[OOP].1. Account Class

- **Problem**
- Submissions
- Discussions

Cho lớp Account gồm các thuộc tính :

- ID : Mã account
- customerID : Mã khách hàng sở hữu Account
- Username : Tên tài khoản
- Password : Mật khẩu

Cho một danh sách tài khoản sẵn có trong cơ sở dữ liệu, bạn hãy kiểm tra xem các đăng nhập của người dùng có hợp lệ hay không và thông báo ra màn hình.

```
import java.util.Scanner;

public class Main {
    public static void main(String[] args) {
        Scanner sc = new Scanner(System.in);
        int n = sc.nextInt();
        sc.nextLine();
        Account[] accountList = new Account[n];
        for(int i = 0; i < n; i++){
            accountList[i] = new Account(sc.nextLine(), sc.nextLine(), sc.nextLine(), sc.nextLine());
        }
        int q = sc.nextInt(); sc.nextLine();
        for(int i = 0; i < q; i++){
            String username = sc.nextLine();
            String password = sc.nextLine();
            boolean check = false;
            for(int j = 0; j < n; j++){
                if(accountList[j].checkLogin(username, password)){
                    check = true;
                    break;
                }
            }
            if(check){
                System.out.println("Login Successful");
            }
            else{
                System.out.println("Login Failed");
            }
```

```
    }
  }
}

class Account {
  private String id, customerId, username, password;

  public Account(String id, String customerId, String username, String password) {
    this.id = id;
    this.customerId = customerId;
    this.username = username;
    this.password = password;
  }

  public boolean checkLogin(String username, String password){
    if(this.username.equals(username) && this.password.equals(password)){
      return true;
    }
    else{
      return false;
    }
  }
}
```

Input Format

- Dòng 1 là N : số lượng Account
- 4 * N dòng tiếp theo mô tả thông tin của từng Account, mỗi Account gồm 4 dòng lần lượt là ID, customerID, Username, Password
- Dòng tiếp theo là Q : số lượng lượt đăng nhập
- 2 * Q dòng tiếp theo mô tả tên đăng nhập và mật khẩu của từng người dùng

Constraints

- 1 <= N,Q <= 10000
- ID, customerID, Username, Password là các xâu ký tự có không quá 50 ký tự.

Output Format

- Đối với mỗi lượt truy cập in ra "Login Successful" nếu truy cập thành công và "Login Failed" nếu truy cập thất bại.

Sample Input 0

```
2
346sds
cus002
28tech
```

```
28tech123xyz
122clo
cus009
nguyenmy
my123456
1
28tech
28tech123xyz
```

Sample Output 0

```
Login Successful
```

[OOP].2. Bank Account

- **Problem**
- Submissions
- Discussions

Cho lớp BankAccount gồm những thông tin :

- ID : Mã Bank account
- CustomerID : Mã khách hàng sở hữu tài khoản này
- Số tài khoản
- PIN Code : Mã pin
- Số dư tài khoản : Số nguyên

Bạn hãy cập nhật số dư của từng tài khoản sau khi thực hiện 1 loạt các giao dịch chuyển, rút, nạp tiền.

- Nếu là giao dịch chuyển tiền sẽ có dạng : X Y Z trong đó X là số tài khoản người gửi, Y là số tài khoản người nhận, Z là số tiền. Bạn chỉ được thực hiện giao dịch nếu số dư khả dụng tài khoản X lớn hơn hoặc bằng số tiền cần chuyển Z, biết rằng để duy trì tài khoản cần 50.000
- Nếu giao dịch là rút sẽ có dạng : X Y, trong đó X là tài khoản rút tiền, Y là số tiền cần rút, chỉ thực hiện giao dịch khi số dư khả dụng của tài khoản X lớn hơn hoặc bằng số tiền cần rút Y
- Nếu giao dịch là nạp sẽ có dạng : X Y, trong đó X là tài khoản nạp tiền, Y là lượng tiền cần nạp

```
import java.util.Scanner;

public class Main {
    public static void main(String[] args) {
        Scanner sc = new Scanner(System.in);
        int n = sc.nextInt();
        Account[] accountList = new Account[n];
        for(int i = 0; i < n; i++){
            sc.nextLine();
            accountList[i] = new Account(sc.nextLine(), sc.nextLine(), sc.nextLine(), sc.nextLine(),
sc.nextInt());
        }
        int q = sc.nextInt();
        for(int i = 0; i < q; i++){
            sc.nextLine();
            String tran = sc.nextLine();
            if(tran.equals("withdraw")){
                String taiKhoan = sc.next();
                int tien = sc.nextInt();
                int pos = Account.findPos(accountList, taiKhoan);
                accountList[pos].withdraw(tien);
            }
            else if(tran.equals("deposit")){
                String taiKhoan = sc.next();
                int tien = sc.nextInt();
                int pos = Account.findPos(accountList, taiKhoan);
                accountList[pos].deposit(tien);
            }
            else{
                String X = sc.next();
                String Y = sc.next();
                int tien = sc.nextInt();
                int pos1 = Account.findPos(accountList, X);
                int pos2 = Account.findPos(accountList, Y);
                if(accountList[pos1].getSoDu() - 50000 >= tien){
                    accountList[pos1].withdraw(tien);
                    accountList[pos2].deposit(tien);
                }
            }
        }
        for(Account x : accountList){
            System.out.println(x);
        }
    }
}

class Account {
    private String id, customerId;
    private String soTaiKhoan, PIN;
    private int soDu;

    public Account(String id, String customerId, String soTaiKhoan, String PIN, int soDu) {
        this.id = id;
```

```
    this.customerId = customerId;
    this.soTaiKhoan = soTaiKhoan;
    this.PIN = PIN;
    this.soDu = soDu;
  }

  public void deposit(int tien){
    this.soDu += tien;
  }

  public void withdraw(int tien){
    if(this.soDu - 50000 >= tien){
      this.soDu -= tien;
    }
  }

  public int getSoDu(){
    return this.soDu;
  }
  public static int findPos(Account[] a, String soTaiKhoan){
    for(int i = 0; i < a.length; i++){
      if(a[i].soTaiKhoan.equals(soTaiKhoan)){
        return i;
      }
    }
    return -1;
  }

  public String toString(){
    return "ID : " + this.id + "\nCusID : " + this.customerId + "\nNumber : " + this.soTaiKhoan +
"\nPIN : " + this.PIN
        + "\nBalance : " + this.soDu + "VND\n--------------------";
  }
}
```

Input Format

- Dòng 1 là N : Số lượng BankAccount
- 5 * N dòng tiếp theo mô tả thông tin tài khoản, mỗi tài khoản gồm 5 dòng
- Dòng tiếp theo là T : số giao dịch
- T dòng tiếp theo mô tả giao dịch, mỗi giao dịch gồm 2 dòng, dòng 1 là loại giao dịch : transfer : Chuyển tiền, withdraw : rút tiền, deposit : nạp tiền. Dòng 2 là mô tả giao dịch

Constraints

- 1<=N<=5000
- 1<=T<=5000

- Lượng tiền trong các giao dịch là số nguyên

Output Format

- In ra danh sách tài khoản theo thứ tự ban đầu và các thông tin liên quan, mỗi thông tin trên 1 dòng, các tài khoản viết cách nhau một dòng gồm các dấu gạch dưới, xem Output mẫu để rõ hơn.

