Name: 林佩憓 Althee   ID: B10323019

1. An Employee has a one-to-one association with an Account object that tracks all
    the incomes and charges accrued from transactions in the ABC Company. The
    Customer can call the operations of the Account object, but the Account never
    invokes operations of the Customer. Since the reference to the Account object
    does not change over time, we need to do something to prevent callers from
    accidentally modifying the Account. Please write the Java code. 5% What
    principle?

2. The direction of an association changes during the development of the system.
    Assume that we modify the Account class so that the display name of the Account
    is updated from the name of the Employee. In this situation, the Account needs to
    access its corresponding Employee object. Therefore, we plan to add an owner
    attribute to Account. We need to ensure that if a given Account has a reference to
    a specific Employee, and the Employee has a reference to that same Account.
    Since neither the Employee class nor the Account class can modify the field
    anywhere else, this ensures that both reference attributes remain consistent. Please
    write the Java code. 8% (Hint: As the Account object is created by the
    Employee constructor, add a parameter to the Account constructor to initialize the
    owner field to the correct value.)

3. Time passes, and the direction of an association changes again during the
    development of the system. Assume that an Employee can have several Accounts
    to track the expenses accrued for different projects. In this situation, the Employee
    object has a one-to-many association with the Account class. That means an
    Employee can have a varying number of Accounts. Since the Account object does
    not invoke the Account constructor, we design a control object for creating and
    archiving Accounts for invoking the constructor. Remember that Accounts have
    no specific order and an Account can be part of an Employee at most once, we use
    references, called accounts to model the "many" part of the association. Moreover,
    we decide to realize this association as a bidirectional association, and so add the
    addAccount(), removeAccount(), and setOwner() methods to the Employee and
    Account classes to update the accounts and owner fields. Please write the Java
    code. 10%

