        getB(acc);
        mark ma = new mark();
        GUI msg = new GUI();
        while (validatebalance()){
            msg.displayMsg("##insufficient balance.##");
            System.out.printf("amount:");
            Amount = keyboard.nextDouble();
            amount = Amount;
        }
        if(validate10000()){
            msg.displayMsg("##amount exceed NT$10000.##");
            ma.setMark();
        }
        if(ma.getResult()){
            staff sta = new staff();
            System.out.println("##give to staff for validate##");
            sta.check(Deadline, Content, Amount);
        }else{
            System.out.println("##the case has been sent out.##");
            acc.minus(Amount);
            platFormDB rDB = new platFormDB();
            rDB.update(Deadline, Content, Amount);
        }
    }
    public void getB (account acc) {
        balance=acc.getBalance();
    }
    public static boolean validatebalance(){
        if(balance >= amount){
            return false;
        }else{
            return true;
        }
    }
    public static boolean validate10000(){
        if(amount > 10000){
            return true;
        }else{
            return false;
        }
    }
}

```


8. Assume that you are going to adopt RDBMs to your project, please describe the referential integrity.

Referential integrity :

1. All pid values in the Reward form table must exist in the Proposer table beforehand.
2. All StateType values in the Reward form table must exist in the StateType table beforehand.



9. Using the steps of normalization, create a model that represents the file of your project in third normal form. Please make necessary assumptions to explain why the tables are related.

The model does not have redundant data and null cells, which means it is in first normal form (1NF).

For all classes, primary key is made up of one field, which means the model is in second normal form (2NF).

However, the StateDescription only depends on StateType, and StateType is not a primary key, we add a new table called “State Table”, which set StateType as primary key, and StateDescription depends on it, the data model is in third normal form (3NF).

3NF

Reward form Table

Reward form table						
<u>Form Number</u>	Date	pid	content	deadline	State Type	amount
R123	2018/5/29	P001	Looking for a out of print doll	2018/6/1	EXC	2,000
R124	2018/5/29	P002	Asking people to help line up	2018/6/3	FINISH	1,000
R164	2018/5/30	P003	Looking for out of print book	2018/6/4	UNRECE	1,000
R189	2018/5/31	P004	Take care of children	2018/6/4	UNRECE	1,500
R210	2018/6/1	P005	Dog-walking	2018/6/6	EXC	600
R295	2018/6/2	P006	move house	2018/6/9	EXC	3,000
R478	2018/6/4	P007	clean the room	2018/6/11	FINISH	2,500
R598	2018/6/5	P008	Help grab tickets	2018/6/12	UNRECE	500
R682	2018/6/7	P009	fix computer	2018/6/13	UNRECE	1,000
R711	2018/6/7	P010	Organize documents	2018/6/15	EXC	600

Proposer Table

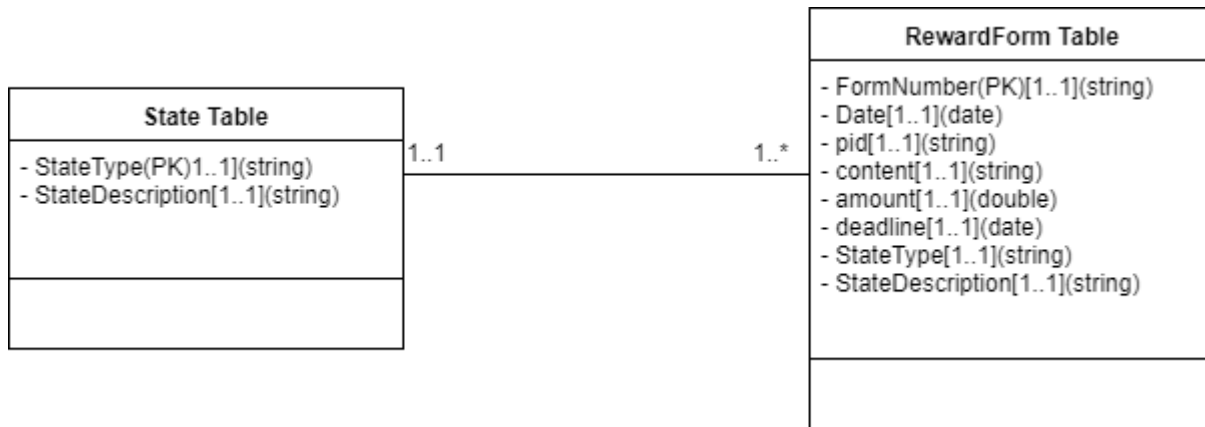
Proposer table				
<u>pid</u>	name	balance	Prior postcount	Post limition
P001	Ryan	3,000	2	5
P002	Yee	2,500	3	5
P003	Peggy	1,500	3	5
P004	Tony	4,000	1	5
P005	James	2,500	2	5
P006	William	3,500	2	5
P007	Sandy	1,000	3	5
P008	Peter	1,800	4	5
P009	David	1,600	4	5
P010	Lucy	1,900	3	5

State Table

State Table	
<u>State Type</u>	State Description
EXC	excuting
FINISH	finish
UNRECE	unreceived

10. Describe how you would denormalize the model that you created in question 9. Draw the new class diagram based on your suggested changes.