Sample Input 0

```
6
8s312
s1a8k
690868516564
08799297
14000000
s7dw2
1d17c
438356058671
17847396
2000000
ir188
ac8ak
001206584176
30339544
37000000
7c1zi
lak2l
413371350595
69057560
66000000
s2w88
icdkd
652652106407
67576215
63000000
1l3a3
12sb3
227430161615
76168654
85000000
8
deposit
227430161615 95000000
withdraw
652652106407 26000000
transfer
001206584176 652652106407 89000000
withdraw
413371350595 67000000
transfer
413371350595 227430161615 12000000
withdraw
```

```
001206584176 98000000
deposit
227430161615 17000000
transfer
001206584176 227430161615 54000000
```

Sample Output 0

```
ID : 8s312
CusID : s1a8k
Number : 690868516564
PIN : 08799297
Balance : 14000000VND
--------------------
ID : s7dw2
CusID : 1d17c
Number : 438356058671
PIN : 17847396
Balance : 2000000VND
--------------------
ID : ir188
CusID : ac8ak
Number : 001206584176
PIN : 30339544
Balance : 37000000VND
--------------------
ID : 7c1zi
CusID : lak2l
Number : 413371350595
PIN : 69057560
Balance : 54000000VND
--------------------
ID : s2w88
CusID : icdkd
Number : 652652106407
PIN : 67576215
Balance : 37000000VND
--------------------
ID : 1l3a3
CusID : 12sb3
Number : 227430161615
PIN : 76168654
Balance : 209000000VND
--------------------
```

[OOP].3. Giải Cứu

- **Problem**
- Submissions
- Discussions

Để cứu công chúa BraveQ, hoàng tử Lừa phải trải qua các cuộc giao đấu với ác quỷ, nhân vật hoàng tử có các thông tin : - Power : Chỉ số sức mạnh

- Blood : Chỉ số máu
- Alive : Mô tả nhân vật còn sống hay đã chết

Các sự kiện mà nhân vật hoàng tử Lừa có thể gặp phải. Nếu nhân vật có máu <= 0 hoặc sức mạnh <= 0 thì nhân vật sẽ bị chết

1. Gặp nấm độc (mushroom) : Máu giảm đi 15, Sức mạnh giảm 2
2. Gặp phù thùy (witch) : Hai bên giao đấu và nếu sức mạnh của phù thủy lớn hơn hoặc bằng chỉ số sức mạnh của hoàng tử thì hoàng tử sẽ thua cuộc và bị chết, ngược lại nếu thắng chỉ số sức mạnh tăng thêm 5
3. Gặp cây đậu thần (pea) : Hoàng tử sẽ ăn cây đậu thần và chỉ số máu được cộng thêm 10, chỉ số sức mạnh tăng thêm 2
4. Gặp quân lính (soldier) : Hai bên giao đấu và nếu quân lính có sức mạnh lớn hơn hoặc bằng chỉ số sức mạnh của hoàng tử thì hoàng tử thua cuộc và bị chết ngay lập tức, ngược lại nếu đánh thắng quân lính thì chỉ số máu tăng thêm 5 và chỉ số sức mạnh tăng thêm 7.

```
import java.util.Date;
import java.util.Scanner;

public class Main {
   public static void main(String[] args) {
      Scanner sc = new Scanner(System.in);
      int power = Integer.parseInt(sc.nextLine().substring(8));
      int blood = Integer.parseInt(sc.nextLine().substring(8));
      String tmp = sc.nextLine();
      boolean alive = false;
      if(tmp.equals("ALIVE")) alive = true;
      NhanVat nv = new NhanVat(power, blood, alive);
      int suKien = sc.nextInt(); sc.nextLine();
      for(int i = 0; i < suKien; i++){
         String s = sc.nextLine();
         if(s.equals("pea")){
            nv.pea();
         }
         else if(s.equals("mushroom")){
            nv.mushroom();
```

```
        }
        else if(s.charAt(0) == 's'){
            int power1 = Integer.parseInt(s.substring(8));
            nv.soldier(power1);
        }
        else{
            int power1 = Integer.parseInt(s.substring(6));
            nv.witch(power1);
        }
        System.out.print(nv);
    }
  }
}


class NhanVat {

  private int power, blood;
  private boolean alive;

  public NhanVat(int power, int blood, boolean alive) {
    this.power = power;
    this.blood = blood;
    this.alive = alive;
  }

  public String toString() {
    String res = "";
    if (this.alive) {
      res = "ALIVE";
    } else {
      res = "DEAD";
    }
    return "POWER : " + this.power + "\nBLOOD : " + this.blood + "\n" + res + "\n--------------------\n";
  }

  public void mushroom() {
    if (this.alive) {
      this.power -= 2;
      this.blood -= 15;
      if (this.power <= 0 || this.blood <= 0) {
        this.power = 0;
        this.blood = 0;
        this.alive = false;
      }
    }
  }

  public void witch(int power) {
    if (this.alive) {
      if (power >= this.power) {
        this.power = 0;
        this.blood = 0;
```

```
            this.alive = false;
          } else {
            this.power += 5;
          }
        }
    }

    public void pea() {
      if (this.alive) {
        this.power += 2;
        this.blood += 10;
      }
    }

    public void soldier(int power) {
      if (this.alive) {
        if (power >= this.power) {
          this.power = 0;
          this.blood = 0;
          this.alive = false;
        } else {
          this.power += 7;
          this.blood += 5;
        }
      }
    }
}
```

Input Format

- Dòng 1 gồm thông tin của hoàng tử lừa bao gồm chỉ số sức mạnh, chỉ số máu, tình trạng ban đầu của nhân vật này là còn sống
- Dòng thứ 2 là N : Số lượng sự kiện mà hoàng tử Lừa gặp phải trên đường đi giải cứu công chúa
- N dòng tiếp theo mô tả sự kiện, nếu là sự kiện 2 và 4 thì có thêm thông tin chỉ số sức mạnh của phù thủy và quân lính

Constraints

- 1<=N<=1000

Output Format

- Đối với mỗi sự kiện bạn hãy in ra trạng thái của nhân vật hoàng tử, trong đó nếu nhân vật chết thì in DEAD, ngược lại còn sống thì in ALIVE. Chú ý nếu trong 1 sự kiện nào đó hoàng tử bị chết thì các sự kiện tiếp theo đó coi như không được thực hiện.