Preconditions and postconditions can be used to specify dependencies among

12. We want to map the problem domain objects to RDBMS Schema. Please draw its corresponding diagram. 6%
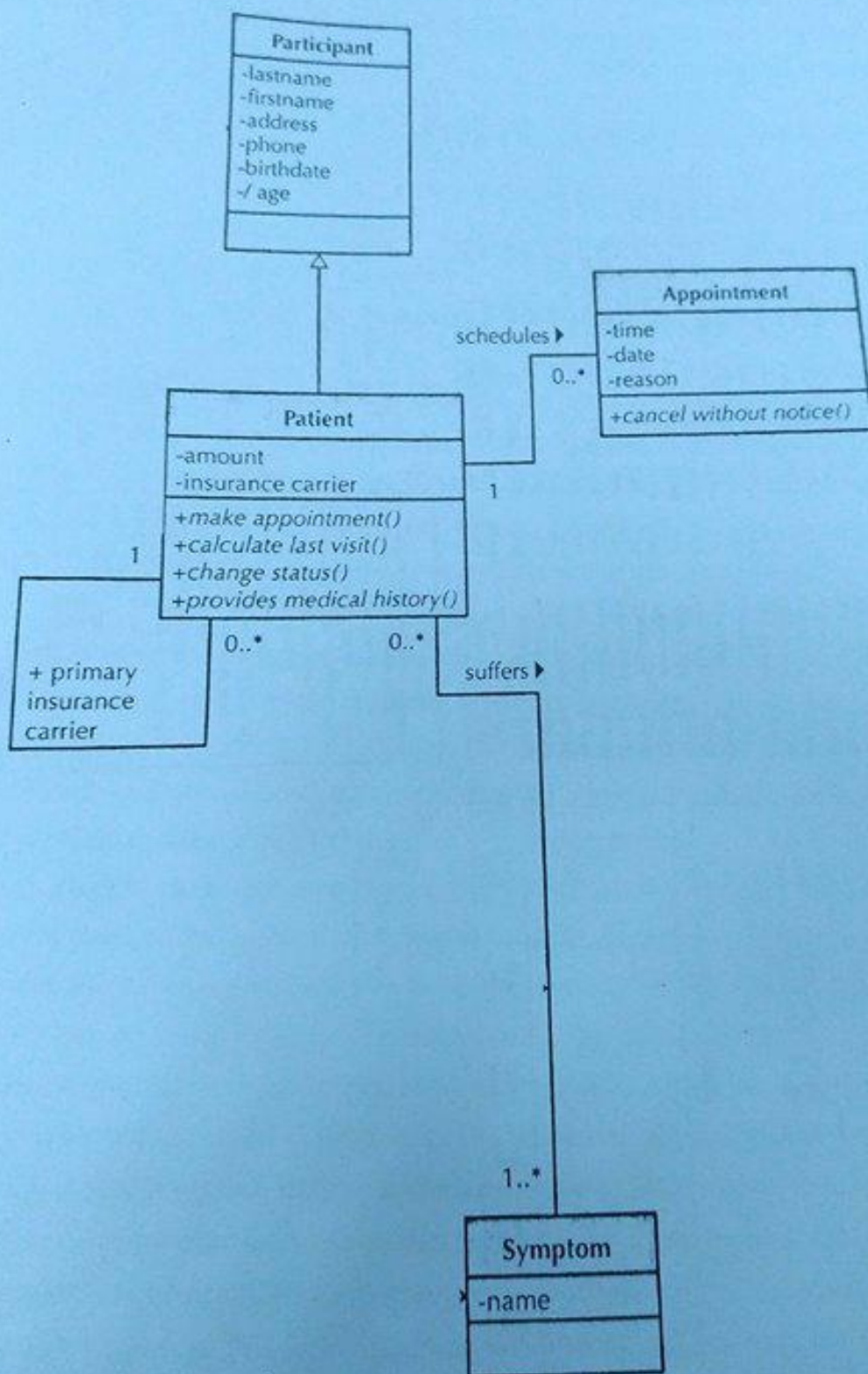
Problem Domain Classes

```
                    ┌─────────────────────┐
                    │     Participant     │
                    ├─────────────────────┤
                    │ -lastname           │
                    │ -firstname          │
                    │ -address            │
                    │ -phone              │
                    │ -birthdate          │
                    │ -/ age              │
                    ├─────────────────────┤
                    │                     │
                    └─────────────────────┘
                              △
                              │
                                              schedules ▶      ┌─────────────────────────────┐
                                                                │         Appointment         │
                                                                ├─────────────────────────────┤
                                                                │ -time                       │
                                                                │ -date                       │
                                                       0..*     │ -reason                     │
   ┌──────────────────────────────────┐                        ├─────────────────────────────┤
   │             Patient              │                        │ +cancel without notice()    │
   ├──────────────────────────────────┤                        └─────────────────────────────┘
   │ -amount                          │         1
   │ -insurance carrier               │
   ├──────────────────────────────────┤
   │ +make appointment()              │
   │ +calculate last visit()          │
   │ +change status()                 │
   │ +provides medical history()      │
   └──────────────────────────────────┘
      1        0..*           0..*
                                       suffers ▶
┌──────────────┐
│ + primary    │
│ insurance    │
│ carrier      │
└──────────────┘
```

1..*

```
        ┌─────────────────┐
        │    Symptom      │
        ├─────────────────┤
        │ -name           │
        ├─────────────────┤
        │                 │
        └─────────────────┘
```

13. Given the following Order figure displaying redundant data and null cells in the file. Please use normalization rule to create first normal form, second normal form, and third normal form for it. Please also **add necessary attributes** to the normalized form. You need to identify primary and foreign keys and explain what referential integrity is. 15%

[handwritten] table & class

[handwritten] primary key

[handwritten] forie key

Sample Records:

