**[OOP]. Bài 1. Lớp Sinh Viên**

Viết chương trình khai báo lớp Sinh Viên gồm các thông tin: Mã SV, Họ tên, Lớp, Ngày sinh và Điểm GPA (dạng số thực float). Hàm khởi tạo không có tham số, gán các giá trị thuộc tính ở trạng thái mặc định (xâu ký tự rỗng, giá trị số bằng 0). Đọc thông tin 1 sinh viên từ bàn phím (không có mã sinh viên) và in ra màn hình. Trong đó Mã SV được gán là SV001. Ngày sinh được chuẩn hóa về dạng dd/mm/yyyy.

**Input Format**

Gồm 4 dòng : 1. Họ tên không quá 30 chữ cái.

1. Lớp là một chuỗi không có dấu cách
2. Ngày sinh có đủ 3 phần ngày tháng năm nhưng có thể chưa đúng chuẩn dd/mm/yyyy.
3. Điểm GPA đảm bảo trong thang điểm 4 với 2 nhiều nhất 2 số sau dấu phẩy.

**Constraints**

N/A

**Output Format**

Ghi thông tin sinh viên trên 1 dòng, mỗi thông tin cách nhau 1 khoảng trống, gpa lấy 1 chữ số sau dấu phẩy

**Sample Input 0**

Nguyen Van Nam

CNTT1

16/8/2001

2.8

**Sample Output 0**

SV001 Nguyen Van Nam CNTT1 16/08/2001 2.8

**[OOP]. Bài 2. Lớp nhân viên**

Một nhân viên làm việc trong công ty được lưu lại các thông tin sau:

1. Mã nhân viên: được gán giá trị là 00001
2. Họ tên: Xâu ký tự không quá 50 chữ cái.
3. Giới tính: Nam hoặc Nu
4. Ngày sinh: đúng theo chuẩn dd/mm/yyyy
5. Địa chỉ: Xâu ký tự không quá 100 chữ cái
6. Mã số thuế: Dãy số có đúng 10 chữ số
7. Ngày ký hợp đồng: đúng theo chuẩn dd/mm/yyyy

Viết chương trình nhập một nhân viên (không nhập mã) in ra màn hình thông tin của nhân viên đó.

**Input Format**

Gồm 6 dòng lần lượt ghi các thông tin theo thứ tự đã ghi trong đề bài. Không có mã nhân viên.

**Constraints**

N/A

**Output Format**

Ghi ra đầy đủ thông tin nhân viên trên một dòng, các thông tin cách nhau đúng một khoảng trống.

**Sample Input 0**

Nguyen Thi Nhung

Nu

28/09/1999

Ha Nam

8850288467

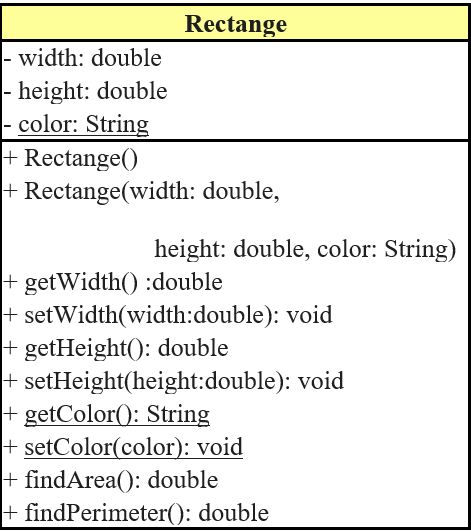
13/07/2021

**Sample Output 0**

00001 Nguyen Thi Nhung Nu 28/09/1999 Ha Nam 8850288467 13/07/2021

**[OOP]. Bài 3. Lớp HCN**

Viết chương trình khai báo lớp Rectange với các thuộc tính và phương thức như sau:



Viết chương trình nhập vào giá trị độ dài hai cạnh của hình chữ nhật và màu sắc. In ra thông tin về chu vi, diện tích và màu sắc (đã đưa về dạng chuẩn trong đó ký tự đầu viết hoa, các ký tự sau viết thường) của hình chữ nhật đó.

**Input Format**

Gồm 2 số nguyên a, b là độ dài 2 cạnh hình chữ nhật và một xâu ký tự (không có khoảng trống) mô tả màu sắc.

**Constraints**

-1000<=a,b<=1000;

**Output Format**

Nếu hình chữ nhật là hợp lệ (các cạnh đều nguyên dương) thì in ra 3 thông tin: chu vi, diện tích, màu sắc, mỗi thông tin cách nhau một khoảng trống. Nếu dữ liệu không hợp lệ in ra INVALID

**Sample Input 0**

22 25 Gray

**Sample Output 0**

94 550 Gray

**[OOP]. Bài 4. Thu nhập của giáo viên**

Trường phổ thông XYZ tính lương giáo viên theo quy tắc sau: Mỗi giáo viên có mã ngạch gồm 4 ký tự trong đó. 2 ký tự đầu là chức vụ (HT: Giáo viên kiêm nhiệm Hiệu trưởng, HP: Giáo viên kiêm nhiệm Hiệu phó, GV: Giáo viên thường). 2 ký tự số cuối cùng cho biết hệ số bậc lương (không quá 20). Lương cơ bản của mỗi giáo viên cũng có thể khác nhau. Phụ cấp quy định như sau. HT: 2.000.000. HP: 900.000. GV: 500.000. Thu nhập được tính bằng lương cơ bản nhân với hệ số bậc lương cộng thêm phụ cấp. Hãy tính lương cho 1 giáo viên theo quy tắc trên.

**Input Format**

Có 3 dòng lần lượt là mã ngạch, họ tên và lương cơ bản.

**Constraints**

Lương cơ bản là số nguyên không quá 10^7

**Output Format**

Chỉ có một dòng ghi lần lượt các thông tin: mã ngạch, họ tên, bậc lương, thu nhập.

**Sample Input 0**

HP05

Nguyen Van Ha

1748000

**Sample Output 0**

HP05 Nguyen Van Ha 5 9640000

**[OOP]. Bài 5. Tính lương nhân viên**

Thông tin về nhân viên bao gồm: Mã nhân viên (tự động tăng theo thứ tự nhập, tính từ NV01, nếu chỉ có 1 nhân viên thì mã là NV01) - Họ và tên - Lương cơ bản mỗi ngày công - Số ngày công - Chức vụ Tiền lương được tính bằng lương cơ bản nhân với số ngày công. Giả sử quy tắc tính tiền thưởng như sau:

* Số ngày công từ 25 trở lên thì thưởng 20% tiền lương
* Số ngày công từ 22 đến dưới 25 thì thưởng 10% tiền lương
* Dưới 22 ngày công thì không có thưởng. Mỗi nhân viên có thể có thêm phụ cấp chức vụ:
* GD: 250000
* PGD: 200000
* TP: 180000
* NV: 150000 Hãy nhập thông tin 1 nhân viên và tính toán thu nhập theo quy tắc trên.

**Input Format**

Gồm 4 dòng lần lượt ghi Họ tên, lương cơ bản, số ngày công và chức vụ; Số ngày công là số nguyên không âm không vượt quá 31;

**Constraints**

Lương cơ bản là số nguyên dương không quá 10^7;

**Output Format**

Ghi ra một dòng gồm: mã nhân viên, tên nhân viên, lương tháng, thưởng, phụ cấp và thu nhập. Mỗi thông tin cách nhau một khoảng trống.

**Sample Input 0**

Nguyen Van Teo

26000

21

PGD

**Sample Output 0**

NV01 Nguyen Van Teo 546000 0 200000 746000

**[OOP]. Bài 6. Tuyển sinh**

Trường Đại học XYZ tuyển sinh theo hình thức xét điểm thi ba môn Toán – Lý – Hóa. Để đơn giản, khu vực tuyển sinh được quy định luôn bởi ba chữ cái đầu tiên trong mã thí sinh. Do rất thích các thí sinh đến từ Khu vực 3 nên trường XYZ tự quy định giá trị điểm ưu tiên Khu vực như trong bảng sau:

KV1-0.5

KV2-1.0

KV3-2.5

Giả sử biết trước điểm chuẩn là 24. Hãy xác định tình trạng tuyển sinh của thí sinh.

**Input Format**

Chỉ bao gồm thông tin của một thí sinh trên 5 dòng lần lượt là: - Mã thí sinh - Họ tên - Điểm toán - Điểm lý - Điểm hóa

**Constraints**

Các giá trị điểm đều đảm bảo trong phạm vi [0,10] và có thể có 1 chữ số phần thập phân.

**Output Format**

Ghi ra các thông tin: - Mã thí sinh - Họ tên - Khu vực - Tổng điểm – có tính ưu tiên (có thể có 1 số phần thập phân), trong trường hợp điểm là số thực với phần thập phân bằng 0 thì không in ra phần thập phân. - Trạng thái: TRUNG TUYEN hoac TRUOT (sau khi đã tính cả điểm ưu tiên)

**Sample Input 0**

KV2B

Hoang Ngoc Long

3.4

9

0

**Sample Output 0**

KV2B Hoang Ngoc Long 2 13.4 TRUOT

**[OOP]. Bài 7. Lớp phân số**

Phân số là sự biểu diễn số hữu tỷ dưới dạng tỷ lệ của hai số nguyên, trong đó số ở trên được gọi là tử số, còn số ở dưới được gọi là mẫu số. Cho hai phân số A và B có tử số và mẫu số được nhập từ bàn phím. Viết chương trình thực hiện hai nhiệm vụ sau: 1. Thực hiện phép tính C = (A + B)^2 và rút gọn kết quả. 2. Thực hiện phép tính D = A x B x C và rút gọn kết quả.

**Input Format**

Dòng đầu tiên là số bộ test T; T dòng tiếp theo, mỗi dòng gồm 4 số lần lượt là tử và mẫu số của phân số A và phân số B

**Constraints**

(T <= 100); Các phân số có tử số và mẫu số là số nguyên trị tuyệt đối <= 100; Dữ liệu đảm bảo mẫu số khác 0.

**Output Format**

Ghi ra kết quả của hai phép tính theo định dạng phân số, cách nhau một khoảng trống.

**Sample Input 0**

6

1 2 3 4

2 3 4 5

-23 -36 99 90

24 -1 99 60

-40 -9 98 91

37 -31 61 92

**Sample Output 0**

25/16 75/128

484/225 3872/3375

97969/32400 24786157/11664000

199809/400 19781091/-1000

417316/13689 233696960/1601613

2289169/8133904 5166654433/-23197894208

**[OOP]. Bài 8. Lớp Set 1**

Trong lý thuyết tập hợp, một tập hợp chỉ được phép chứa các giá trị phân biệt và luôn luôn lưu các giá trị theo thứ tự tăng dần. Khai báo lớp IntSet và viết các phương thức để thực hiện các thao tác trên tập hợp số nguyên. Sử dụng lớp IntSet để in ra tập hợp các số nguyên là giao của hai tập số trong 2 dãy ban đầu. Sử dụng hàm main có nội dung như sau :

public static void main(String[] args){

    Scanner sc = new Scanner(System.in);

    int n = sc.nextInt(), m = sc.nextInt(), a[] = new int[n], b[] = new int[m];

    for(int i = 0; i<n; i++) a[i] = sc.nextInt();

    for(int i = 0; i<m; i++) b[i] = sc.nextInt();

    IntSet s1 = new IntSet(a);

    IntSet s2 = new IntSet(b);

    IntSet s3 = s1.intersection(s2);

    System.out.println(s3);

}

**Input Format**

Dòng đầu ghi 2 số n và m (1 < n,m <100). Dòng thứ 2 ghi n số của a[]. Dòng thứ 3 ghi m số của b[]. Các số đều dương và nhỏ hơn 1000.

**Constraints**

N/A

**Output Format**

Ghi ra hợp của hai tập theo thứ tự tăng dần.

**Sample Input 0**

14 5

92 36 73 11 78 19 57 76 96 58 36 38 29 42

19 21 15 4 78

**Sample Output 0**

19 78

**[OOP]. Bài 9. Lớp Set 2**

Trong lý thuyết tập hợp, một tập hợp chỉ được phép chứa các giá trị phân biệt và luôn luôn lưu các giá trị theo thứ tự tăng dần. Khai báo lớp IntSet và viết các phương thức để thực hiện các thao tác trên tập hợp số nguyên. Sử dụng lớp IntSet để in ra tập hợp các số nguyên là hợp của hai tập số trong 2 dãy ban đầu. Yêu cầu sử dụng hàm main có nội dung như sau để chạy chương trình.

public static void main(String[] args){

    Scanner sc = new Scanner(System.in);

    int n = sc.nextInt(), m = sc.nextInt(), a[] = new int[n], b[] = new int[m];

    for(int i = 0; i<n; i++) a[i] = sc.nextInt();

    for(int i = 0; i<m; i++) b[i] = sc.nextInt();

    IntSet s1 = new IntSet(a);

    IntSet s2 = new IntSet(b);

    IntSet s3 = s1.union(s2);

    System.out.println(s3);

}

**Input Format**

Dòng đầu ghi 2 số n và m (1 < n,m <100). Dòng thứ 2 ghi n số của a[]. Dòng thứ 3 ghi m số của b[]. Các số đều dương và nhỏ hơn 1000;

**Constraints**

N/A

**Output Format**

Ghi ra hợp của hai tập theo thứ tự tăng dần.

