作业要求：

1、时间要求：提交作业的截止日期就是 11 月 15 日；

2、形式要求：以电子档的方式提交，通过 Email 到 homework.xjtu@qq.com 邮箱，邮件主题的格式为"班级--学号--姓名"即可；（比如：软件 31-2131601100-张五）

3、内容要求：提交源程序（以压缩文件格式）和一个作业报告，其中作业报告为 word 文档，包含设计过程、主干代码呈现和运行结果展示。请不要将 word 文档和源程序压缩到一个包中，谢谢。

Homework:

1. Implementing Some Inherited Classes

(The Person, Student, Employee, Faculty, and Staff classes) Design a class named Person and its two subclasses named Student and Employee.Make Faculty and Staff subclasses of Employee. A person has a name,address, phone number, and email address. A student has a class status (freshman,sophomore, junior, or senior). Define the status as a constant. An employee has an office, salary, and date hired. Define a class named MyDate that contains the fields year, month, and day. A faculty member has office hours and a rank. A staff member has a title. Override the toString method in each class to display the class name and the person's name.

Draw the UML diagram for the classes. Implement the classes. Write a test program that creates a Person,Student, Employee, Faculty, and Staff, and invokes their toString() methods.

Extension:Modify each class to implementing Comparable interface,so they are compared using the compareTo method.The method can be used to sort an array of any objects as long as their classes implement the Comparable interface.Write a test program to verify sorting.

2. Implementing an Account class

Design a class named Account that contains:
■ A private int data field named id for the account (default 0).
■ A private double data field named balance for the account (default 0).
■ A private double data field named annualInterestRate that stores the current interest rate (default 0).Assume all accounts have the same interest rate.
■ A private Date data field named dateCreated that stores the date when the account was

created.

■ A private String data field named name for the name of the customer.

■ A private data field named transactions whose type is ArrayList (a jdk api class)that stores the transaction for the accounts. Each transaction is an instance of the Transaction class. The Transaction class is defined as shown in Figure 1.

■ A no-arg constructor that creates a default account.

■ A constructor that creates an account with the specified id and initial balance.

■ A constructor that constructs an account with the specified name, id, and balance.

■ The accessor and mutator methods for id,name, balance, and annualInterestRate.

■ The accessor method for dateCreated.

■ A method named getMonthlyInterestRate() that returns the monthly interest rate.

■ A method named withdraw that withdraws a specified amount from the account and add a transaction to the transactions array list.

■ A method named deposit that deposits a specified amount to the account and add a transaction to the transactions array list.

Implement the class. Write a test program that creates an Account with annual interest rate 1.5%,balance 1000, id 1122, and name George. Deposit $30, $40, $50 to the account and withdraw $5, $4, $2 from the account. Print an account summary that shows account holder name, interest rate, balance, and all transactions.

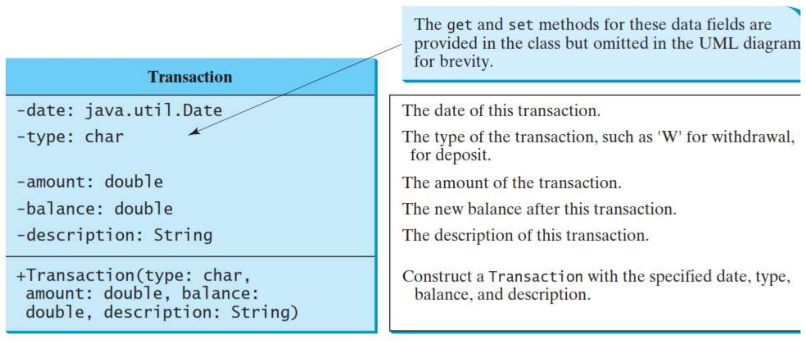
The get and set methods for these data fields are provided in the class but omitted in the UML diagram for brevity.
Transaction
-date: java.util.Date
-type: char
-amount: double
-balance: double
-description: String
+Transaction(type: char, amount: double, balance: double, description: String)
The date of this transaction.
The type of the transaction, such as 'W' for withdrawal, 'D' for deposit.
The amount of the transaction.
The new balance after this transaction.
The description of this transaction.
Construct a Transaction with the specified date, type, balance, and description.

Figure 1