Redundant Data

Null Cells

Order

- Order Number : unsigned long
- Date : Date
- Cust ID : unsigned long
- Last Name : String
- First Name : String
- State : String
- Tax Rate : float
- Product 1 Number : unsigned long
- Product 1 Desc. : String
- Product 1 Price : double
- Product 1 Qty. : unsigned long
- Product 2 Number : unsigned long
- Product 2 Desc. : String
- Product 2 Price : double
- Product 2 Qty. : unsigned long
- Product 3 Number : unsigned long
- Product 3 Desc. : String
- Product 3 Price : double
- Product 3 Qty. : unsigned long

Order Number	Date	Cust ID	Last Name	First Name	State	Tax Rate	Prod. 1 Number	Prod. 1 Desc.	Prod. 1 Price	Prod. 1 Qty.	Prod. 2 Number	Prod. 2 Desc.	Prod. 2 Price	Prod. 2 Qty.	Prod. 3 Number	Prod. 3 Desc.	Prod. 3 Price	Prod. 3 Qty.
239	11/23/00	1035	Black	John	MO	0.05	555	Cheese Tray	$45.00	2								
240	11/24/00	1035	Black	John	MO	0.05	444	Wine Gift Pack	$60.00	1								
233	11/27/00	1035	Black	John	MO	0.05	222	Bottle Opener	$12.00	1								
241	11/23/00	1123	William	Mary	CA	0.08	444	Wine Gift Pack	$60.00	1								
282	11/24/00	1123	William	Mary	CA	0.08	222	Bottle Opener	$12.00	2								
287	11/27/00	1123	William	Mary	CA	0.08	222	Bottle Opener	$12.00	2								
290	11/09/00	1123	William	Mary	CA	0.08	555	Cheese Tray	$45.00	3								
234	11/03/00	2242	Delivery	Ann	DC	0.065	555	Cheese Tray	$45.00	2								
212	11/23/00	2242	Delivery	Ann	DC	0.065	111	Wine Guide	$15.00	1								
238	11/24/00	2242	Delivery	Ann	DC	0.065	444	Wine Gift Pack	$60.00	1	444	Wine Gift Pack	$60.00	1				
245	11/24/00	2242	Delivery	Ann	DC	0.065	222	Bottle Opener	$12.00	1								
250	11/24/00	2242	Delivery	Ann	DC	0.065	222	Bottle Opener	$12.00	1								
252	11/24/00	2242	Delivery	Ann	DC	0.065	222	Bottle Opener	$12.00	1								
253	11/24/00	2242	Delivery	Ann	DC	0.065	222	Bottle Opener	$12.00	2	444	Wine Gift Pack	$60.00	2				
297	11/00/00	2242	Delivery	Ann	DC	0.065	222	Bottle Opener	$12.00	1	444	Wine Gift Pack	$60.00	1				
243	11/24/00	4254	Baldy	Ryan	MO	0.05	333	Jams & Jellies	$20.00	2								
246	11/24/00	4254	Baldy	Ryan	MO	0.05	555	Cheese Tray	$45.00	2								
248	11/04/00	4254	Baldy	Ryan	MO	0.05	333	Jams & Jellies	$20.00	1								
235	11/03/00	9500	Orie	April	KS	0.05	222	Bottle Opener	$12.00	3								
242	11/03/00	9500	Orie	April	KS	0.05	222	Bottle Opener	$12.00	1	333	Jams & Jellies	$20.00	2	111	Wine Guide	$15.00	1
244	11/24/00	9500	Orie	April	KS	0.05	333	Jams & Jellies	$20.00	3								
251	11/24/00	9500	Orie	April	KS	0.05	222	Bottle Opener	$12.00	2								

Please state the type of cohesion for the following situations.