When we search for the RewardForm Table, we need its state description to let us understand the status of the RewardForm, so we add StateDescription in the RewardForm Table, we don't need to join the State Table when searching for RewardForm Table.



11. Examine the model that you created in question 10. Develop the inter-file clustering and index strategies. Describe how your clustering strategy will improve the performance of the database. List possible indices you would recommend and describe the reasons.

We combine RewardForm table and state table, because they usually search together.

RewardForm clustering							State clustering	
FormNumber	Date	pid	content	deadline	StateType	amount	StateType	StateDescription
R124	2018/5/29	P002	Asking people to help line up	2018/6/3	FINISH	1,000	FINISH	finish
R478	2018/6/4	P007	clean the room	2018/6/11	FINISH	2,500	FINISH	finish
R123	2018/5/29	P001	Looking for a out of print doll	2018/6/1	EXC	2,000	EXC	excuting
R210	2018/6/1	P005	Dog-walking	2018/6/6	EXC	600	EXC	excuting
R295	2018/6/2	P006	move house	2018/6/9	EXC	3,000	EXC	excuting
R711	2018/6/7	P010	Organize documents	2018/6/15	EXC	600	EXC	excuting
R164	2018/5/30	P003	Looking for out of print book	2018/6/4	UNRECE	1,000	UNRECE	unreceived
R189	2018/5/31	P004	Take care of children	2018/6/4	UNRECE	1,500	UNRECE	unreceived
R598	2018/6/5	P008	Help grab tickets	2018/6/12	UNRECE	500	UNRECE	unreceived
R682	2018/6/7	P009	fix computer	2018/6/13	UNRECE	1,000	UNRECE	unreceived

We used StateType as our Index to let our system to search the data of RewardForm faster.

State_Index				RewardForm Table							
StateType	pointer			FormNumber	Date	pid	content	amount	deadline	stateType	State Description
EXC	R123			→ R123	2018/5/29	P001	Looking for a out of print doll	2,000	2018/6/1	EXC	excuting
EXC	R210			→ R124	2018/5/29	P002	Asking people to help line up	1,000	2018/6/3	FINISH	finish
EXC	R295			→ R164	2018/5/30	P003	Looking for out of print book	1,000	2018/6/4	UNRECE	unreceived
EXC	R711			→ R189	2018/5/31	P004	Take care of children	1,500	2018/6/4	UNRECE	unreceived
UNRECE	R164			→ R210	2018/6/1	P005	Dog-walking	600	2018/6/6	EXC	excuting
UNRECE	R189			→ R295	2018/6/2	P006	move house	3,000	2018/6/9	EXC	excuting
UNRECE	R598			→ R478	2018/6/4	P007	clean the room	2,500	2018/6/11	FINISH	finish
UNRECE	R682			→ R598	2018/6/5	P008	Help grab tickets	500	2018/6/12	UNRECE	unreceived
FINISH	R124			→ R682	2018/6/7	P009	fix computer	1,000	2018/6/13	UNRECE	unreceived
FINISH	R478			→ R711	2018/6/7	P010	Organize documents	600	2018/6/15	EXC	excuting

B10423023	Ryan	Discuss all diagram& do the 9,10,11 question	100
A10523048	Tony	Discuss all diagram& do the 1,5 question	100
B10423022	William	Discuss all diagram& do the 1,5 question	90
B10523006	Peggy	Discuss all diagram& do the 2,3,4 question	100
B10523013	Yvonne	Discuss all diagram& do the 6,7,8 question	100
B10523017	Peggy	Discuss all diagram& do the 9,10,11 question	100
B10523028	James	Discuss all diagram& do the 1,5 question	80
B10523037	Yee	Discuss all diagram& Code	100
B10523056	Sandy	Discuss all diagram& do the 6,7,8 question	100