Sample Input 0

```
POWER : 100
```

```
BLOOD : 100
ALIVE
6
witch 77
mushroom
pea
pea
soldier 164
soldier 137
```

Sample Output 0

```
POWER : 105
BLOOD : 100
ALIVE
--------------------
POWER : 103
BLOOD : 85
ALIVE
--------------------
POWER : 105
BLOOD : 95
ALIVE
--------------------
POWER : 107
BLOOD : 105
ALIVE
--------------------
POWER : 0
BLOOD : 0
DEAD
--------------------
POWER : 0
BLOOD : 0
DEAD
--------------------
```

[OOP].4. Sunday League

- **Problem**
- Submissions
- Discussions

Một đội bóng tại giải bóng đá ngày Chủ Nhật gồm những thông tin :

- Mã đội bóng
- Tên đội bóng

- Số trận đã đấu
- Điểm số
- Hiệu số thắng thua

Bạn được cung cấp bảng xếp hạng hiện thời của của giải đấu và một loạt các trận đấu tại vòng hiện tại, nhiệm vụ của bạn là hãy tính toán điểm số, hiệu số và cập nhật lại BXH của giải đấu. Biết rằng có 20 đội bóng trong giải đấu này. Sắp xếp thứ hạng đội bóng thứ tự điểm, hiệu số thắng thua, mã đội bóng theo thứ tự tăng dần về từ điểm

Input Format

- Mỗi thông tin của 1 đội bóng gồm 4 dòng
- Dòng 1 : Vị trí, Dòng 2 : Mã đội bóng, Dòng 3 : tên đội bóng, dòng 4 lần lượt là số trận đã đấu, hiệu số bàn thắng thua và điểm số.
- Thông tin mỗi đội bóng được phân cách nhau một dòng các dấu gạch dưới.
- 10 dòng tiếp theo có dạng : X Y - Z T mang ý nghĩa đội X đấu với đội T, tỷ số là Y - Z

Constraints

N/A

Output Format

- In ra bảng xếp hạng sau khi cập nhật kết quả của vòng đấu này.

Sample Input 0

```
#1
ARS
Arsenal
29 43 72
------------------
#2
MC
Manchester City
28 45 64
------------------
#3
NEW
Newcastle
27 22 50
------------------
#4
TOT
Tottenham
```

29 12 50

#5
MU
Manchester United
27 4 50

#6
BRI
Brighton
27 17 46

#7
ASL
Aston Villa
29 -1 44

#8
LIV
Liverpool
28 15 43

#9
BREN
Brentford
28 9 43

#10
FUL
Fullham
28 0 39

#11
CHE
Chelsea
29 -1 39

#12
CRY
Crystal Palace
29 -15 30

#13
LEED
Leeds United
29 -11 29

#14
WOL
Wolves
29 -19 28

#15

```
WEST
West Ham
27 -9 27
------------------
#16
EVE
Everton
29 -18 27
------------------
#17
FOR
Forest
29 -28 27
------------------
#18
BOU
Bournemouth
29 -30 27
------------------
#19
LEI
Leicester
29 -11 25
------------------
#20
SOU
Southampton
29 -24 23
------------------
Arsenal 1 - 0 Chelsea
Manchester City 0 - 2 Crystal Palace
Newcastle 0 - 3 Leeds United
Tottenham 4 - 4 Wolves
Manchester United 3 - 3 West Ham
Brighton 4 - 4 Everton
Aston Villa 2 - 1 Forest
Liverpool 3 - 1 Bournemouth
Brentford 2 - 3 Leicester
Fullham 2 - 2 Southampton
```

Sample Output 0

```
#1 ARS Arsenal 30 44 75
------------------
#2 MC Manchester City 29 43 64
------------------
#3 TOT Tottenham 30 12 51
------------------
#4 MU Manchester United 28 4 51
------------------
#5 NEW Newcastle 28 19 50
------------------
#6 BRI Brighton 28 17 47
------------------
```

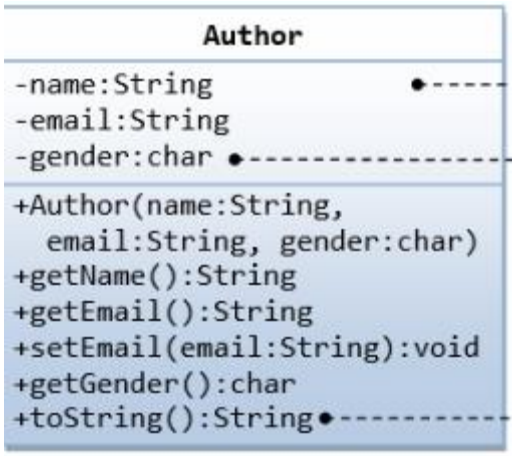
```
#7 ASL Aston Villa 30 0 47
------------------
#8 LIV Liverpool 29 17 46
------------------
#9 BREN Brentford 29 8 43
------------------
#10 FUL Fullham 29 0 40
------------------
#11 CHE Chelsea 30 -2 39
------------------
#12 CRY Crystal Palace 30 -13 33
------------------
#13 LEED Leeds United 30 -8 32
------------------
#14 WOL Wolves 30 -19 29
------------------
#15 WEST West Ham 28 -9 28
------------------
#16 LEI Leicester 30 -10 28
------------------
#17 EVE Everton 30 -18 28
------------------
#18 FOR Forest 30 -29 27
------------------
#19 BOU Bournemouth 30 -32 27
------------------
#20 SOU Southampton 30 -24 24
------------------
```

[OOP].5. Book and Author Composition

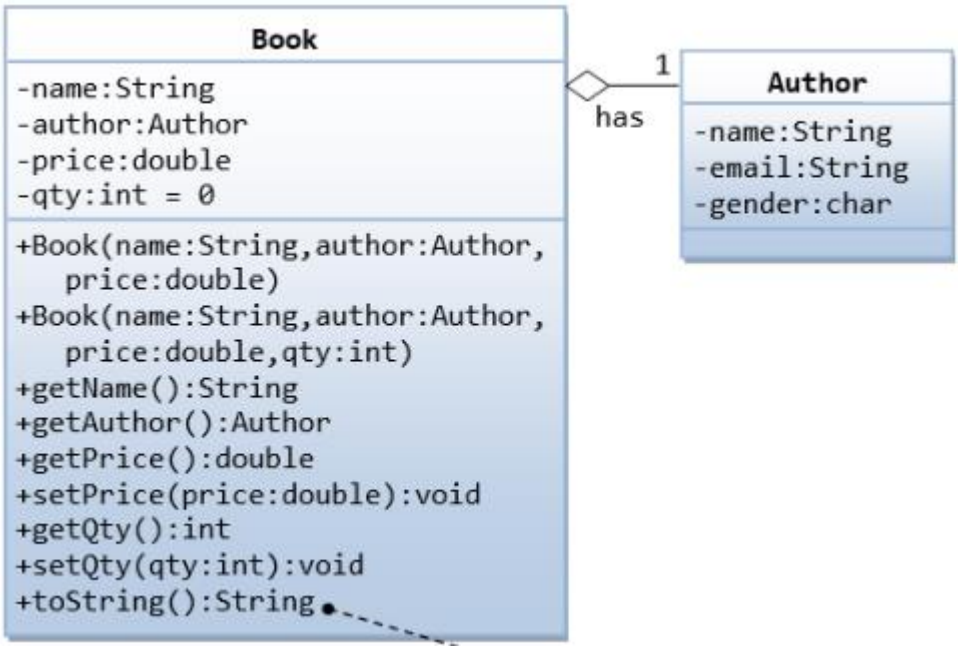
- **Problem**
- Submissions
- Discussions

Xây dựng lớp Author và lớp Book theo bản thiết kế sau

- Lớp Author



- Lớp Book và mối quan hệ



Cho thông tin cuốn sách và tác giả tương ứng, bạn hãy sắp xếp các cuốn sách và in ra theo thứ tự giá bán giảm dần, nếu cùng giá bán thì sắp xếp theo thứ tự từ điển tên sách tăng dần, khi in cần in cả thông tin tác giả.