**Sample Input 0**

12 5

89 16 18 57 93 25 40 25 75 71 30 18

23 93 8 16 80

**Sample Output 0**

8 16 18 23 25 30 40 57 71 75 80 89 93

**[OOP]. Bài 10. Lớp set 3**

Trong lập trình cơ bản, một từ được hiểu là một dãy ký tự liên tiếp không chứa khoảng trống, dấu tab hoặc dấu xuống dòng. Xây dựng lớp WordSet để quản lý tập hợp các từ khác nhau trong một xâu ký tự, sau khi đã chuyển hết về dạng chữ thường. Khi liệt kê thì tập từ thì sẽ luôn theo thứ tự từ điển tăng dần Viết chương trình nhập hai dòng ký tự và hiển thị hợp và giao của hai tập từ tương ứng. Sử dụng hàm main có nội dung sẵn như sau :

public static void main(String[] args) {

    Scanner sc = new Scanner(System.in);

    WordSet s1 = new WordSet(sc.nextLine());

    WordSet s2 = new WordSet(sc.nextLine());

    System.out.println(s1.union(s2));

    System.out.println(s1.intersection(s2));

}

**Input Format**

Chỉ có 2 dòng, mỗi dòng có độ dài không quá 1000 ký tự.

**Constraints**

N/A

**Output Format**

Dòng 1 ghi hợp của 2 tập từ theo thứ tự từ điển Dòng 2 ghi giao của 2 tập từ theo thứ tự từ điển.

**Sample Input 0**

aabc cbcac bbaa aaa acaba ccc abaaa

caa aaa abaa aba abc acb aacb aabaa

**Sample Output 0**

aaa aabaa aabc aacb aba abaa abaaa abc acaba acb bbaa caa cbcac ccc

aaa

**[OOP]. Bài 11. Lớp ma trận**

Viết chương trình khai báo lớp Matrix mô tả ma trận các số nguyên. Sau đó nhập và tính tích hai ma trận A cỡ n\*m và ma trận B cỡ m\*p. Với 1 < n,m,p < 50. Các giá trị trong ma trận đều nguyên dương và không vượt quá 1000. Sử dụng hàm main có sẵn :

public static void main(String[] args) {

    Scanner sc = new Scanner(System.in);

    int n = sc.nextInt(), m=sc.nextInt(), p=sc.nextInt();

    Matrix a = new Matrix(n,m);

    a.nextMatrix(sc);

    Matrix b = new Matrix(m,p);

    b.nextMatrix(sc);

    System.out.println(a.mul(b));

}

**Input Format**

Dòng đầu ghi 3 số n,m,p; n dòng tiếp theo ghi ma trận A; m dòng tiếp theo ghi ma trận B;

**Constraints**

N/A

**Output Format**

In ra tích của ma trận

**Sample Input 0**

3 2 4

147 790

393 462

675 445

690 172 776 550

344 761 900 713

**Sample Output 0**

373190 626474 825072 644120

430098 419178 720768 545556

618830 454745 924300 688535

**[OOP]. Bài 12. Lớp ma trận 2**

Khai báo lớp Matrix mô tả ma trận các số nguyên. Nhập ma trận A cấp N \* M. Hãy viết chương trình tính tích của A với ma trận chuyển vị của A.  
Sử dụng hàm main có nội dũng sẵn như sau :

public static void main(String[] args) {

    Scanner sc = new Scanner(System.in);

    int n = sc.nextInt(), m = sc.nextInt();

    Matrix a = new Matrix(n,m);

    a.nextMatrix(sc);

    Matrix b = a.trans();

    System.out.println(a.mul(b));

}

**Input Format**

Dòng đầu tiên ghi hai số n và m là bậc của ma trân a; n dòng tiếp theo, mỗi dòng ghi m số của một dòng trong ma trận A.

**Constraints**

n và m đều nguyên dương và nhỏ hơn 50. Các giá trị trong ma trận không vượt quá 100.

**Output Format**

Với mỗi bộ test ghi ra ma trận tích tương ứng, mỗi số cách nhau đúng một khoảng trống.

**Sample Input 0**

5 2

198 725

686 514

183 392

392 717

134 62

**Sample Output 0**

564829 508478 320434 597441 71482

508478 734792 327026 637450 123792

320434 327026 187153 352800 48826

597441 637450 352800 667753 96982

71482 123792 48826 96982 21800

**[OOP]. Bài 13. Lớp Time**

Giá trị độ đo thời gian được biểu diễn bởi ba thành phần: giờ, phút, giây. Cho N giá trị thời gian, hãy sắp xếp danh sách theo thứ tự tăng dần.

**Input Format**

Dòng đầu ghi số nguyên dương N (không quá 5000) là số lượng giá trị thời gian cần sắp xếp. Mỗi giá trị thời gian biểu diễn trên một dòng bằng ba số nguyên dương, lần lượt là số giờ, số phút, số giây. Trong đó số giờ đảm bảo nhỏ hơn 100, số phút và số giây đảm bảo đúng quy tắc (tức là không quá 59).

**Constraints**

N/A

**Output Format**

In ra danh sách đã sắp xếp theo thứ tự tăng dần.

**Sample Input 0**

5

25 2 33

18 3 17

12 6 19

1 6 12

98 51 26

**Sample Output 0**

1 6 12

12 6 19

18 3 17

25 2 33

98 51 26

**[OOP]. Bài 14. Danh sách mặt hàng**

Bài toán quản lý danh sách mặt hàng trong đó mỗi mặt hàng sẽ có các thông tin: - Mã mặt hàng: tự động tăng, tính từ MH0001.

* Tên mặt hàng: xâu ký tự độ dài không quá 100.
* Đơn vị tính: xâu ký tự độ dài không quá 10.
* Giá mua: số nguyên dương không quá 7 chữ số.
* Giá bán: số nguyên dương không quá 7 chữ số.

Viết chương trình nhập danh sách mặt hàng, sắp xếp theo lợi nhuận (giá bán trừ đi giá mua) giảm dần. Nếu lợi nhuận bằng nhau thì in ra theo thứ tự mã tăng dần.

**Input Format**

Dòng đầu ghi số M là số mặt hàng (không quá 5000). Tiếp theo là thông tin của M mặt hàng, mỗi mặt hàng ghi trên 4 dòng theo đúng thứ tự đã mô tả (không có mã)

**Constraints**

N/A

**Output Format**

Ghi ra danh sách mặt hàng có đầy đủ thông tin ở trên và lợi nhuận tính được (mỗi thông tin cách nhau một khoảng trống)

**Sample Input 0**

6

Ca phe

Thung

388000

492000

Dep

Thung

376000

495000

Giay Nguoi Lon

Cai

389000

580000

Giay Nguoi Lon

Thung

383000

503000

Dep

Thung

484000

647000

Vay Nu

Cai

366000

516000

**Sample Output 0**

MH0003 Giay Nguoi Lon Cai 389000 580000 191000

MH0005 Dep Thung 484000 647000 163000

MH0006 Vay Nu Cai 366000 516000 150000

MH0004 Giay Nguoi Lon Thung 383000 503000 120000

MH0002 Dep Thung 376000 495000 119000

MH0001 Ca phe Thung 388000 492000 104000

**[OOP]. Bài 15. Sinh viên 1**

Viết chương trình khai báo lớp Sinh Viên gồm các thông tin: Mã SV, Họ tên, Ngày sinh, Lớp và Điểm GPA (dạng số thực). Hàm khởi tạo không có tham số, gán các giá trị thuộc tính ở trạng thái mặc định (xâu ký tự rỗng, giá trị số bằng 0). Đọc thông tin N sinh viên từ bàn phím (không có mã sinh viên) và in ra lần lượt màn hình mỗi dòng 1 sinh viên theo đúng thứ tự ban đầu. Trong đó Mã SV được tự tạo ra theo quy tắc thêm mã SV sau đó là giá trị nguyên tự động tăng tính từ 001 (tối đa là 099). Ngày sinh được chuẩn hóa về dạng dd/mm/yyyy

**Input Format**

Dòng đầu tiên ghi số sinh viên N (0 < N < 50). Mỗi sinh viên ghi trên 4 dòng lần lượt là Họ tên, Lớp, Ngày sinh và Điểm GPA. Trong đó: • Họ tên không quá 30 chữ cái. • Lớp là một chuỗi không có dấu cách. • Ngày sinh có đủ 3 phần ngày tháng năm nhưng có thể chưa đúng chuẩn dd/mm/yyyy. • Điểm GPA đảm bảo trong thang điểm 4 với 2 nhiều nhất 2 số sau dấu phẩy.

**Constraints**

N/A

**Output Format**

Ghi ra danh sách lần lượt các sinh viên có đầy đủ Mã sinh viên, Họ tên, Lớp, Ngày sinh (đã chuẩn hóa về dạng dd/mm/yyyy), Điểm GPA (với đúng 2 số sau dấu phẩy). Mỗi sinh viên ghi trên 1 dòng, mỗi thông tin cách nhau 1 khoảng trống.

**Sample Input 0**

5

Nguyen Van A

CNTT2

19/11/2000

3.60

Nguyen Van B

CNTT1

19/3/1999

0.10

Nguyen Van C

DTVT2

21/11/2001

1.30

Nguyen Van D

DTVT2

20/1/2000

2.70

Nguyen Van E

DTVT2

17/5/2003

0.30

**Sample Output 0**

SV001 Nguyen Van A CNTT2 19/11/2000 3.60

SV002 Nguyen Van B CNTT1 19/03/1999 0.10

SV003 Nguyen Van C DTVT2 21/11/2001 1.30

SV004 Nguyen Van D DTVT2 20/01/2000 2.70

SV005 Nguyen Van E DTVT2 17/05/2003 0.30

**[OOP]. Bài 16. Sinh viên 2**

Viết chương trình khai báo lớp Sinh Viên gồm các thông tin: Mã SV, Họ tên, Ngày sinh, Lớp và Điểm GPA (dạng số thực). Hàm khởi tạo không có tham số, gán các giá trị thuộc tính ở trạng thái mặc định (xâu ký tự rỗng, giá trị số bằng 0). Đọc thông tin N thí sinh từ bàn phím (không có mã sinh viên) và in ra lần lượt màn hình mỗi dòng 1 sinh viên theo đúng thứ tự ban đầu. Trong đó Mã SV được tự tạo ra theo quy tắc thêm mã SV sau đó là giá trị nguyên tự động tăng tính từ 001 (tối đa là 099). Họ tên được xử lý đưa về dạng chuẩn(viết hoa chữ cái đầu của từng từ trong tên). Ngày sinh được chuẩn hóa về dạng dd/mm/yyyy

**Input Format**

Dòng đầu tiên ghi số sinh viên N (0 < N < 50). Mỗi sinh viên ghi trên 4 dòng lần lượt là Họ tên, Lớp, Ngày sinh và Điểm GPA. Trong đó: • Họ tên không quá 30 chữ cái. • Lớp là một chuỗi không có dấu cách. • Ngày sinh có đủ 3 phần ngày tháng năm nhưng có thể chưa đúng chuẩn dd/mm/yyyy. • Điểm GPA đảm bảo trong thang điểm 4 với 2 nhiều nhất 2 số sau dấu phẩy.

**Constraints**

N/A

**Output Format**

Ghi ra danh sách lần lượt các sinh viên có đầy đủ Mã sinh viên, Họ tên, Lớp, Ngày sinh (đã chuẩn hóa), điểm GPA (với đúng 2 số sau dấu phẩy). Mỗi sinh viên ghi trên 1 dòng, mỗi thông tin cách nhau 1 khoảng trống.

**Sample Input 0**

5

Hoang Van Long

CNTT2

26/7/2000

0.20

Tran Xuan Loc

CNTT1

3/1/1999

2.40

Nguyen Van Manh

CNTT3

18/11/2000

3.30

Ho Sy Kien

CNTT3

4/6/2000

1.50

Le Van Manh

CNTT1

13/3/2002

3.10

**Sample Output 0**

SV001 Hoang Van Long CNTT2 26/07/2000 0.20

SV002 Tran Xuan Loc CNTT1 03/01/1999 2.40

SV003 Nguyen Van Manh CNTT3 18/11/2000 3.30

SV004 Ho Sy Kien CNTT3 04/06/2000 1.50

SV005 Le Van Manh CNTT1 13/03/2002 3.10

**[OOP]. Bài 17. Sắp xếp sinh viên theo GPA và Mã sinh viên**

Viết chương trình khai báo lớp Sinh Viên gồm các thông tin:

* Mã SV
* Họ tên
* Ngày sinh
* Lớp
* Điểm GPA (dạng số thực).