1) A system initialization routine: this routine contains all of the code for initializing all of the parts of the system. Lots of different activities occur, all at init time. _____ | 2% |

2) An object "calculate totals" may keep a running total of the quantity times price subtotal for each item. _____ | 2% |

8. Please use an example to specify a method's algorithm for a compute pay method associated with an hourly employee class using an activity diagram. The procedure should include recognition of employee status, the check for hourly employment, calculate the number of hours worked, calculate tax, and the printing of check. | 5% |　　　　活动图

9. Use the following example to describe the following questions:
   1) What is encapsulation? | 3% |
   2) What does the "final" mean? | 2% |

```
public class Animal {
    private String name = "Animal";   // The attribute is hidden
    public String favFood = "Food";

    protected final void changeName(String newName){
        this.name = newName;
    }

    protected final String getName(){
        return this.name;
    }

    public void eatStuff(){
        System.out.println("Yum " + favFood);
    }

    public void walkAround(){
        System.out.println(this.name + " walks around");
    }
}
```

10. Use the following example to describe polymorphism | 5% |.

```
public static void main(String[] args) {
    StreetRacer streetRacer = new StreetRacer();
    FormulaOne formulaOne = new FormulaOne();
    Helicopter helicopter = new Helicopter();
    Jet jet = new Jet();


    streetRacer.go();
```

3

pre-condition   post-condition

operations in the same class. Please select the right statement for the following situations. 8%

1) To ensure that we invoke TournamentControl to select sponsor only once.

4

2) To assume that the Player is not yet part of the Tournament of interest.

1

3) To ensure that sponsors cannot be selected before there are interested advertisers. *3*

4) To specify how TournamentControl sets the advertisers association when select sponsor. *2*

context TournamentControl::IsPlayerOverbooked(p) **pre:** not p.tournaments->includes(self.tournament)	(1)

context TournamentControl:: selectSponsors (advertisers) **post:** tournament.sponsors->sponsors.equals(advertisers)	(2)

context TournamentControl::selectSponsors (advertisers) **pre:** interestedSponsors->notEmpty()	(3)

context TournamentControl::selectSponsors (advertisers) **pre:** tournament.sponsors->isEmpty()	(4)

5. What is an invariant? 2% *不变式* Give an example of an invariant for an hourly employee class. 3%

6. There are six types of interaction coupling including no direct coupling, data coupling, stamp coupling, control coupling, common or global coupling, and content or pathological coupling. Please state the type of interaction for the following situations.

1) When customer not found, component adds customer by directly modifying the contents of the data structure containing customer data.
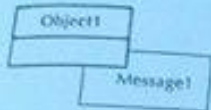   _____ 2%

2) The print routine of the customer billing accepts customer data structure as an argument, parses it, and prints the name, address, and billing information. _____ 2% How to solve this problem? 5%
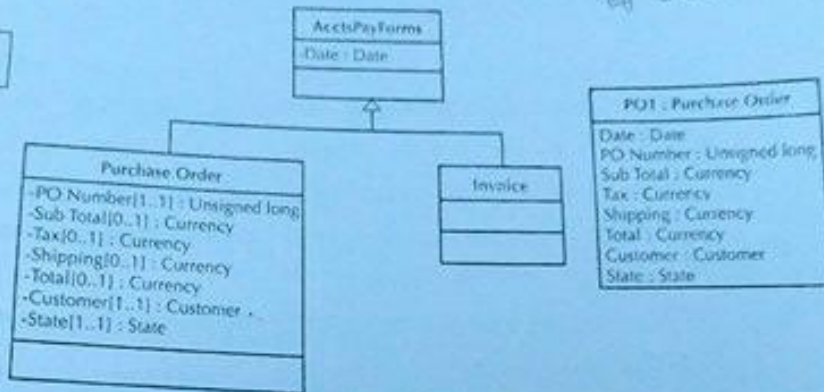
7. There are six types of interaction coupling including functional, sequential, communicational, procedural, temporal or classical, logical, and coincidental.

```
    formulaOne.go();
    helicopter.go();
    jet.go();
```

11. Please apply the Law of Demeter (LoD) to the following example. 15%.

1) code

(a)

(b)

4