Input Format

- Dòng 1 là N : số lượng cuốn sách, mỗi cuốn sách gồm 6 dòng
- Thông tin sách gồm 3 dòng : Dòng 1 : Tên, dòng 2 : giá, dòng 3 : số lượng
- Thông tin tác giả của cuốn sách gồm 3 dòng : Tên, email, giới tính

Constraints

- 1<=N<=1000

Output Format

In ra thông tin của từng cuốn sách theo mẫu, giá tiền in ra số nguyên gần nhất với giá bán, xem ví dụ mẫu để rõ hơn

Sample Input 0

```
3
Song
800000
5000
Xuan Quynh
xuanquynh@gmail.com
M
Ha Do
400000
6000
Nguyen Nhat Anh
nhatanh@gmail.com
M
To Kill a Mockingbird
100000
15000
Harper Lee
lee@gmail.com
F
```

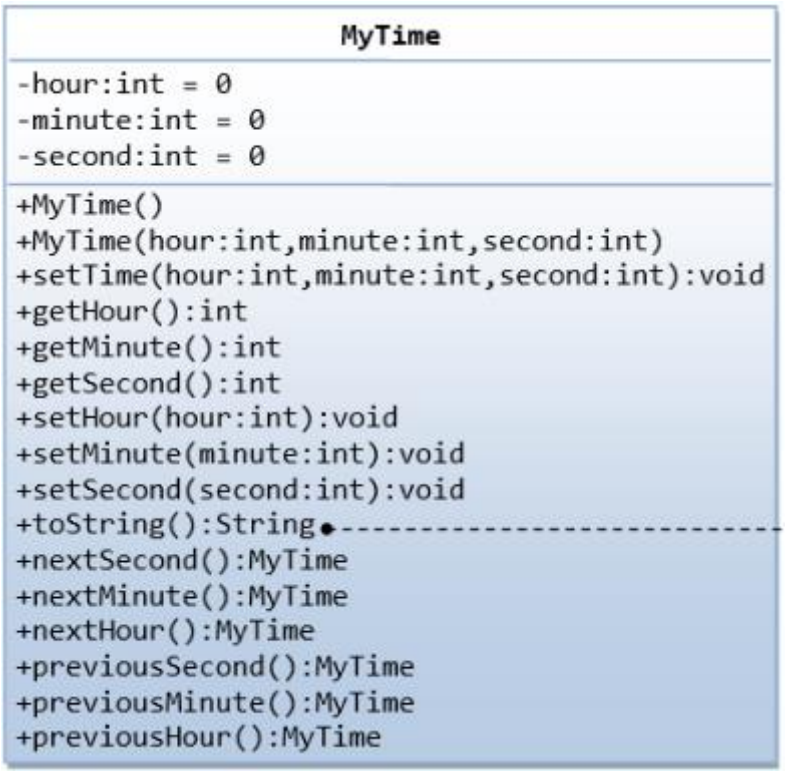
Sample Output 0

```
Book Details :
Song
800000
5000
Author Information :
Xuan Quynh
xuanquynh@gmail.com
M
--------------------
Book Details :
Ha Do
400000
6000
Author Information :
Nguyen Nhat Anh
nhatanh@gmail.com
M
--------------------
Book Details :
To Kill a Mockingbird
100000
15000
Author Information :
Harper Lee
lee@gmail.com
F
--------------------
```

[OOP].6. MyTime Class

- **Problem**
- Submissions
- Discussions

Cho thiết kế lớp MyTime như sau :



Cho thông tin thời gian hiện tại, bạn hãy sử dụng 6 hàm đã xây dựng trong lớp MyTime để in ra :

- Thời gian kế tiếp sau thời gian hiện tại 1s, 1 phút, 1 giờ
- Thời gian trước thời gian hiện tại 1s, 1 phút, 1 giờ

Input Format

- Dòng duy nhất mô tả thời gian dạng : hh:mm:ss

Constraints

- 0<=hh<=23
- 0<=mm<=59

- 0<=ss<=59

Output Format

In ra lần lượt 6 dòng thời gian kết quả định dạng : hh:mm:ss

Sample Input 0

```
23:00:37
```

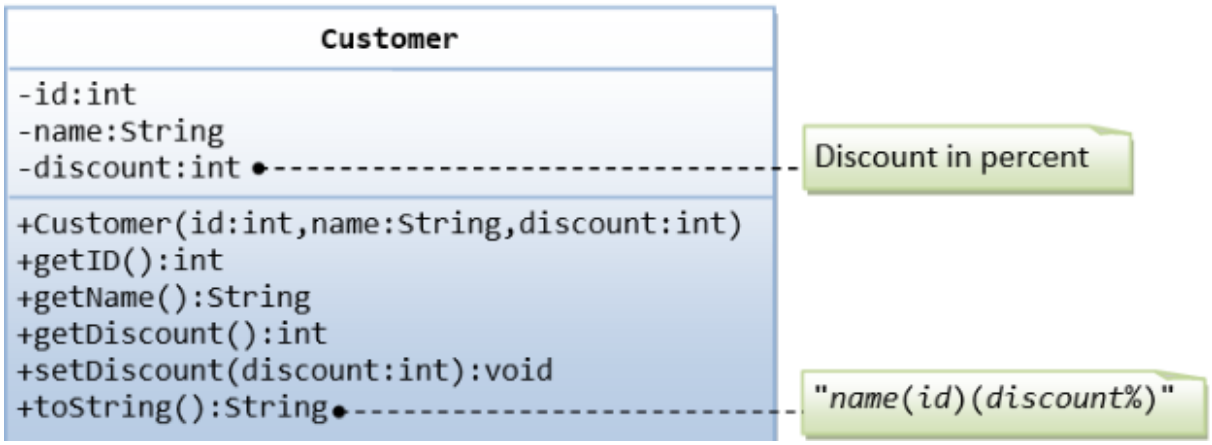
Sample Output 0

```
23:00:38
23:01:37
00:00:37
23:00:36
22:59:37
22:00:37
```