Đọc thông tin N thí sinh từ bàn phím (không nhập mã sinh viên) sau đó sắp xếp theo điểm GPA giảm dần và in ra lần lượt màn hình mỗi dòng 1 sinh viên. Nếu 2 bạn có cùng GPA thì bạn nào có mã sinh viên nhỏ hơn sẽ xếp trước.

Trong đó Mã SV được tự tạo ra theo quy tắc thêm mã SV sau đó là giá trị nguyên tự động tăng tính từ 001 (tối đa là 050). Họ tên được xử lý đưa về dạng chuẩn. Ngày sinh được chuẩn hóa về dạng dd/mm/yyyy

**Input Format**

Dòng đầu tiên ghi số sinh viên N (0 < N < 50). Mỗi sinh viên ghi trên 4 dòng lần lượt là Họ tên, Lớp, Ngày sinh và Điểm GPA. Trong đó: - Họ tên không quá 30 chữ cái.

* Lớp là một chuỗi kí tự không có dấu cách.
* Ngày sinh có đủ 3 phần ngày tháng năm nhưng có thể chưa đúng chuẩn dd/mm/yyyy.
* Điểm GPA đảm bảo trong thang điểm 4 với 2 nhiều nhất 2 số sau dấu phẩy.

**Constraints**

N/A

**Output Format**

Ghi ra danh sách lần lượt các sinh viên có đầy đủ Mã sinh viên, Họ tên, Lớp, Ngày sinh (đã chuẩn hóa), điểm GPA (với đúng 2 số sau dấu phẩy) đã được sắp xếp theo điểm GPA giảm dần. Mỗi sinh viên ghi trên 1 dòng, mỗi thông tin cách nhau 1 khoảng trống.

**Sample Input 0**

2

ngUYen Van NaM

CNTT1

2/12/1994

2.17

Nguyen QuanG hAi

CNTT2

1/9/1994

3.0

**Sample Output 0**

SV002 Nguyen Quang Hai CNTT2 01/09/1994 3.00

SV001 Nguyen Van Nam CNTT1 02/12/1994 2.17

**[OOP]. Bài 18. Nhân viên 1**

Một nhân viên làm việc trong công ty được lưu lại các thông tin sau: • Mã nhân viên: được gán tự động tăng, bắt đầu từ 00001 • Họ tên: Xâu ký tự không quá 40 chữ cái. • Giới tính: Nam hoặc Nu • Ngày sinh: đúng theo chuẩn dd/mm/yyyy • Địa chỉ: Xâu ký tự không quá 100 chữ cái • Mã số thuế: Dãy số có đúng 10 chữ số • Ngày ký hợp đồng: đúng theo chuẩn dd/mm/yyyy Viết chương trình nhập danh sách nhân viên (không nhập mã) trong đó có sử dụng chồng toán tử nhập/xuất và in ra màn hình danh sách vừa nhập.

**Input Format**

Dòng đầu ghi số N là số nhân viên (không quá 40). Mối nhân viên ghi trên 6 dòng lần lượt ghi các thông tin theo thứ tự đã ghi trong đề bài. Không có mã nhân viên.

**Constraints**

N/A

**Output Format**

Ghi ra danh sách đầy đủ nhân viên, mỗi nhân viên trên một dòng, các thông tin cách nhau đúng một khoảng trống. Thông tin nhân viên được chuẩn hóa ngày sinh và ngày kí hợp đồng.

**Sample Input 0**

3

Nguyen Van A

Nam

22/10/1982

Mo Lao-Ha Dong-Ha Noi

8333012345

31/12/2013

Ly Thi B

Nu

15/10/1988

Mo Lao-Ha Dong-Ha Noi

8333012346

22/08/2011

Hoang Thi C

Nu

04/02/1981

Mo Lao-Ha Dong-Ha Noi

8333012347

22/08/2011

**Sample Output 0**

00001 Nguyen Van A Nam 22/10/1982 Mo Lao-Ha Dong-Ha Noi 8333012345 31/12/2013

00002 Ly Thi B Nu 15/10/1988 Mo Lao-Ha Dong-Ha Noi 8333012346 22/08/2011

00003 Hoang Thi C Nu 04/02/1981 Mo Lao-Ha Dong-Ha Noi 8333012347 22/08/2011

**[OOP]. Bài 19. Nhân viên 2**

Một nhân viên làm việc trong công ty được lưu lại các thông tin sau: • Mã nhân viên: được gán tự động tăng, bắt đầu từ 00001. • Họ tên: Xâu ký tự không quá 40 chữ cái. • Giới tính: Nam hoặc Nu. • Ngày sinh: đúng theo chuẩn dd/mm/yyyy. • Địa chỉ: Xâu ký tự không quá 100 chữ cái. • Mã số thuế: Dãy số có đúng 10 chữ số. • Ngày ký hợp đồng: đúng theo chuẩn dd/mm/yyyy. Viết chương trình nhập danh sách nhân viên (không nhập mã) sau đó sắp xếp theo thứ tự ngày sinh từ già nhất đến trẻ nhất và in ra màn hình danh sách đối tượng nhân viên đã sắp xếp. Nếu có 2 nhân viên có cùng ngày sinh thì in ra nhân viên có mã nhân viên nhỏ hơn.

**Input Format**

Dòng đầu ghi số N là số nhân viên (không quá 40). Mỗi nhân viên ghi trên 6 dòng lần lượt ghi các thông tin theo thứ tự đã ghi trong đề bài. Không có mã nhân viên. Thông tin về ngày sinh và ngày kí hợp đồng được chuẩn hóa.

**Constraints**

N/A

**Output Format**

Ghi ra danh sách đầy đủ nhân viên đã sắp xếp, mỗi nhân viên trên một dòng, các thông tin cách nhau đúng một khoảng trống.

**Sample Input 0**

3

Nguyen Van A

Nam

17/10/1995

Bac Ninh

9778128747

6/11/2022

Nguyen Thi B

Nu

18/09/1997

Hai Duong

7823490168

5/4/2022

Nguyen Thi C

Nu

02/04/1995

Ninh Binh

7400512775

27/1/2020

**Sample Output 0**

00003 Nguyen Thi C Nu 02/04/1995 Ninh Binh 7400512775 27/01/2020

00001 Nguyen Van A Nam 17/10/1995 Bac Ninh 9778128747 06/11/2022

00002 Nguyen Thi B Nu 18/09/1997 Hai Duong 7823490168 05/04/2022

**[OOP]. Bài 20. Tính giờ quán net**

Quán Game mùa này vắng khách nên chủ quán quyết định tính tiền chi tiết đến từng phút. Dựa trên dữ liệu giờ vào và giờ ra, hãy tính thời gian chơi game của các Game thủ nhé.

**Input Format**

Dòng đầu của dữ liệu vào ghi số lượng game thủ trong ngày (không quá 20). Thông tin về một game thủ đến chơi game được ghi lại trên 4 dòng lần lượt là:

* username (xâu ký tự độ dài không quá 20, không có khoảng trống).
* password (xâu ký tự độ dài không quá 30, không có khoảng tróng).
* Tên người chơi (xâu ký tự độ dài không quá 100, có thể có khoảng trống).
* Giờ vào (định dạng hh:mm)
* Giờ ra (định dạng hh:mm).

**Constraints**

* 0<=hh<=23
* 0<=mm<=59
* Giờ vào nhỏ hơn giờ ra

**Output Format**

Ghi ra danh sách game thủ đã được sắp xếp theo thời gian chơi game giảm dần, nếu có 2 game thủ có cùng giờ chơi game thì bạn nào có username có thứ tự từ điển nhỏ hơn được in ra trước.

**Sample Input 0**

5

anhtuanvip

123

Nguyen Van Tuan

05:18

07:06

chickenzero

124

Nguyen Van Phuc

05:38

14:19

anhhung123

matkhau

Nguyen Manh Hung

06:58

14:18

loveyou

acb

Luong Van Manh

02:01

06:47

taikhoan123

matkhaumanh

Nguyen Thi Uyen

00:09

06:19

**Sample Output 0**

chickenzero 124 Nguyen Van Phuc 8 gio 41 phut

anhhung123 matkhau Nguyen Manh Hung 7 gio 20 phut

taikhoan123 matkhaumanh Nguyen Thi Uyen 6 gio 10 phut

loveyou acb Luong Van Manh 4 gio 46 phut

anhtuanvip 123 Nguyen Van Tuan 1 gio 48 phut

**Sample Input 1**

5

bHaqxvui

iGYZMmhekYzBq

PaGc VdA UaVWtt IsPt DhgKH

09:50

18:30

pGZKupUVGMeI

aADAgxFMmMvUn

kPmi OQt hnMIp NzYHS

02:24

12:18

LnIHfIzsUEz

FLlpZRmPWpROGF

tpsFu gbBtEt wlvk GGF INg

04:31

07:10

FfqqlBTOB

vKFBIHmBhR

VgFVPZ sju YqOZeV qnjxcO

00:31

03:05

aUuZXfMAx

cWcgqrKmq

muTd HZDtO EOXaNE LtuTy SGNA

11:52

13:42

**Sample Output 1**

pGZKupUVGMeI aADAgxFMmMvUn kPmi OQt hnMIp NzYHS 9 gio 54 phut

bHaqxvui iGYZMmhekYzBq PaGc VdA UaVWtt IsPt DhgKH 8 gio 40 phut

LnIHfIzsUEz FLlpZRmPWpROGF tpsFu gbBtEt wlvk GGF INg 2 gio 39 phut

FfqqlBTOB vKFBIHmBhR VgFVPZ sju YqOZeV qnjxcO 2 gio 34 phut

aUuZXfMAx cWcgqrKmq muTd HZDtO EOXaNE LtuTy SGNA 1 gio 50 phut

**[OOP]. Bài 21. Bảng điểm**

Trường THCS XYZ lập bảng điểm tổng kết cho học sinh. Có 10 môn học lần lượt gồm: Toán, Tiếng Việt, Ngoại ngữ, Vật lý, Hóa học, Sinh học, Lịch Sử, Địa, Giáo dục công dân và môn Công nghệ. Trong đó các môn đều có hệ số là 1. Học sinh được xếp hạng theo điểm trung bình: • Từ 9 trở lên: loại XUAT SAC. • Từ 8 đến 8.9: loại GIOI. • Từ 7 đến 7.9: loại KHA. • Từ 5 đến 6.9: loại TB. • Dưới 5: loai YEU Hãy lập bảng điểm tổng kết và sắp xếp theo điểm trung bình giảm dần.

**Input Format**

Dòng đầu ghi số học sinh (không quá 50). Thông tin về mỗi học sinh có hai dòng: dòng đầu là họ tên (độ dài không quá 50), dòng thứ 2 gồm 10 số thực trong đoạn [0..10] lần lượt là điểm 10 môn theo đúng thứ tự đã mô tả.

**Constraints**

Chú ý : Sau điểm cuối cùng có dấu cách :)

**Output Format**

Danh sách đã sắp xếp được ghi ra bao gồm các thông tin: • Mã học sinh (tự động gán tăng dần theo thứ tự nhập, bắt đầu là HS01, chú ý bài này học sinh có mã 10 sẽ có mã học sinh là SV010) • Họ và tên • Điểm trung bình (với 1 chữ số phần thập phân) • Xếp loại Trong trường hợp điểm trung bình bằng nhau thì học sinh nào có mã học sinh nhỏ hơn sẽ xếp trên.

**Sample Input 0**

5

Tran Duc Luong

7.8 5.9 9.0 7.2 6.5 4.6 8.3 9.6 4.0 6.0

Do Xuan Loc

8.4 0.9 4.4 4.6 2.4 9.5 2.1 3.6 9.5 6.3

Vu Duc Hieu

6.5 8.8 7.7 9.4 2.9 0.5 1.5 3.7 4.8 6.9

Phan Van Nam

8.2 6.4 8.3 2.3 8.3 2.9 8.7 7.1 9.4 3.4

Vu Duc Minh

7.8 5.3 8.3 1.6 2.6 9.8 0.6 0.1 5.2 2.6

**Sample Output 0**

HS01 Tran Duc Luong 6.9 TB

HS04 Phan Van Nam 6.5 TB

HS03 Vu Duc Hieu 5.3 TB

HS02 Do Xuan Loc 5.2 TB

HS05 Vu Duc Minh 4.4 YEU

**[OOP]. Bài 22. Sắp xếp sinh viên theo lớp**

Thông tin về mỗi sinh viên gồm: • Mã sinh viên: dãy ký tự không có khoảng trống (không quá 15). Đảm bảo không trùng nhau. • Họ và tên: độ dài không quá 100. • Lớp: dãy ký tự không có khoảng trống (không quá 15). • Email: dãy ký tự không có khoảng trống (không quá 15). Hãy nhập danh sách sinh viên và sắp xếp theo lớp tăng dần (thứ tự từ điển).