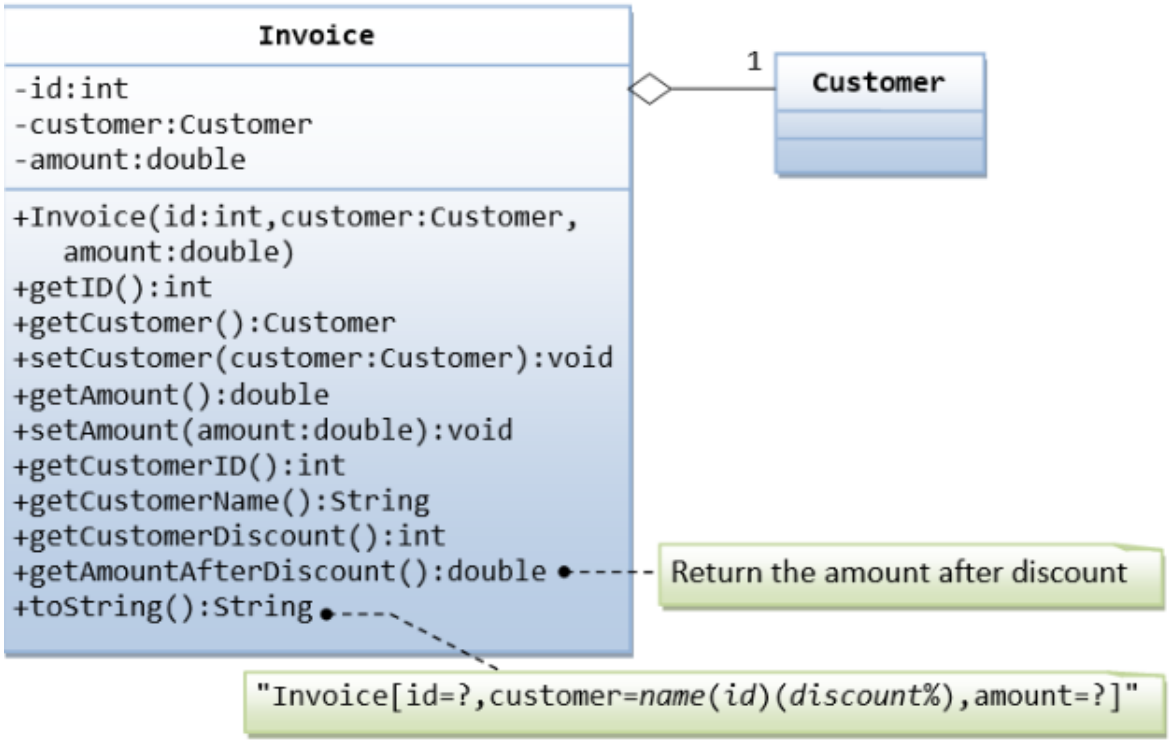
[OOP].7. Customer And Invoice

- **Problem**
- Submissions
- Discussions

- Cho lớp Customer :



- Lớp Invoice và mối quan hệ giữa lớp Customer và Invoice



Bạn được cung cấp 1 loạt khách hàng cùng hóa đơn của họ, nhiệm vụ của bạn là tính toán số tiền cần phải trả sau khi áp dụng giảm giá. In ra danh sách khách hàng và hóa đơn của họ theo thứ tự tiền phải trả giảm dần. Nếu 2 khách hàng cùng số tiền hóa đơn thì sắp theo ID khách hàng tăng dần.

```
import java.io.*;
import java.util.*;

class Customer{
   private int id;
   private String name;
   private int discount;

   public Customer(int id, String name, int discount) {
      this.id = id;
      this.name = name;
      this.discount = discount;
   }

   public int getId() {
      return id;
   }

   public String getName() {
      return name;
   }

   public int getDiscount() {
      return discount;
   }
```

```
    public void setId(int id) {
        this.id = id;
    }

    public void setName(String name) {
        this.name = name;
    }

    public void setDiscount(int discount) {
        this.discount = discount;
    }

    public String toString(){
        return "Customer ID : "+this.id+"\n"+"Full Name : "+this.name;
    }
}
class Invoice{
    private Customer customer;
    private int idInvoice;
    private double amount;

    public Invoice(Customer customer, int idInvoice, double amount) {
        this.customer = customer;
        this.idInvoice = idInvoice;
        this.amount = amount;
    }

    public Customer getCustomer() {
        return customer;
    }

    public int getIdInvoice() {
        return idInvoice;
    }

    public double getAmount() {
        return amount;
    }

    public void setCustomer(Customer customer) {
        this.customer = customer;
    }

    public void setIdInvoice(int idInvoice) {
        this.idInvoice = idInvoice;
    }

    public void setAmount() {
        this.amount = this.amount - this.amount*customer.getDiscount()/100;
    }

    public String toString(){
```

```
    return customer.toString()+"\n"+ "Amount : "+String.format("%.2f", this.amount)+" $" +"\n"+"---
----------";
  }
}
public class Main{
  public static void main(String[] args) {
    Scanner sc = new Scanner(System.in);
    int n = sc.nextInt();
    sc.nextLine();
    ArrayList<Invoice> arr = new ArrayList<>();
    for(int i=1;i<=n;i++){
      sc.nextLine();
      String cusID[] = sc.nextLine().split("\\s+");
      String name[] = sc.nextLine().split("\\s+");
      String dis[] = sc.nextLine().split("\\s+");
      String invoiID[] = sc.nextLine().split("\\s+");
      String amount[] = sc.nextLine().split("\\s+");

      String fullName = "";
      for(int j=3;j<name.length;j++){
        fullName += name[j]+" ";
      }
      fullName = fullName.substring(0,fullName.length()-1);
      Customer tmp = new Customer(Integer.parseInt(cusID[cusID.length-
1]),fullName,Integer.parseInt(dis[dis.length-1]));
      Invoice ans = new Invoice(tmp,Integer.parseInt(invoiID[invoiID.length-
1]),Double.parseDouble(amount[amount.length-2]));
      ans.setAmount();
      arr.add(ans);
    }
    Collections.sort(arr,new Comparator<Invoice>(){
      @Override
      public int compare(Invoice o1, Invoice o2) {
        if(o1.getAmount()==o2.getAmount()){
          if(o1.getCustomer().getId()<o2.getCustomer().getId())
            return -1;
          else
            return 1;
        }
        else if(o1.getAmount()>o2.getAmount())
          return -1;
        else
          return 1;
      }

    });
    for(Invoice tmp:arr){
      System.out.println(tmp);
    }
  }
}
```

Input Format

- Dòng 1 là N : số lượng khách hàng
- Mỗi khách hàng được mô tả thông tin thông qua 5 dòng : ID, Name, Discount, Invoice ID, Amount (Số tiền của hóa đơn), Mỗi khách hàng được phân cách nhau bởi 1 dòng các dấu gạch dưới

Constraints

- 1<=N<=1000

Output Format

In ra danh sách khách hàng gồm ID khách hàng, tên khách hàng và số tiền của hóa đơn theo mẫu. Số tiền được in ra lấy 2 số sau dấu thập phân. Mỗi khách hàng được phân cách nhau bởi 1 dòng các dấu gạch dưới

Sample Input 0

```
5
--------------------
Customer ID : 100
Full Name : Wayne Rooney
Discount : 6
Invoice ID : 1355
Amount : 4472.00 $
--------------------
Customer ID : 101
Full Name : Peter Cech
Discount : 7
Invoice ID : 1525
Amount : 4690.00 $
--------------------
Customer ID : 102
Full Name : Andrew Tate
Discount : 16
Invoice ID : 1607
Amount : 722.00 $
--------------------
Customer ID : 103
Full Name : Ryan Giggs
Discount : 16
Invoice ID : 1714
Amount : 4524.00 $
--------------------
Customer ID : 104
Full Name : Thomas Tuchel
Discount : 4
Invoice ID : 1280
Amount : 805.00 $
--------------------
```

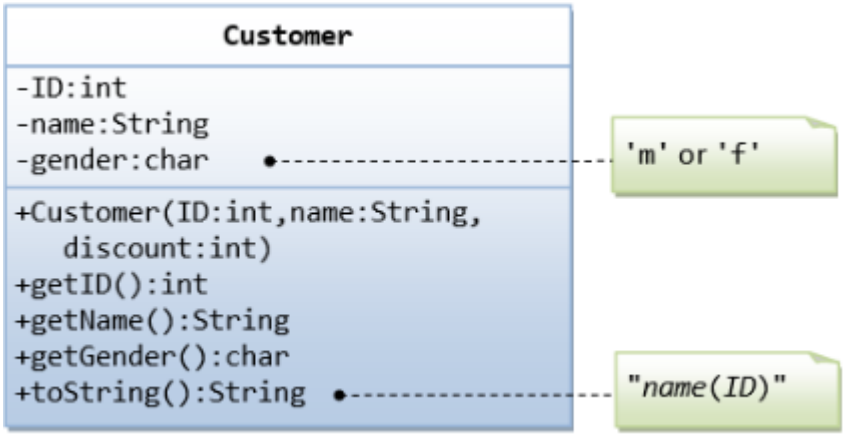
Sample Output 0

```
Customer ID : 101
Full Name : Peter Cech
Amount : 4361.70 $
--------------------
Customer ID : 100
Full Name : Wayne Rooney
Amount : 4203.68 $
--------------------
Customer ID : 103
Full Name : Ryan Giggs
Amount : 3800.16 $
--------------------
Customer ID : 104
Full Name : Thomas Tuchel
Amount : 772.80 $
--------------------
Customer ID : 102
Full Name : Andrew Tate
Amount : 606.48 $
--------------------
```

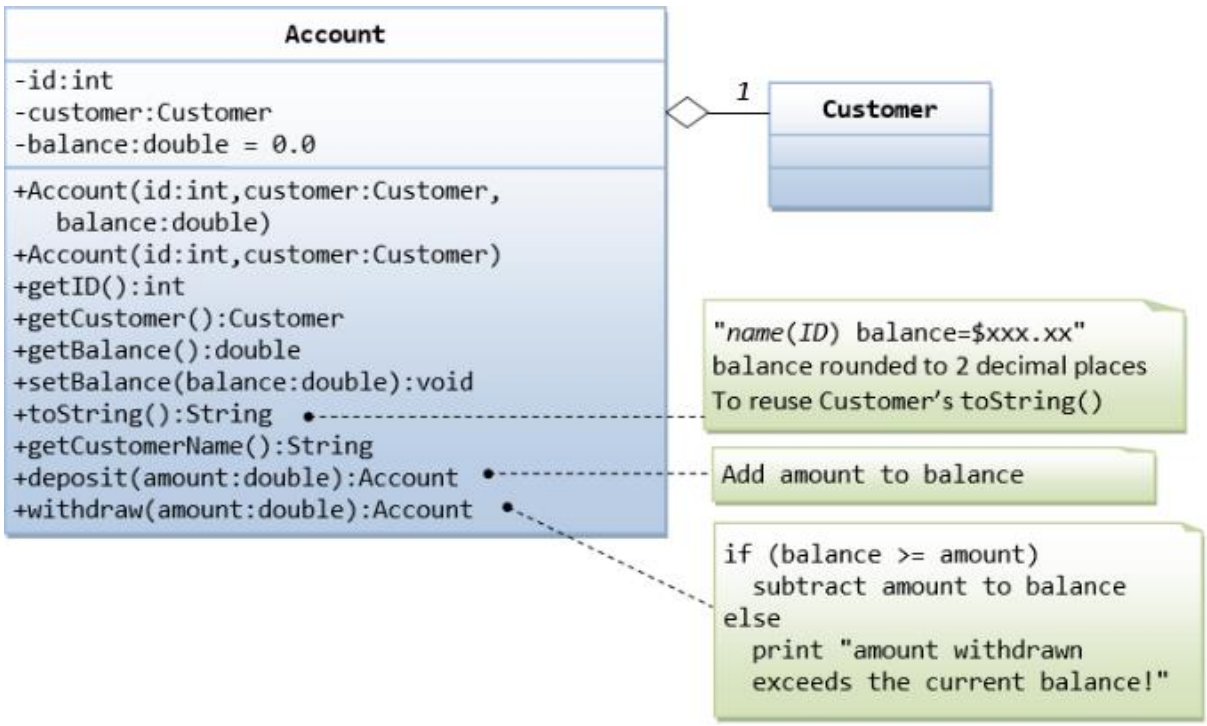
[OOP].8. Customer And Account

- **Problem**
- Submissions
- Discussions

- Cho lớp Customer :



- Lớp Account và mối quan hệ với lớp Customer :



Bạn được cung cấp thông tin khách hàng và tài khoản của họ, tiếp theo đó là một loạt các thao tác nạp và rút tiền. Nhiệm vụ của bạn là in ra thông tin tài khoản của khách hàng sau khi thực hiện 1 loạt các thao tác trên.

```
import java.util.*;
import java.util.Map.Entry;

class Account_Bank {
   private int id;
   private Customer_Bank Customer;
   private double balance=0.0;


   public Account_Bank(int id, Customer_Bank Customer, double balance) {
      this.id = id;
      this.Customer = Customer;
      this.balance = balance;
   }


   public Account_Bank(int id, Customer_Bank Customer) {
      this.id = id;
      this.Customer = Customer;
   }


   public int getId() {
      return this.id;
   }

   public void setId(int id) {
      this.id = id;
```

```
  }

  public Customer_Bank getCustomer() {
    return this.Customer;
  }

  public void setCustomer(Customer_Bank Customer) {
    this.Customer = Customer;
  }

  public double getBalance() {
    return this.balance;
  }

  public void setBalance(double balance) {
    this.balance = balance;
  }

  public String getCustomerName(){
    return this.getCustomer().getName();
  }

  public Account_Bank deposit(double amount ){
    Account_Bank tmp = new Account_Bank(this.getId(),this.getCustomer(),this.getBalance());
    tmp.setBalance(tmp.getBalance()+amount);
    System.out.println("transaction successful");
    return tmp;
  }

  public Account_Bank withdraw(double amount){
    Account_Bank tmp = new Account_Bank(this.getId(),this.getCustomer(),this.getBalance());
    if(tmp.getBalance()>=amount){
      System.out.println("transaction successful");
      tmp.setBalance(tmp.getBalance()-amount);

    }
    else{
      System.out.println( "amount withdrawn exceeds the current balance!");
    }
    return tmp;
  }


  @Override
  public String toString() {
    return ("Account ID : "+this.getId()+"\n"+
    "Balance : "+String.format("%.2f",this.getBalance())+" $");
  }



}
```

```
class Customer_Bank {
    private int ID;
    private String name;
    private char gender;


    public Customer_Bank(int ID, String name, char gender) {
        this.ID = ID;
        this.name = name;
        this.gender = gender;
    }


    public int getID() {
        return this.ID;
    }

    public String getName() {
        return this.name;
    }

    public char getGender() {
        return this.gender;
    }


    @Override
    public String toString() {
        return ("Customer ID : "+this.getID()+"\n"+
        "Full Name : "+this.getName()+"\n"+
        "Gender : "+this.getGender()+"\n");
    }


}


public class vidu_001{
    public static void main(String[] args) {
        Scanner sc = new Scanner(System.in);
        String CustomerID = sc.nextLine().substring(14);
        String FullName = sc.nextLine().substring(12);
        char gender = sc.nextLine().charAt(9);
        String AccountID = sc.nextLine().substring(13);
        String Balance1 = sc.nextLine();
        String Balance = Balance1.substring(10,Balance1.length()-2);
        Customer_Bank cb = new Customer_Bank(Integer.parseInt(CustomerID), FullName, gender);
        Account_Bank ab = new Account_Bank(Integer.parseInt(AccountID),
cb,Double.parseDouble(Balance));
        sc.nextLine();
        int n = sc.nextInt();
```

```
    sc.nextLine();
    while(n>0){

       String step = sc.nextLine();
       String [] all = step.split("\\s+");
       if(all[0].equals("withdraw")){
          ab = ab.withdraw(Double.parseDouble(all[1]));
       }

       if(all[0].equals("deposit")){
          ab = ab.deposit(Double.parseDouble(all[1]));
       }

       n--;

    }

    System.out.println("--------------------");
    System.out.println(cb.toString()+ab.toString());

  }


}
```

Input Format

- Dòng 1 : ID khách hàng
- Dòng 2 : Tên khách hàng
- Dòng 3 : Giới tính (M hoặc F)
- Dòng 4 : ID tài khoản
- Dòng 5 : Số dư ban đầu
- Dòng 6 là N : Số lượng giao dịch nạp và rút
- N dòng tiếp theo có dạng : X Y trong đó X là deposit hoặc withdraw và Y là số tiền nạp rút tương ứng. Với giao dịch rút tiền, in ra thông báo như trong mẫu thiết kế lớp.