**Input Format**

Dòng đầu ghi số sinh viên. Mỗi sinh viên ghi trên 4 dòng lần lượt là: mã, họ tên, lớp, email. Có không quá 5000 sinh viên trong danh sách.

**Constraints**

N/A

**Output Format**

Ghi ra danh sách sinh viên đã sắp xếp theo lớp. Mỗi sinh viên trên một dòng, các thông tin cách nhau một khoảng trống. Nếu 2 sinh viên có cùng lớp thì sắp xếp theo mã tăng dần (thứ tự từ điển)

**Sample Input 0**

4

SV001

Nguyen Trong Duc Anh

CNTT1

sv1@gmail.com

SV002

To Ngoc Hieu

CNTT2

sv2@gmail.com

SV003

Nguyen Ngoc Son

HTTT2

sv3@gmail.com

SV004

Nguyen Trong Tung

CNTT1

sv4@gmail.com

**Sample Output 0**

SV001 Nguyen Trong Duc Anh CNTT1 sv1@gmail.com

SV004 Nguyen Trong Tung CNTT1 sv4@gmail.com

SV002 To Ngoc Hieu CNTT2 sv2@gmail.com

SV003 Nguyen Ngoc Son HTTT2 sv3@gmail.com

**[OOP]. Bài 33. Sắp xếp sinh viên theo mã**

Thông tin về mỗi sinh viên gồm: • Mã sinh viên: dãy ký tự không có khoảng trống (không quá 15). Đảm bảo không trùng nhau. • Họ và tên: độ dài không quá 100. • Lớp: dãy ký tự không có khoảng trống (không quá 15). • Email: dãy ký tự không có khoảng trống (không quá 15). Hãy nhập danh sách sinh viên và sắp xếp theo mã sinh viên tăng dần (thứ tự từ điển).

**Input Format**

Mỗi sinh viên ghi trên 4 dòng lần lượt là: mã, họ tên, lớp, email. Không cho biết số sinh viên nhưng dữ liệu đảm bảo là chẵn lần 4 dòng. Có không quá 5000 sinh viên trong danh sách.

**Constraints**

N/A

**Output Format**

Ghi ra danh sách sinh viên đã sắp xếp theo mã. Mỗi sinh viên trên một dòng, các thông tin cách nhau một khoảng trống.

**Sample Input 0**

SV004

Nguyen Trong Tung

CNTT1

sv4@gmail.com

SV001

Nguyen Trong Duc Anh

CNTT1

sv1@gmail.com

SV002

To Ngoc Hieu

CNTT2

sv2@gmail.com

SV003

Nguyen Ngoc Son

HTTT2

sv3@gmail.com

**Sample Output 0**

SV001 Nguyen Trong Duc Anh CNTT1 sv1@gmail.com

SV002 To Ngoc Hieu CNTT2 sv2@gmail.com

SV003 Nguyen Ngoc Son HTTT2 sv3@gmail.com

SV004 Nguyen Trong Tung CNTT1 sv4@gmail.com

**[OOP]. Bài 24. Liệt kê sinh viên theo lớp**

Thông tin về mỗi sinh viên gồm: • Mã sinh viên: dãy ký tự không có khoảng trống (không quá 15). Đảm bảo không trùng nhau. • Họ và tên: độ dài không quá 100, chưa được chuẩn hóa. • Lớp: dãy ký tự không có khoảng trống (không quá 15). • Email: dãy ký tự không có khoảng trống (không quá 15). Hãy nhập danh sách sinh viên và liệt kê sinh viên theo lớp.

**Input Format**

Dòng đầu ghi số sinh viên (không quá 1000). Mỗi sinh viên ghi trên 4 dòng lần lượt là: mã, họ tên, lớp, email. Sau đó sẽ có giá trị số Q là số truy vấn. Tiếp theo là Q dòng, mỗi dòng ghi một lớp.

**Constraints**

1<=Q<=20;

**Output Format**

Với mỗi truy vấn, liệt kê danh sách sinh viên của lớp đó theo mẫu như trong ví dụ. Mỗi sinh viên ghi trên một dòng, các thông tin cách nhau một khoảng trống. Thứ tự sinh viên vẫn giữ nguyên như thứ tự ban đầu.

**Sample Input 0**

4

20132238

hoang dinh NAm

CNTT1

namhd@gmail.com

20142231

PhAM nGoc tHuan

HTTT2

thuanpn@gmail.com

20148971

Nguyen duc XUAN

CNTT1

xuannd@gmail.com

20132038

PhAM vAn Hung

hungpv@gmail.com

CNTT2

1

CNTT1

**Sample Output 0**

DANH SACH SINH VIEN LOP CNTT1 :

20132238 Hoang Dinh Nam CNTT1 namhd@gmail.com

20148971 Nguyen Duc Xuan CNTT1 xuannd@gmail.com

**[OOP]. Bài 25. Liệt kê sinh viên theo khóa**

Thông tin về mỗi sinh viên gồm: - Mã sinh viên: dãy ký tự không có khoảng trống (không quá 15). Đảm bảo không trùng nhau.

* Họ và tên: độ dài không quá 100, chưa được chuẩn hóa và cần được chuẩn hóa trước khi in.
* Lớp: dãy ký tự không có khoảng trống (không quá 15).
* Email: dãy ký tự không có khoảng trống (không quá 15).

Hãy nhập danh sách sinh viên và liệt kê sinh viên theo khóa học. Chú ý: dữ liệu khóa học thể hiện qua 4 chữ số đầu tiên trong mã sinh viên

**Input Format**

* Dòng đầu ghi số sinh viên (không quá 1000).
* Mỗi sinh viên ghi trên 4 dòng lần lượt là: mã, họ tên, lớp, email.
* Sau đó sẽ có giá trị số Q là số truy vấn.
* Tiếp theo là Q dòng, mỗi dòng ghi năm bắt đầu khóa học theo định dạng yyyy.

**Constraints**

1<=Q<=100

**Output Format**

Với mỗi truy vấn, liệt kê danh sách sinh viên của khóa đó theo mẫu như trong ví dụ. Mỗi sinh viên ghi trên một dòng, các thông tin cách nhau một khoảng trống. Thứ tự sinh viên vẫn giữ nguyên như thứ tự ban đầu.

**Sample Input 0**

4

20132238

hoang dinh NAm

CNTT1

namhd@gmail.com

20142231

PhAM nGoc tHuan

HTTT2

thuanpn@gmail.com

20148971

Nguyen duc XUAN

CNTT1

xuannd@gmail.com

20132038

PhAM vAn Hung

CNTT2

hungpv@gmail.com

2

2013

2014

**Sample Output 0**

DANH SACH SINH VIEN KHOA 2013 :

20132238 Hoang Dinh Nam CNTT1 namhd@gmail.com

20132038 Pham Van Hung CNTT2 hungpv@gmail.com

DANH SACH SINH VIEN KHOA 2014 :

20142231 Pham Ngoc Thuan HTTT2 thuanpn@gmail.com

20148971 Nguyen Duc Xuan CNTT1 xuannd@gmail.com

**[OOP]. Bài 26. Liệt kê sinh viên theo ngành**

Thông tin về mỗi sinh viên gồm: • Mã sinh viên: dãy ký tự không có khoảng trống (không quá 15). Đảm bảo không trùng nhau. • Họ và tên: độ dài không quá 100.  
• Lớp: dãy ký tự không có khoảng trống (không quá 15). • Email: dãy ký tự không có khoảng trống (không quá 15). Hãy nhập danh sách sinh viên và liệt kê sinh viên theo ngành. Ngành học của sinh viên sẽ dựa trên 4 chữ cái đầu tiên của lớp bao gồm các ngành : CNTT : Công nghệ thông tin. DTVT : Điện tử viễn thông. KT : Kế toán. MKT : Marketing.

**Input Format**

Dòng đầu ghi số sinh viên (không quá 1000) Mỗi sinh viên ghi trên 4 dòng lần lượt là: mã, họ tên, lớp, email. Sau đó sẽ có giá trị số Q là số truy vấn. Mỗi truy vấn sẽ là 1 trong 4 ngành nêu trên.

**Constraints**

1<=q<=100

**Output Format**

Với mỗi truy vấn, liệt kê danh sách sinh viên của ngành đó theo mẫu như trong ví dụ. Mỗi sinh viên ghi trên một dòng, các thông tin cách nhau một khoảng trống. Thứ tự sinh viên vẫn giữ nguyên như thứ tự ban đầu

**Sample Input 0**

3

20211000

Nguyen Ngoc Huong

CNTT2

DqFvSInO@gmail.com

20181001

Vu Duc Huong

CNTT2

KBvHj@gmail.com

20191002

Pham Duc Hoang

DTVT2

HEKglq@gmail.com

1

CNTT

**Sample Output 0**

DANH SACH SINH VIEN NGANH CONG NGHE THONG TIN :

20211000 Nguyen Ngoc Huong CNTT2 DqFvSInO@gmail.com

20181001 Vu Duc Huong CNTT2 KBvHj@gmail.com

**[OOP]. Bài 27. Sắp xếp giảng viên**

Danh sách giảng viên của trường đại học cần được sắp xếp lại theo tên. Thông tin về giảng viên ban đầu chỉ có họ tên và Bộ môn. Mã giảng viên tự động tăng, tính từ GV01. Cần sắp xếp lại theo tên (tức là từ cuối cùng trong xâu họ tên). Các giảng viên có cùng tên thì được sắp xếp theo mã giảng viên.

**Input Format**

Dòng đầu ghi số giảng viên (không quá 50). Mỗi giảng viên ghi trên 2 dòng gồm họ tên (không quá 50 ký tự) và Bộ môn (không quá 30 ký tự).

**Constraints**

N/A

**Output Format**

Danh sách đã sắp xếp trong đó mỗi giảng viên ghi trên một dòng. Mã được tự động điền theo thứ tự nhập, bộ môn được viết tắt theo các chữ cái đầu của từng từ và ở dạng in hoa.

**Sample Input 0**

5

Tran Duc Huong

Toan tin

Vu Ngoc Manh

Da phuong tien

Tran Van Huong

Cong nghe thong tin

Phan Xuan Minh

Ke toan

Phan Ngoc Huong

Ke toan

**Sample Output 0**

GV01 Tran Duc Huong TT

GV03 Tran Van Huong CNTT

GV05 Phan Ngoc Huong KT

GV02 Vu Ngoc Manh DPT

GV04 Phan Xuan Minh KT

**[OOP].Bài 28. Account Class**

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].Bài 29. Bank Account**

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].Bài 30. Giải Cứu**

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].Bài 31. Sunday League**

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

------------------

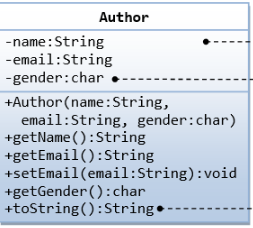
#20 SOU Southampton 30 -24 24

------------------

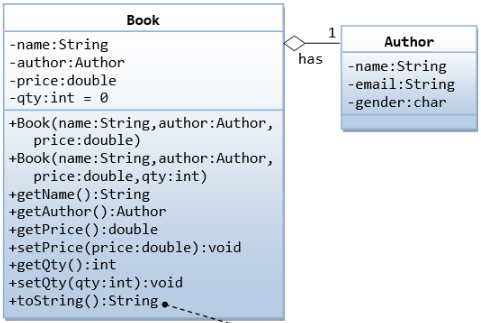
**[OOP].Bài 32. Book and Author Composition**

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

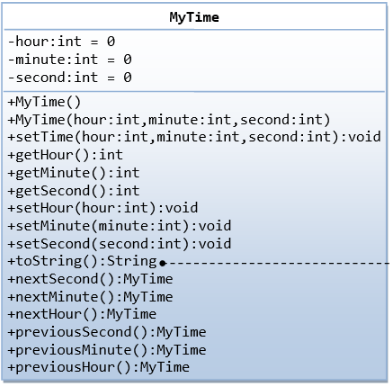
lee@gmail.com

F

--------------------

**[OOP].Bài 33. MyTime Class**

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

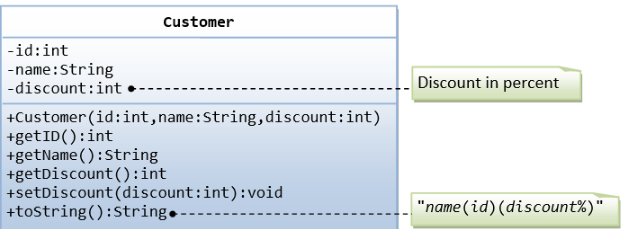
23:00:36

22:59:37

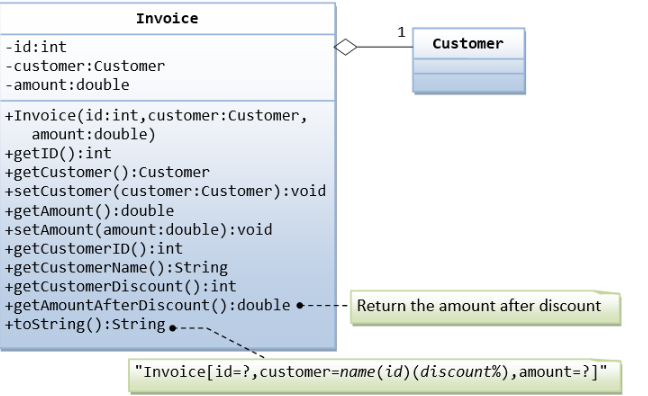
22:00:37

**[OOP].Bài 34. Customer And Invoice**

* 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

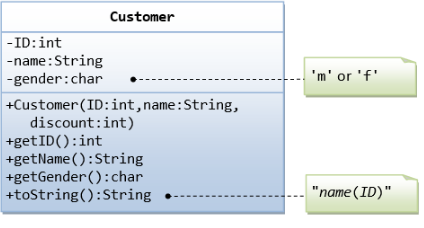
Full Name : Andrew Tate

Amount : 606.48 $

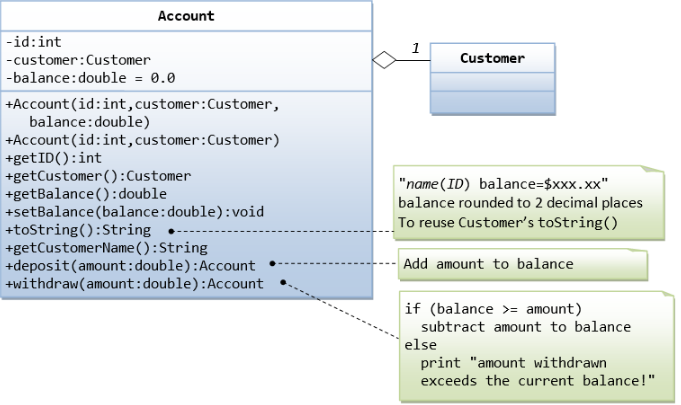
--------------------

**[OOP].Bài 35. Customer And Account**

* 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

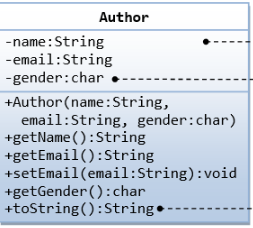
Gender : M

Account ID : 886123

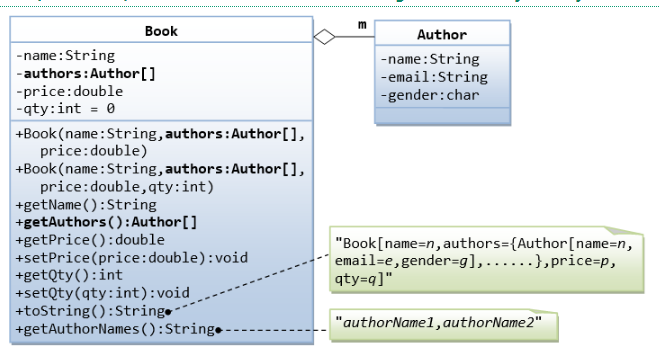
Balance : 71362.00 $

**[OOP].Bài 36. Book and Authors**

* 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

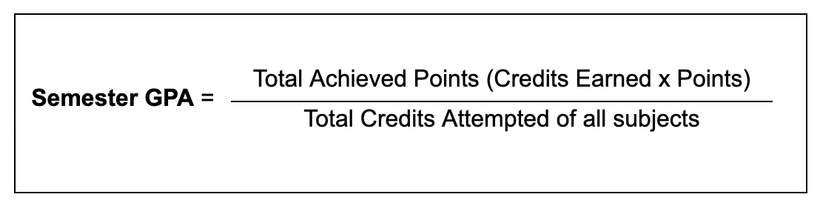
Name : Kirk Wise

Email : Kirk Wise.author@gmail.com

Gender : M

**[OOP].Bài 37. Student And Subject (One to many relationship)**

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 :

**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

------------------------

**[Kế thừa - Đa hình]. Bài 38. Kế thừa lớp Person và sắp xếp theo tên**

Xây dựng lớp Person với các thuộc tính : Tên (xâu kí tự không quá 100 kí tự), ngày sinh, địa chỉ và các hàm khởi tạo không tham số (gán các trường là xâu rỗng) và hàm khởi tạo đầy đủ tham số, phương thức toString để trả về thông tin. Lớp Student kế thừa từ lớp Person và có thêm thuộc tính là mã sinh viên, GPA và lớp, ghi đè phương thức toString. Nhập danh sách sinh viên từ bàn phím và in ra màn hình danh sách sinh viên trong đó tên được chuẩn hóa và ngày sinh đưa về đúng chuẩn dd/mm/yyyy. Tên sinh viên được sắp xếp theo thứ tự từ điển tăng dần, thứ tự từ điển của tên được xét từ tên, họ, đệm. Nếu 2 bạn cùng tên thì bạn nào xuất hiện trong danh sách trước được in ra trước.

import java.util.ArrayList;

import java.util.Arrays;

import java.util.Collections;

import java.util.Comparator;

import java.util.InputMismatchException;

import java.util.Scanner;

class Person {

private String ten, ngaySinh, diaChi;

public Person() {

ten = ngaySinh = diaChi = "";

}

public Person(String ten, String ngaySinh, String diaChi) {

this.ten = ten;

this.ngaySinh = ngaySinh;

this.diaChi = diaChi;

}

public void chuanHoa(){

String[] arr = this.ten.split("\\s+");

String res = "";

for(String x : arr){

res += Character.toUpperCase(x.charAt(0));

for(int j = 1; j < x.length(); j++){

res += Character.toLowerCase(x.charAt(j));

}

res += " ";

}

this.ten = res.substring(0, res.length() - 1);

StringBuilder sb = new StringBuilder(this.ngaySinh);

if(sb.charAt(1) == '/') sb.insert(0, "0");

if(sb.charAt(4) == '/') sb.insert(3, "0");

this.ngaySinh = sb.toString();

}

//Nguyen Van Nam => NamNguyenVan

public String getSortedName(){

String[] arr = this.ten.split("\\s+");

String res = arr[arr.length - 1];

for(int i = 0; i < arr.length - 1; i++){

res += arr[i] + " ";

}

return res;

}

@Override

public String toString(){

return this.ten + " " + this.ngaySinh + " " + this.diaChi;

}

}

class Student extends Person{

private String maSinhVien, lop;

private double gpa;

public Student(int maSinhVien, String lop, double gpa, String ten, String ngaySinh, String diaChi) {

super(ten, ngaySinh, diaChi);

this.maSinhVien = String.format("%04d", maSinhVien);

this.lop = lop;

this.gpa = gpa;

}

@Override

public String toString(){

return this.maSinhVien + " " + super.toString() + " " + this.lop + " " + String.format("%.2f", this.gpa);

}

}

public class Main {

public static void main(String[] args) {

ArrayList<Student> arr = new ArrayList<>();

Scanner sc = new Scanner(System.in);

int n = sc.nextInt();

for(int i = 0; i < n; i++){

sc.nextLine();

String ten = sc.nextLine();

String ngaySinh = sc.nextLine();

String diaChi = sc.nextLine();

String lop = sc.nextLine();

double diem = sc.nextDouble();

Student sinhVien = new Student(i + 1, lop, diem, ten, ngaySinh, diaChi);

sinhVien.chuanHoa();

arr.add(sinhVien);

}

Collections.sort(arr, new Comparator<Student>(){

@Override

public int compare(Student o1, Student o2) {

return o1.getSortedName().compareTo(o2.getSortedName());

}

});

for(Student x : arr){

System.out.println(x);

}

}

}

**Input Format**

Dòng 1 là N : số lượng sinh viên. Các dòng tiếp theo là thông tin sinh viên, mỗi sinh viên được mô tả bằng 5 dòng : Tên, ngày sinh, địa chỉ, lớp, gpa.

**Constraints**

1<=N<=1000;

**Output Format**

In ra danh sách sinh viên sau khi được chuẩn hóa, mã sinh viên tăng tự động từ 0001. Các thông tin viết cách nhau một dấu cách, điểm GPA in ra với 2 số sau dấu phẩy.

**Sample Input 0**

6

trAN Phuong HaI

17/4/2004

Ha Noi

DTVT1

2.50

trAN Phuong TuaN

28/1/2004

Ha Nam

DTVT1

2.50

Nguyen AnH MaNH

11/3/2004

Ha Noi

CNTT1

2.70

pham duC TuaN

21/5/2004

Ha Noi

DTVT1

2.50

trAN VAn LoNG

24/6/2004

Ha Noi

CNTT1

2.80

Luong Ngoc LoNG

12/11/2004

Nam Dinh

CNTT2

3.05

**Sample Output 0**

0001 Tran Phuong Hai 17/04/2004 Ha Noi DTVT1 2.50

0006 Luong Ngoc Long 12/11/2004 Nam Dinh CNTT2 3.05

0005 Tran Van Long 24/06/2004 Ha Noi CNTT1 2.80

0003 Nguyen Anh Manh 11/03/2004 Ha Noi CNTT1 2.70

0004 Pham Duc Tuan 21/05/2004 Ha Noi DTVT1 2.50

0002 Tran Phuong Tuan 28/01/2004 Ha Nam DTVT1 2.50**[Kế thừa - Đa hình]. Bài 2. Kế thừa lớp Person và sắp xếp theo tên**

Xây dựng lớp Person với các thuộc tính : Tên (xâu kí tự không quá 100 kí tự), ngày sinh, địa chỉ và các hàm khởi tạo không tham số (gán các trường là xâu rỗng) và hàm khởi tạo đầy đủ tham số, phương thức toString để trả về thông tin. Lớp Student kế thừa từ lớp Person và có thêm thuộc tính là mã sinh viên, GPA và lớp, ghi đè phương thức toString. Nhập danh sách sinh viên từ bàn phím và in ra màn hình danh sách sinh viên trong đó tên được chuẩn hóa và ngày sinh đưa về đúng chuẩn dd/mm/yyyy. Tên sinh viên được sắp xếp theo thứ tự từ điển tăng dần, thứ tự từ điển của tên được xét từ tên, họ, đệm. Nếu 2 bạn cùng tên thì bạn nào xuất hiện trong danh sách trước được in ra trước.

import java.util.ArrayList;

import java.util.Arrays;

import java.util.Collections;

import java.util.Comparator;

import java.util.InputMismatchException;

import java.util.Scanner;

class Person {

private String ten, ngaySinh, diaChi;

public Person() {

ten = ngaySinh = diaChi = "";

}

public Person(String ten, String ngaySinh, String diaChi) {

this.ten = ten;

this.ngaySinh = ngaySinh;

this.diaChi = diaChi;

}

public void chuanHoa(){

String[] arr = this.ten.split("\\s+");

String res = "";

for(String x : arr){

res += Character.toUpperCase(x.charAt(0));

for(int j = 1; j < x.length(); j++){

res += Character.toLowerCase(x.charAt(j));

}

res += " ";

}

this.ten = res.substring(0, res.length() - 1);

StringBuilder sb = new StringBuilder(this.ngaySinh);

if(sb.charAt(1) == '/') sb.insert(0, "0");

if(sb.charAt(4) == '/') sb.insert(3, "0");

this.ngaySinh = sb.toString();

}

//Nguyen Van Nam => NamNguyenVan

public String getSortedName(){

String[] arr = this.ten.split("\\s+");

String res = arr[arr.length - 1];

for(int i = 0; i < arr.length - 1; i++){

res += arr[i] + " ";

}

return res;

}

@Override

public String toString(){

return this.ten + " " + this.ngaySinh + " " + this.diaChi;

}

}

class Student extends Person{

private String maSinhVien, lop;

private double gpa;

public Student(int maSinhVien, String lop, double gpa, String ten, String ngaySinh, String diaChi) {

super(ten, ngaySinh, diaChi);

this.maSinhVien = String.format("%04d", maSinhVien);

this.lop = lop;

this.gpa = gpa;

}

@Override

public String toString(){

return this.maSinhVien + " " + super.toString() + " " + this.lop + " " + String.format("%.2f", this.gpa);

}

}

public class Main {

public static void main(String[] args) {

ArrayList<Student> arr = new ArrayList<>();

Scanner sc = new Scanner(System.in);

int n = sc.nextInt();

for(int i = 0; i < n; i++){

sc.nextLine();

String ten = sc.nextLine();

String ngaySinh = sc.nextLine();

String diaChi = sc.nextLine();

String lop = sc.nextLine();

double diem = sc.nextDouble();

Student sinhVien = new Student(i + 1, lop, diem, ten, ngaySinh, diaChi);

sinhVien.chuanHoa();

arr.add(sinhVien);

}

Collections.sort(arr, new Comparator<Student>(){

@Override

public int compare(Student o1, Student o2) {

return o1.getSortedName().compareTo(o2.getSortedName());

}

});

for(Student x : arr){

System.out.println(x);

}

}

}

**Input Format**

Dòng 1 là N : số lượng sinh viên. Các dòng tiếp theo là thông tin sinh viên, mỗi sinh viên được mô tả bằng 5 dòng : Tên, ngày sinh, địa chỉ, lớp, gpa.