Constraints

- 1<=N<=1000

Output Format

- Đối với mỗi giao dịch in ra "transaction successful" nếu việc nạp rút thành công, nếu việc rút không thành công thì in ra dòng thông báo "amount withdrawn exceeds the current balance!"
- Cuối cùng in ra thông tin tài khoản sau khi thực hiện N giao dịch

Sample Input 0

```
Customer ID : 28
Full Name : Elon Musk
Gender : M
Account ID : 886123
Balance : 23000.00 $
--------------------
6
withdraw 8680
withdraw 2379
deposit 14547
deposit 19205
deposit 29487
withdraw 3818
```

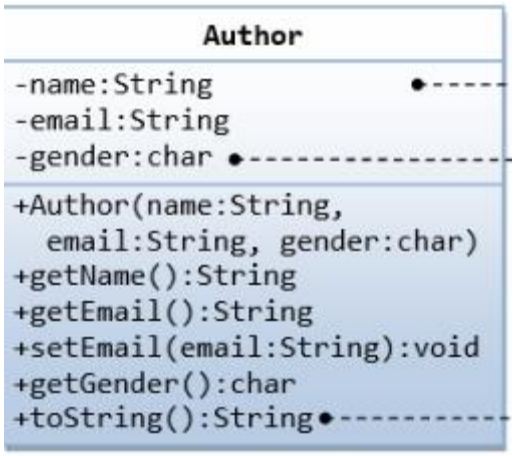
Sample Output 0

```
transaction successful
transaction successful
transaction successful
transaction successful
transaction successful
transaction successful
--------------------
Customer ID : 28
Full Name : Elon Musk
Gender : M
Account ID : 886123
Balance : 71362.00 $
```

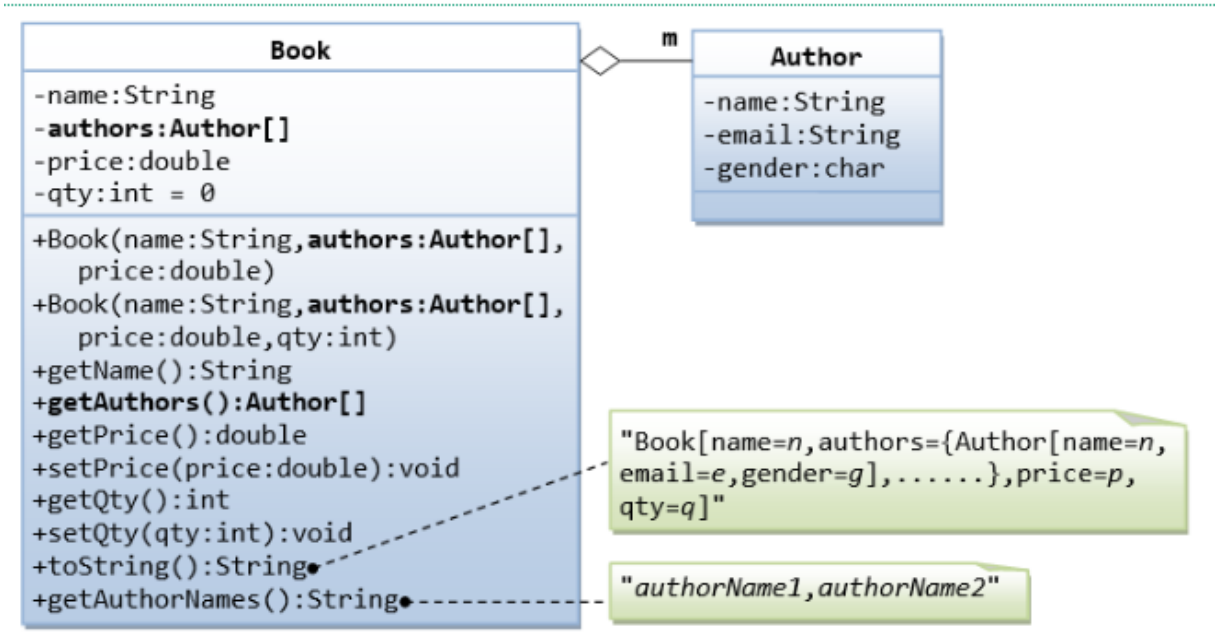
[OOP].9. Book and Authors

- **Problem**
- Submissions
- Discussions

- Cho lớp Author :



- Cho lớp Book và mối quan hệ với lớp Author



Cho danh sách sách các cuốn sách và tác giả của cuốn sách đó, 1 cuốn sách có thể có nhiều tác giả. Bạn hãy nhập danh sách này và sắp xếp danh sách sách theo thứ tự tên sách tăng dần về từ điển.

```
import java.io.*;
import java.util.*;

class Author{
   private String name,email;
   private Character gender;

   public Author(String name, String email, Character gender) {
      this.name = name;
      this.email = email;
      this.gender = gender;
   }

   public String getName() {
```

```
        return name;
    }

    public String getEmail() {
        return email;
    }

    public Character getGender() {
        return gender;
    }

    public void setName(String name) {
        this.name = name;
    }

    public void setEmail(String email) {
        this.email = email;
    }

    public void setGender(Character gender) {
        this.gender = gender;
    }

    public String toString(){
        return "Name : "+this.name+"\n"+"Email : "+this.email+"\n"+"Gender : "+this.gender;
    }
}
class Book {
    private String nameBook;
    private Author[] author;
    private double price;
    private int soTG;
    private int qty;

    public Book(String nameBook, double price,int qty,int soTG,Author[] author) {
        this.nameBook = nameBook;
        this.author = author;
        this.price = price;
        this.soTG = soTG;
        this.qty = qty;
    }

    public int getSoTG() {
        return soTG;
    }

    public String getNameBook() {
        return nameBook;
    }

    public Author[] getAuthor() {
        return author;
    }
```

```
    public double getPrice() {
        return price;
    }

    public int getQty() {
        return qty;
    }

    public void setNameBook(String nameBook) {
        this.nameBook = nameBook;
    }

    public void setAuthor(Author[] author) {
        this.author = author;
    }

    public void setPrice(double price) {
        this.price = price;
    }

    public void setQty(int qty) {
        this.qty = qty;
    }

    public void display(int n){
        System.out.println("------------------------");
        System.out.println("Book information :");
        System.out.println("Name : "+this.nameBook);
        System.out.println("Price : "+Math.round(this.price));
        System.out.println("Quantity : " + this.qty);
        System.out.println("Author information :");
        for(int i=0;i<n;i++){
            System.out.println("#"+(int)(i+1));
            System.out.println(author[i]);
        }

    }
}


public class Main{
    public static void main(String[] args) {
        Scanner sc = new Scanner(System.in);
        ArrayList<Book> arr = new ArrayList<>();
        int n = sc.nextInt();
        sc.nextLine();
        for(int i=1;i<=n;i++){
            sc.nextLine();
            String tenSach = sc.nextLine();
            double gia = sc.nextDouble();
            int cnt = sc.nextInt();
            int luongTacGia = sc.nextInt();
```

```
        sc.nextLine();
        Author b[] = new Author[luongTacGia];
        for(int j=0;j<luongTacGia;j++){
            String tenTacGia = sc.nextLine();
            String email = sc.nextLine();
            Character gt = sc.nextLine().charAt(0);
            Author tg = new Author(tenTacGia,email,gt);
            b[j] = tg;
        }
        Book tmp = new Book(tenSach,gia,cnt,luongTacGia,b);
        arr.add(tmp);
    }
    Collections.sort(arr,new Comparator<Book>(){
        @Override
        public int compare(Book o1, Book o2) {
            if(o1.getNameBook().compareTo(o2.getNameBook())<0)
                return -1;
            else
                return 1;
        }

    });
    for(Book tmp:arr){
        tmp.display(tmp.getSoTG());
    }
  }
}
```