**Constraints**

1<=N<=1000;

**Output Format**

In ra danh sách sinh viên sau khi được chuẩn hóa, mã sinh viên tăng tự động từ 0001. Các thông tin viết cách nhau một dấu cách, điểm GPA in ra với 2 số sau dấu phẩy.

**Sample Input 0**

6

trAN Phuong HaI

17/4/2004

Ha Noi

DTVT1

2.50

trAN Phuong TuaN

28/1/2004

Ha Nam

DTVT1

2.50

Nguyen AnH MaNH

11/3/2004

Ha Noi

CNTT1

2.70

pham duC TuaN

21/5/2004

Ha Noi

DTVT1

2.50

trAN VAn LoNG

24/6/2004

Ha Noi

CNTT1

2.80

Luong Ngoc LoNG

12/11/2004

Nam Dinh

CNTT2

3.05

**Sample Output 0**

0001 Tran Phuong Hai 17/04/2004 Ha Noi DTVT1 2.50

0006 Luong Ngoc Long 12/11/2004 Nam Dinh CNTT2 3.05

0005 Tran Van Long 24/06/2004 Ha Noi CNTT1 2.80

0003 Nguyen Anh Manh 11/03/2004 Ha Noi CNTT1 2.70

0004 Pham Duc Tuan 21/05/2004 Ha Noi DTVT1 2.50

0002 Tran Phuong Tuan 28/01/2004 Ha Nam DTVT1 2.50

**[Kế thừa - Đa hình]. Bài 39. Sinh Viên và Giáo Viên**

Trường đại học XYZ cần quản lý các đối tượng là sinh viên và giáo viên. Sinh viên gồm các thông tin : mã sinh viên, tên, ngày sinh, địa chỉ, lớp, điểm gpa. Giáo viên gồm các thông tin : mã giáo viên, tên, ngày sinh, địa chỉ, khoa, lương. Thực hiện đọc các thông tin danh sách sinh viên và giáo viên từ bàn phím sau đó chuẩn hóa tên, ngày sinh và in ra danh sách sinh viên sau đó là danh sách giáo viên, biết rằng sinh viên sẽ có mã bắt đầu bằng SV (ví dụ SV112), giáo viên có mã bắt đầu bằng GV (ví dụ GV222).

import java.util.ArrayList;

import java.util.Arrays;

import java.util.Collections;

import java.util.Comparator;

import java.util.InputMismatchException;

import java.util.Scanner;

class Person {

private String ma, ten, ngaySinh, diaChi;

public Person(String ma, String ten, String ngaySinh, String diaChi) {

this.ma = ma;

this.ten = ten;

this.ngaySinh = ngaySinh;

this.diaChi = diaChi;

}

@Override

public String toString(){

return this.ma + " " + this.ten + " " + this.ngaySinh + " " + this.diaChi;

}

public void chuanHoa(){

String[] arr = this.ten.split("\\s+");

String res = "";

for(String x : arr){

res += Character.toUpperCase(x.charAt(0));

for(int j = 1; j < x.length(); j++){

res += Character.toLowerCase(x.charAt(j));

}

res += " ";

}

this.ten = res.substring(0, res.length() - 1);

StringBuilder sb = new StringBuilder(this.ngaySinh);

if(sb.charAt(1) == '/') sb.insert(0, "0");

if(sb.charAt(4) == '/') sb.insert(3, "0");

this.ngaySinh = sb.toString();

}

}

class Student extends Person{

private String lop;

private double gpa;

public Student(String lop, double gpa, String ma, String ten, String ngaySinh, String diaChi) {

super(ma, ten, ngaySinh, diaChi);

this.lop = lop;

this.gpa = gpa;

}

@Override

public String toString(){

return super.toString() + " " + this.lop + " " + String.format("%.2f", this.gpa);

}

}

class Lecturer extends Person{

private String khoa;

private int luong;

public Lecturer(String khoa, int luong, String ma, String ten, String ngaySinh, String diaChi) {

super(ma, ten, ngaySinh, diaChi);

this.khoa = khoa;

this.luong = luong;

}

@Override

public String toString(){

return super.toString() + " " + this.khoa + " " + this.luong;

}

}

//SinhVien, GiaoVien, NhanVien : Subclass, derived class

//Nguoi : superclass, base class

public class Main {

public static void main(String[] args) {

ArrayList<Student> arr1 = new ArrayList<>();

ArrayList<Lecturer> arr2 = new ArrayList<>();

Scanner sc = new Scanner(System.in);

int n = sc.nextInt();

for(int i = 0; i < n; i++){

sc.nextLine();

String ma = sc.nextLine();

String ten = sc.nextLine();

String ngaySinh = sc.nextLine();

String diaChi = sc.nextLine();

if(ma.substring(0, 2).equals("GV")){

String khoa = sc.nextLine();

int luong = sc.nextInt();

Lecturer lec = new Lecturer(khoa, luong, ma, ten, ngaySinh, diaChi);

lec.chuanHoa();

arr2.add(lec);

}

else{

String lop = sc.nextLine();

double gpa = sc.nextDouble();

Student student = new Student(lop, gpa, ma, ten, ngaySinh, diaChi);

student.chuanHoa();

arr1.add(student);

}

}

System.out.println("DANH SACH GIAO VIEN :");

for(Lecturer x : arr2){

System.out.println(x);

}

System.out.println("DANH SACH SINH VIEN :");

for(Student x : arr1){

System.out.println(x);

}

}

}

**Input Format**

Dòng đầu tiên là N : số lượng giáo viên và sinh viên. Các dòng tiếp theo mô tả thông tin của giáo viên hoặc sinh viên, mỗi thông tin gồm 6 dòng, đối với sinh viên 6 dòng gồm : mã sinh viên, tên, ngày sinh, địa chỉ, lớp, điểm gpa, đối với giáo viên 6 dòng gồm : mã giáo viên, tên, ngày sinh, địa chỉ, khoa, lương.

**Constraints**

1<=N<=1000;

**Output Format**

Đầu tiên in ra danh sách giáo viên, mỗi giáo viên in ra thông tin trên 1 dòng, các thông tin cách nhau một dấu cách. Những dòng tiếp theo in ra danh sách sinh viên, mỗi sinh viên in thông tin trên 1 dòng, các thông tin cách nhau một dấu cách, gpa in 2 số sau dấu phẩy.

**Sample Input 0**

9

GV1

Nguyen duC TuaN

4/6/1977

Thai Binh

CNTT

12000000

SV2

Luong VAn HaI

1/6/2004

Thai Binh

CNTT2

2.50

SV3

Nguyen AnH MaNH

14/2/2004

Thai Binh

CNTT2

2.50

GV4

Nguyen AnH HaI

20/3/1974

Ha Nam

KT

20000000

SV5

pham AnH MaNH

8/5/2004

Ha Nam

CNTT1

2.70

SV6

pham Phuong MaNH

18/7/2004

Ha Noi

CNTT2

2.50

GV7

trAN Phuong LoNG

6/2/1979

Ha Noi

ATTT

20000000

SV8

Nguyen VAn HaI

25/8/2004

Nam Dinh

CNTT2

2.70

SV9

Luong Ngoc HaI

16/11/2004

Ha Noi

CNTT1

3.20

**Sample Output 0**

DANH SACH GIAO VIEN :

GV1 Nguyen Duc Tuan 04/06/1977 Thai Binh CNTT 12000000

GV4 Nguyen Anh Hai 20/03/1974 Ha Nam KT 20000000

GV7 Tran Phuong Long 06/02/1979 Ha Noi ATTT 20000000

DANH SACH SINH VIEN :

SV2 Luong Van Hai 01/06/2004 Thai Binh CNTT2 2.50

SV3 Nguyen Anh Manh 14/02/2004 Thai Binh CNTT2 2.50

SV5 Pham Anh Manh 08/05/2004 Ha Nam CNTT1 2.70

SV6 Pham Phuong Manh 18/07/2004 Ha Noi CNTT2 2.50

SV8 Nguyen Van Hai 25/08/2004 Nam Dinh CNTT2 2.70

SV9 Luong Ngoc Hai 16/11/2004 Ha Noi CNTT1 3.20

**[Kế thừa - Đa hình]. Bài 40. Truy vấn sinh viên, giáo viên**

Trường đại học XYZ cần quản lý các đối tượng là sinh viên và giáo viên. Sinh viên gồm các thông tin : mã sinh viên, tên, ngày sinh, địa chỉ, lớp, điểm gpa. Giáo viên gồm các thông tin : mã giáo viên, tên, ngày sinh, địa chỉ, khoa, lương, lớp mà giáo viên này phụ trách chủ nhiệm. Thực hiện đọc các thông tin danh sách sinh viên và giáo viên từ bàn phím sau đó chuẩn hóa tên, ngày sinh và in ra danh sách sinh viên sau đó là danh sách giáo viên, biết rằng sinh viên sẽ có mã bắt đầu bằng SV (ví dụ SV112), giáo viên có mã bắt đầu bằng GV (ví dụ GV222) theo địa chỉ tìm kiếm.

import java.util.ArrayList;

import java.util.Arrays;

import java.util.Collections;

import java.util.Comparator;

import java.util.InputMismatchException;

import java.util.Scanner;

class Person {

private String ma, ten, ngaySinh, diaChi;

public Person(String ma, String ten, String ngaySinh, String diaChi) {

this.ma = ma;

this.ten = ten;

this.ngaySinh = ngaySinh;

this.diaChi = diaChi;

}

@Override

public String toString(){

return this.ma + " " + this.ten + " " + this.ngaySinh + " " + this.diaChi;

}

public String getDiaChi() {

return diaChi;

}

public void chuanHoa(){

String[] arr = this.ten.split("\\s+");

String res = "";

for(String x : arr){

res += Character.toUpperCase(x.charAt(0));

for(int j = 1; j < x.length(); j++){

res += Character.toLowerCase(x.charAt(j));

}

res += " ";

}

this.ten = res.substring(0, res.length() - 1);

StringBuilder sb = new StringBuilder(this.ngaySinh);

if(sb.charAt(1) == '/') sb.insert(0, "0");

if(sb.charAt(4) == '/') sb.insert(3, "0");

this.ngaySinh = sb.toString();

}

}

class Student extends Person{

private String lop;

private double gpa;

public Student(String lop, double gpa, String ma, String ten, String ngaySinh, String diaChi) {

super(ma, ten, ngaySinh, diaChi);

this.lop = lop;

this.gpa = gpa;

}

@Override

public String toString(){

return super.toString() + " " + this.lop + " " + String.format("%.2f", this.gpa);

}

}

class Lecturer extends Person{

private String khoa, lop;

private int luong;

public Lecturer(String khoa, int luong, String lop, String ma, String ten, String ngaySinh, String diaChi) {

super(ma, ten, ngaySinh, diaChi);

this.khoa = khoa;

this.luong = luong;

this.lop = lop;

}

@Override

public String toString(){

return super.toString() + " " + this.khoa + " " + this.luong;

}

}

public class Main {

public static void main(String[] args) {

ArrayList<Student> arr1 = new ArrayList<>();

ArrayList<Lecturer> arr2 = new ArrayList<>();

Scanner sc = new Scanner(System.in);

int n = sc.nextInt(); sc.nextLine();

for(int i = 0; i < n; i++){

String ma = sc.nextLine();

String ten = sc.nextLine();

String ngaySinh = sc.nextLine();

String diaChi = sc.nextLine();

if(ma.substring(0, 2).equals("GV")){

String khoa = sc.nextLine();

int luong = Integer.parseInt(sc.nextLine());

String lop = sc.nextLine();

Lecturer lec = new Lecturer(khoa, luong, lop, ma, ten, ngaySinh, diaChi);

lec.chuanHoa();

arr2.add(lec);

}

else{

String lop = sc.nextLine();

double gpa = Double.parseDouble(sc.nextLine());

Student student = new Student(lop, gpa, ma, ten, ngaySinh, diaChi);

student.chuanHoa();

arr1.add(student);

}

}

String diaChi = sc.nextLine();

System.out.println("DANH SACH GIAO VIEN CO DIA CHI TAI " + diaChi + " :");

for(Lecturer x : arr2){

if(x.getDiaChi().equals(diaChi))

System.out.println(x);

}

System.out.println("DANH SACH SINH VIEN CO DIA CHI TAI " + diaChi + " :");

for(Student x : arr1){

if(x.getDiaChi().equals(diaChi)){

System.out.println(x);

}

}

}

}

import java.util.ArrayList;

import java.util.Arrays;

import java.util.Collections;

import java.util.Comparator;

import java.util.InputMismatchException;

import java.util.Scanner;

class Person {

private String ma, ten, ngaySinh, diaChi;

public Person(String ma, String ten, String ngaySinh, String diaChi) {

this.ma = ma;

this.ten = ten;

this.ngaySinh = ngaySinh;

this.diaChi = diaChi;

}

@Override

public String toString(){

return this.ma + " " + this.ten + " " + this.ngaySinh + " " + this.diaChi;

}

public String getDiaChi() {

return diaChi;

}

public void chuanHoa(){

String[] arr = this.ten.split("\\s+");

String res = "";

for(String x : arr){

res += Character.toUpperCase(x.charAt(0));

for(int j = 1; j < x.length(); j++){

res += Character.toLowerCase(x.charAt(j));

}

res += " ";

}

this.ten = res.substring(0, res.length() - 1);

StringBuilder sb = new StringBuilder(this.ngaySinh);

if(sb.charAt(1) == '/') sb.insert(0, "0");

if(sb.charAt(4) == '/') sb.insert(3, "0");

this.ngaySinh = sb.toString();

}

}

class Student extends Person{

private String lop;

private double gpa;

public Student(String lop, double gpa, String ma, String ten, String ngaySinh, String diaChi) {

super(ma, ten, ngaySinh, diaChi);

this.lop = lop;

this.gpa = gpa;

}

@Override

public String toString(){

return super.toString() + " " + this.lop + " " + String.format("%.2f", this.gpa);

}

}

class Lecturer extends Person{

private String khoa, lop;

private int luong;

public Lecturer(String khoa, int luong, String lop, String ma, String ten, String ngaySinh, String diaChi) {

super(ma, ten, ngaySinh, diaChi);

this.khoa = khoa;

this.luong = luong;

this.lop = lop;

}

@Override

public String toString(){

return super.toString() + " " + this.khoa + " " + this.luong;

}

}

public class Main {

public static void main(String[] args) {

ArrayList<Student> arr1 = new ArrayList<>();

ArrayList<Lecturer> arr2 = new ArrayList<>();

Scanner sc = new Scanner(System.in);

int n = sc.nextInt(); sc.nextLine();

for(int i = 0; i < n; i++){

String ma = sc.nextLine();

String ten = sc.nextLine();

String ngaySinh = sc.nextLine();

String diaChi = sc.nextLine();

if(ma.substring(0, 2).equals("GV")){

String khoa = sc.nextLine();

int luong = Integer.parseInt(sc.nextLine());

String lop = sc.nextLine();

Lecturer lec = new Lecturer(khoa, luong, lop, ma, ten, ngaySinh, diaChi);

lec.chuanHoa();

arr2.add(lec);

}

else{

String lop = sc.nextLine();

double gpa = Double.parseDouble(sc.nextLine());

Student student = new Student(lop, gpa, ma, ten, ngaySinh, diaChi);

student.chuanHoa();

arr1.add(student);

}

}

String diaChi = sc.nextLine();

System.out.println("DANH SACH GIAO VIEN CO DIA CHI TAI " + diaChi + " :");

for(Lecturer x : arr2){

if(x.getDiaChi().equals(diaChi))

System.out.println(x);

}

System.out.println("DANH SACH SINH VIEN CO DIA CHI TAI " + diaChi + " :");

for(Student x : arr1){

if(x.getDiaChi().equals(diaChi)){

System.out.println(x);

}

}

}

}

**Input Format**

Dòng đầu tiên là N : số lượng giáo viên và sinh viên. Các dòng tiếp theo mô tả thông tin của giáo viên hoặc sinh viên, mỗi thông tin gồm 6 dòng, đối với sinh viên 6 dòng gồm : mã sinh viên, tên, ngày sinh, địa chỉ, lớp, điểm gpa, đối với giáo viên 6 dòng gồm : mã giáo viên, tên, ngày sinh, địa chỉ, khoa, lương, lớp mà giáo viên này phụ trách chủ nhiệm. Dòng cuối cùng của input là địa chỉ cần tìm kiếm.

**Constraints**

1<=N<=1000;

**Output Format**

Đầu tiên in ra danh sách giáo viên, mỗi giáo viên in ra thông tin trên 1 dòng, các thông tin cách nhau một dấu cách, không cần in thông tin về lớp mà giáo viên này quản lý. Những dòng tiếp theo in ra danh sách sinh viên, mỗi sinh viên in thông tin trên 1 dòng, các thông tin cách nhau một dấu cách, gpa in 2 số sau dấu phẩy.

**Sample Input 0**

7

GV1

trAN duC TuaN

7/4/1974

Thai Binh

ATTT

20000000

CNTT1

SV2

trAN AnH MaNH

27/10/2004

Ha Nam

DTVT1

2.50

SV3

pham Phuong NAM

8/8/2004

Thai Binh

DTVT2

2.50

GV4

trAN AnH MaNH

24/5/1972

Ha Nam

ATTT

12000000

DTVT2

SV5

trAN Phuong LoNG

8/12/2004

Ha Nam

ATTT3

2.50

SV6

Luong duC LoNG

23/1/2004

Ha Noi

DTVT2

3.20

GV7

Luong AnH TuaN

18/10/1972

Ha Nam

Co khi

25000000

DTVT1

Ha Noi

**Sample Output 0**

DANH SACH GIAO VIEN CO DIA CHI TAI Ha Noi :

DANH SACH SINH VIEN CO DIA CHI TAI Ha Noi :

SV6 Luong Duc Long 23/01/2004 Ha Noi DTVT2 3.20

**[Kế thừa - Đa hình]. Bài 41. Sắp xếp theo lương, Gpa**

Trường đại học XYZ cần quản lý các đối tượng là sinh viên và giáo viên. Sinh viên gồm các thông tin : mã sinh viên, tên, ngày sinh, địa chỉ, lớp, điểm gpa. Giáo viên gồm các thông tin : mã giáo viên, tên, ngày sinh, địa chỉ, khoa, lương. Thực hiện đọc các thông tin danh sách sinh viên và giáo viên từ bàn phím sau đó chuẩn hóa tên, ngày sinh và in ra danh sách sinh viên sau đó là danh sách giáo viên, biết rằng sinh viên sẽ có mã bắt đầu bằng SV (ví dụ SV112), giáo viên có mã bắt đầu bằng GV (ví dụ GV222). Đầu tiên in ra danh sách giáo viên theo lương giảm dần, nếu 2 giáo viên có cùng lương thì in theo mã giáo viên tăng dần(từ điển), tiếp đó in ra danh sách sinh viên theo gpa giảm dần, nếu 2 sinh viên có cùng gpa thì in theo mã sinh viên tăng dần(từ điển).

import java.util.ArrayList;

import java.util.Arrays;

import java.util.Collections;

import java.util.Comparator;

import java.util.InputMismatchException;

import java.util.Scanner;

class Person {

private String ma, ten, ngaySinh, diaChi;

public Person(String ma, String ten, String ngaySinh, String diaChi) {

this.ma = ma;

this.ten = ten;

this.ngaySinh = ngaySinh;

this.diaChi = diaChi;

}

@Override

public String toString(){

return this.ma + " " + this.ten + " " + this.ngaySinh + " " + this.diaChi;

}

public String getDiaChi() {

return diaChi;

}

public String getMa() {

return ma;

}

public void chuanHoa(){

String[] arr = this.ten.split("\\s+");

String res = "";

for(String x : arr){

res += Character.toUpperCase(x.charAt(0));

for(int j = 1; j < x.length(); j++){

res += Character.toLowerCase(x.charAt(j));

}

res += " ";

}

this.ten = res.substring(0, res.length() - 1);

StringBuilder sb = new StringBuilder(this.ngaySinh);

if(sb.charAt(1) == '/') sb.insert(0, "0");

if(sb.charAt(4) == '/') sb.insert(3, "0");

this.ngaySinh = sb.toString();

}

}

class Student extends Person{

private String lop;

private double gpa;

public Student(String lop, double gpa, String ma, String ten, String ngaySinh, String diaChi) {

super(ma, ten, ngaySinh, diaChi);

this.lop = lop;

this.gpa = gpa;

}

public double getGpa() {

return gpa;

}

@Override

public String toString(){

return super.toString() + " " + this.lop + " " + String.format("%.2f", this.gpa);

}

}

class Lecturer extends Person{

private String khoa;

private int luong;

public Lecturer(String khoa, int luong, String ma, String ten, String ngaySinh, String diaChi) {

super(ma, ten, ngaySinh, diaChi);

this.khoa = khoa;

this.luong = luong;

}

public int getLuong() {

return luong;

}

@Override

public String toString(){

return super.toString() + " " + this.khoa + " " + this.luong;

}

}

public class Main {

public static void main(String[] args) {

ArrayList<Student> arr1 = new ArrayList<>();

ArrayList<Lecturer> arr2 = new ArrayList<>();

Scanner sc = new Scanner(System.in);

int n = sc.nextInt(); sc.nextLine();

for(int i = 0; i < n; i++){

String ma = sc.nextLine();

String ten = sc.nextLine();

String ngaySinh = sc.nextLine();

String diaChi = sc.nextLine();

if(ma.substring(0, 2).equals("GV")){

String khoa = sc.nextLine();

int luong = Integer.parseInt(sc.nextLine());

Lecturer lec = new Lecturer(khoa, luong, ma, ten, ngaySinh, diaChi);

lec.chuanHoa();

arr2.add(lec);

}

else{

String lop = sc.nextLine();

double gpa = Double.parseDouble(sc.nextLine());

Student student = new Student(lop, gpa, ma, ten, ngaySinh, diaChi);

student.chuanHoa();

arr1.add(student);

}

}

Collections.sort(arr2, new Comparator<Lecturer>(){

@Override

public int compare(Lecturer o1, Lecturer o2) {

if(o1.getLuong() != o2.getLuong())

return o2.getLuong() - o1.getLuong();

else

return o1.getMa().compareTo(o2.getMa());

}

});

System.out.println("DANH SACH GIAO VIEN :");

for(Lecturer x : arr2){

System.out.println(x);

}

Collections.sort(arr1, new Comparator<Student>(){

@Override

public int compare(Student o1, Student o2) {

if(o1.getGpa() != o2.getGpa()){

if(o1.getGpa() > o2.getGpa()) return -1;

return 1;

}

else{

return o1.getMa().compareTo(o2.getMa());

}

}

});

System.out.println("DANH SACH SINH VIEN :");

for(Student x : arr1){

System.out.println(x);

}

}

}

**Input Format**

Dòng đầu tiên là N : số lượng giáo viên và sinh viên. Các dòng tiếp theo mô tả thông tin của giáo viên hoặc sinh viên, mỗi thông tin gồm 6 dòng, đối với sinh viên 6 dòng gồm : mã sinh viên, tên, ngày sinh, địa chỉ, lớp, điểm gpa, đối với giáo viên 6 dòng gồm : mã giáo viên, tên, ngày sinh, địa chỉ, khoa, lương.

**Constraints**

1<=N<=1000;

**Output Format**

Đầu tiên in ra danh sách giáo viên, mỗi giáo viên in ra thông tin trên 1 dòng, các thông tin cách nhau một dấu cách. Những dòng tiếp theo in ra danh sách sinh viên, mỗi sinh viên in thông tin trên 1 dòng, các thông tin cách nhau một dấu cách, gpa in 2 số sau dấu phẩy.