Input Format

- Dòng 1 là N : Số lượng cuốn sách
- Mỗi cuốn sách được mô tả như sau :
- Dòng 1, 2, 3 : Tên, giá, số lượng
- Dòng thứ 4 là số lượng tác giả của cuốn sách
- Các dòng tiếp theo mô tả tác giả của cuốn sách : Tên, email, giới tính trên 3 dòng

Constraints

- 1<=N<=1000

Output Format

- In ra các cuốn sách và thông tin tác giả của cuốn sách, xem output mẫu để rõ hơn yêu cầu in thông tin

Sample Input 0

```
4
----------------------
```

```
A Brief History of Time
700000
2137
1
Andrew Neiman
Andrew Neiman.author@gmail.com
M
----------
Don Quixote
500000
2002
3
Thomas Che
Thomas Che.author@gmail.com
M
Lucas
Lucas.author@gmail.com
M
Happer Lee
Happer Lee.author@gmail.com
M
----------
One Hundred Years of Solitude
700000
1920
2
Feyman
Feyman.author@gmail.com
M
Tom Cruise
Tom Cruise.author@gmail.com
M
----------
The Great Gatsby
600000
1588
1
Kirk Wise
Kirk Wise.author@gmail.com
M
----------
```

Sample Output 0

```
----------
Book information :
Name : A Brief History of Time
Price : 700000
Quantity : 2137
Author information :
#1
Name : Andrew Neiman
Email : Andrew Neiman.author@gmail.com
Gender : M
```

```
----------
Book information :
Name : Don Quixote
Price : 500000
Quantity : 2002
Author information :
#1
Name : Thomas Che
Email : Thomas Che.author@gmail.com
Gender : M
#2
Name : Lucas
Email : Lucas.author@gmail.com
Gender : M
#3
Name : Happer Lee
Email : Happer Lee.author@gmail.com
Gender : M
----------
Book information :
Name : One Hundred Years of Solitude
Price : 700000
Quantity : 1920
Author information :
#1
Name : Feyman
Email : Feyman.author@gmail.com
Gender : M
#2
Name : Tom Cruise
Email : Tom Cruise.author@gmail.com
Gender : M
----------
Book information :
Name : The Great Gatsby
Price : 600000
Quantity : 1588
Author information :
#1
Name : Kirk Wise
Email : Kirk Wise.author@gmail.com
Gender : M
```

[OOP].10. Student And Subject (One to many relationship)

- **Problem**
- Submissions

- Discussions

Cho lớp MonHoc gồm các thuộc tính : Tên môn học, số tín chỉ, điểm số. Lớp Sinh viên gồm các thuộc tính : Mã sinh viên, họ tên, lớp, các môn học đã từng học qua.

Bạn hãy tiến hành tính điểm gpa cho từng sinh viên dựa vào danh sách những môn học mà sinh viên này đã từng học. Điểm gpa được tính theo công thức dưới đây :

$$\text{Semester GPA} = \frac{\text{Total Achieved Points (Credits Earned x Points)}}{\text{Total Credits Attempted of all subjects}}$$

Input Format

- Dòng 1 là N : số lượng sinh viên
- Thông tin của mỗi sinh viên được mô tả như sau : Dòng 1. Mã sinh viên, dòng 2 là họ tên, dòng 3 là lớp, dòng 4 là số lượng môn học đã học. Các dòng tiếp theo mô tả môn học, mỗi môn học gồm 3 dòng : Dòng 1 là tên môn học, dòng 2 là số tín chỉ và điểm số.

Constraints

- 1<=N<=1000

Output Format

- In ra danh sách sinh viên theo điểm gpa giảm dần, nếu 2 sinh viên có cùng điểm gpa thì sắp xếp theo mã sinh viên tăng dần

Sample Input 0

```
5
------------------------
SV1
Cedric Newton
IT1
6
C++
4 1.00
Java
4 1.00
C
3 1.00
DSA
2 3.00
```

Web
3 0.00
Mobile
3 0.00

SV2
Cedric Patterson
IT2
7
C++
3 3.00
Java
3 1.00
C
5 0.00
DSA
2 0.00
Web
4 1.00
Mobile
5 0.00
Machine Learning
3 2.00

SV3
Charlie Burns
IT2
3
C++
3 0.00
Java
3 4.00
C
2 4.00

SV4
Roberto Ellis
IT4
3
C++
5 0.00
Java
5 1.00
C
4 4.00

SV5
Mark Ellis
IT4
7
C++
5 3.00
Java

```
3 2.00
C
3 2.00
DSA
2 1.00
Web
2 2.00
Mobile
5 2.00
Machine Learning
5 1.00
```

Sample Output 0

```
SV3 Charlie Burns IT2 2.50
------------------------
SV5 Mark Ellis IT4 1.92
------------------------
SV4 Roberto Ellis IT4 1.50
------------------------
SV1 Cedric Newton IT1 0.89
------------------------
SV2 Cedric Patterson IT2 0.88
------------------------
```

Sample Input 1

```
5
------------------------
SV1
Mark Rodriguez
IT4
5
C++
5 3.00
Java
2 1.00
C
3 4.00
DSA
4 1.00
Web
4 4.00
------------------------
SV2
Milton Harris
IT1
3
C++
4 4.00
Java
2 2.00
C
5 4.00
------------------------
```

```
SV3
Harold Bennett
IT2
3
C++
4 3.00
Java
2 3.00
C
2 4.00
------------------------
SV4
Aaron Gross
IT1
4
C++
5 1.00
Java
2 0.00
C
2 3.00
DSA
5 4.00
------------------------
SV5
Stuart Stanley
IT1
4
C++
4 1.00
Java
5 1.00
C
4 4.00
DSA
4 4.00
```

Sample Output 1

```
SV2 Milton Harris IT1 3.64
------------------------
SV3 Harold Bennett IT2 3.25
------------------------
SV1 Mark Rodriguez IT4 2.72
------------------------
SV5 Stuart Stanley IT1 2.41
------------------------
SV4 Aaron Gross IT1 2.21
------------------------
```