**Sample Input 0**

5

GV1

pham duC LoNG

4/3/1976

Ha Nam

CNTT

18000000

SV2

Vu AnH LoNG

3/11/2004

Hai Duong

DTVT1

2.70

SV3

trAN AnH NAM

27/11/2004

Nam Dinh

ATTT3

3.05

GV4

Nguyen VAn MaNH

12/4/1978

Ha Noi

Co khi

25000000

SV5

Luong Phuong HaI

3/4/2004

Nam Dinh

DTVT2

2.50

**Sample Output 0**

DANH SACH GIAO VIEN :

GV4 Nguyen Van Manh 12/04/1978 Ha Noi Co khi 25000000

GV1 Pham Duc Long 04/03/1976 Ha Nam CNTT 18000000

DANH SACH SINH VIEN :

SV3 Tran Anh Nam 27/11/2004 Nam Dinh ATTT3 3.05

SV2 Vu Anh Long 03/11/2004 Hai Duong DTVT1 2.70

SV5 Luong Phuong Hai 03/04/2004 Nam Dinh DTVT2 2.50

**[Kế thừa - Đa hình]. Bài 42. Giáo viên chủ nhiệm**

Trường đại học XYZ cần quản lý các đối tượng là sinh viên và giáo viên. Sinh viên gồm các thông tin : mã sinh viên, tên, ngày sinh, địa chỉ, lớp, điểm gpa. Giáo viên gồm các thông tin : mã giáo viên, tên, ngày sinh, địa chỉ, khoa, lương, lớp mà giáo viên này phụ trách. Thực hiện đọc các thông tin danh sách sinh viên và giáo viên từ bàn phím sau đó chuẩn hóa tên, ngày sinh, biết rằng sinh viên sẽ có mã bắt đầu bằng SV (ví dụ SV112), giáo viên có mã bắt đầu bằng GV (ví dụ GV222) sau đó hiển thị giáo viên phụ trách và các sinh viên thuộc về 1 lớp theo truy vấn.

import java.util.\*;

class Person {

private String ten, ngaySinh, diaChi;

public Person() {

this.ten = this.ngaySinh = this.diaChi = "";

}

public Person(String ten, String ngaySinh, String diaChi) {

this.ten = ten;

this.ngaySinh = ngaySinh;

this.diaChi = diaChi;

}

public void chuanHoa(){

StringBuilder sb = new StringBuilder("");

String[] arr = this.ten.split("\\s+");

for(String x : arr){

sb.append(Character.toUpperCase(x.charAt(0)));

for(int i = 1; i < x.length(); i++){

sb.append(Character.toLowerCase(x.charAt(i)));

}

sb.append(" ");

}

sb.deleteCharAt(sb.length() - 1);

this.ten = sb.toString();

StringBuilder sb1 = new StringBuilder(this.ngaySinh);

if(sb1.charAt(1) == '/')

sb1.insert(0, "0");

if(sb1.charAt(4) == '/')

sb1.insert(3, "0");

this.ngaySinh = sb1.toString();

}

public String getTen() {

return ten;

}

@Override

public String toString(){

return this.ten + " " + this.ngaySinh + " " + this.diaChi;

}

}

class Student extends Person{

private String maSinhVien, lop;

private double gpa;

public Student(String maSinhVien, String lop, double gpa, String ten, String ngaySinh, String diaChi) {

super(ten, ngaySinh, diaChi);

this.maSinhVien = maSinhVien;

this.lop = lop;

this.gpa = gpa;

}

public String getMaSinhVien() {

return maSinhVien;

}

public String getLop() {

return lop;

}

public double getGpa() {

return gpa;

}

@Override

public String toString(){

return this.maSinhVien + " " + super.toString() + " " + this.lop + " " + String.format("%.2f", this.gpa);

}

}

class Lecturer extends Person{

private String maGiangVien, khoa, lop;

private int luong;

public Lecturer(String maGiangVien, String khoa, String lop, int luong, String ten, String ngaySinh, String diaChi) {

super(ten, ngaySinh, diaChi);

this.maGiangVien = maGiangVien;

this.khoa = khoa;

this.lop = lop;

this.luong = luong;

}

public String getLop() {

return lop;

}

public String getMaGiangVien() {

return maGiangVien;

}

public int getLuong() {

return luong;

}

@Override

public String toString(){

return this.maGiangVien + " " + super.toString() + " " + this.khoa + " " + this.luong + " " + this.lop;

}

}

public class Main {

public static void main(String[] args) {

Scanner sc = new Scanner(System.in);

int n = sc.nextInt();

List<Student> list1 = new ArrayList<>();

List<Lecturer> list2 = new ArrayList<>();

sc.nextLine();

for(int i = 0; i < n; i++){

String ma = sc.nextLine();

if(ma.substring(0,2).equals("GV")){

String ten = sc.nextLine();

String ngaySinh = sc.nextLine();

String diaChi = sc.nextLine();

String khoa = sc.nextLine();

int luong = Integer.parseInt(sc.nextLine());

String lop = sc.nextLine();

Lecturer l = new Lecturer(ma, khoa, lop, luong, ten, ngaySinh, diaChi);

l.chuanHoa();

list2.add(l);

}

else{

String ten = sc.nextLine();

String ngaySinh = sc.nextLine();

String diaChi = sc.nextLine();

String lop = sc.nextLine();

double gpa = Double.parseDouble(sc.nextLine());

Student s = new Student(ma, lop, gpa, ten, ngaySinh, diaChi);

s.chuanHoa();

list1.add(s);

}

}

String lop = sc.nextLine();

System.out.println("DANH SACH GIAO VIEN PHU TRACH LOP " + lop + " :");

for(Lecturer x : list2){

if(x.getLop().equals(lop))

System.out.println(x);

}

System.out.println("DANH SACH SINH VIEN LOP " + lop + " :");

for(Student x : list1){

if(x.getLop().equals(lop))

System.out.println(x);

}

}

}

**Input Format**

Dòng đầu tiên là N : số lượng giáo viên và sinh viên. Các dòng tiếp theo mô tả thông tin của giáo viên hoặc sinh viên, mỗi thông tin gồm 6 dòng, đối với sinh viên 6 dòng gồm : mã sinh viên, tên, ngày sinh, địa chỉ, lớp, điểm gpa, đối với giáo viên 7 dòng gồm : mã giáo viên, tên, ngày sinh, địa chỉ, khoa, lương, lớp phụ trách. Dòng cuối cùng trong input là tên lớp cần truy vấn.

**Constraints**

1<=N<=1000;

**Output Format**

Đầu tiên in ra giáo viên phụ trách lớp, mỗi giáo viên in ra thông tin trên 1 dòng, các thông tin cách nhau một dấu cách, một lớp có thể có nhiều giáo viên cùng phụ trách. Khi đó hãy liệt kê giáo viên theo danh sách. Những dòng tiếp theo in ra danh sách sinh viên, mỗi sinh viên in thông tin trên 1 dòng, các thông tin cách nhau một dấu cách, gpa in 2 số sau dấu phẩy theo danh sách.

**Sample Input 0**

8

GV1

Nguyen VAn TuaN

6/2/1975

Nam Dinh

DTVT

25000000

CNTT1

SV2

Vu AnH MaNH

13/10/2004

Ha Noi

DTVT1

2.70

SV3

trAN Phuong TuaN

5/9/2004

Hai Duong

ATTT3

2.80

GV4

trAN duC HaI

14/12/1973

Ha Nam

Co khi

25000000

CNTT2

SV5

Nguyen Ngoc TuaN

13/11/2004

Hai Duong

CNTT1

3.05

SV6

Luong duC LoNG

6/1/2004

Hai Duong

CNTT2

2.70

GV7

Nguyen Phuong TuaN

25/4/1974

Nam Dinh

Co khi

12000000

DTVT1

SV8

Luong duC NAM

3/2/2004

Ha Noi

CNTT1

2.50

CNTT1

**Sample Output 0**

DANH SACH GIAO VIEN PHU TRACH LOP CNTT1 :

GV1 Nguyen Van Tuan 06/02/1975 Nam Dinh DTVT 25000000 CNTT1

DANH SACH SINH VIEN LOP CNTT1 :

SV5 Nguyen Ngoc Tuan 13/11/2004 Hai Duong CNTT1 3.05

SV8 Luong Duc Nam 03/02/2004 Ha Noi CNTT1 2.50

**[Kế thừa - Đa hình]. Bài 43. Vehicle**

Một cửa hàng bán oto, xe máy cần quản lý các loại xe máy và xe ô tô. Trong đó xe máy có những thông tin : mã xe, tên xe, hãng, màu sắc, tốc độ tối đa, giá bán. Xe ô tô có những thông tin : mã xe, tên xe, hãng, màu sắc, mã lực, giá bán. Nhập danh sách các phương tiện và tiến hành liệt kê các xe theo hãng cần tìm kiếm. Biết rằng xe máy có mã bắt đầu bằng XM (ví dụ XM001), và ô tô có mã bắt đầu bằng OTO (ví dụ OTO521)

import java.util.Scanner;

import java.util.ArrayList;

class Vehicle {

private String ma, ten, hang, mauSac;

private int giaBan;

public Vehicle(String ma, String ten, String hang, String mauSac, int giaBan) {

this.ma = ma;

this.ten = ten;

this.hang = hang;

this.mauSac = mauSac;

this.giaBan = giaBan;

}

public int getGiaBan(){

return this.giaBan;

}

public String getHang() {

return hang;

}

@Override

public String toString() {

return this.ma + " " + this.ten + " " + this.hang + " " + this.mauSac;

}

}

class Oto extends Vehicle{

private int maLuc;

public Oto(int maLuc, String ma, String ten, String hang, String mauSac, int giaBan) {

super(ma, ten, hang, mauSac, giaBan);

this.maLuc = maLuc;

}

@Override

public String toString(){

return super.toString() + " " + this.maLuc + " " + super.getGiaBan();

}

}

class XeMay extends Vehicle{

private int tocDo;

public XeMay(int tocDo, String ma, String ten, String hang, String mauSac, int giaBan) {

super(ma, ten, hang, mauSac, giaBan);

this.tocDo = tocDo;

}

@Override

public String toString(){

return super.toString() + " " + this.tocDo + " " + super.getGiaBan();

}

}

public class Main {

public static void main(String[] args) {

Scanner sc = new Scanner(System.in);

int n = Integer.parseInt(sc.nextLine());

ArrayList<Vehicle> arr = new ArrayList<>();

for(int i = 0; i < n; i++){

String ma = sc.nextLine();

String ten = sc.nextLine();

String hang = sc.nextLine();

String mau = sc.nextLine();

int tmp = Integer.parseInt(sc.nextLine());

int gia = Integer.parseInt(sc.nextLine());

if(ma.substring(0, 2).equals("XM")){

XeMay xeMay = new XeMay(tmp, ma, ten, hang, mau, gia);

arr.add(xeMay);

}

else{

Oto oto = new Oto(tmp, ma, ten, hang, mau, gia);

arr.add(oto);

}

}

String hangXe = sc.nextLine();

System.out.println("DANH SACH XE HANG " + hangXe + " :");

for(Vehicle x : arr){

if((x instanceof Oto) && x.getHang().equals(hangXe)){

System.out.println(x);

}

}

for(Vehicle x : arr){

if((x instanceof XeMay) && x.getHang().equals(hangXe)){

System.out.println(x);

}

}

}

}

**Input Format**

Dòng đầu tiên chứa N : số lượng phương tiện. Các dòng tiếp theo mô tả phương tiện, nếu là xe máy thì gồm 6 dòng : mã xe, tên xe, hãng, màu sắc, tốc độ tối đa, giá bán, nếu là ô tô thì gồm 6 dòng : mã xe, tên xe, hãng, màu sắc, mã lực, giá bán. Dòng cuối cùng là hãng xe cần tìm kiếm

**Constraints**

1<=N<=1000;

**Output Format**

In ra xe máy hoặc oto có hãng trùng với hãng tìm kiếm theo thứ tự xuất hiện, các thông tin của phương tiện được in cách nhau một dấu cách. Danh sách ô tô được liệt kê trước danh sách xe máy.

**Sample Input 0**

6

OTO1

TU2

NISSAN

Do

186

1200

XM2

SYM125

SYM

Do

200

100

XM3

SYM125

SYM

Vang

200

30

OTO4

F89

FORD

Do

204

850

XM5

HON112

HONDA

Trang

320

25

XM6

Z1000

KAWASAKI

Xanh

320

40

NISSAN

**Sample Output 0**

DANH SACH XE HANG NISSAN :

OTO1 TU2 NISSAN Do 186 1200

**[Kế thừa - Đa hình]. Bài 44. Giá bán giảm dần**

Một cửa hàng bán oto, xe máy cần quản lý các loại xe máy và xe ô tô. Trong đó xe máy có những thông tin : mã xe, tên xe, hãng, màu sắc, tốc độ tối đa, giá bán. Xe ô tô có những thông tin : mã xe, tên xe, hãng, màu sắc, mã lực, giá bán. Nhập danh sách các phương tiện và tiến hành liệt kê các xe theo thứ tự giá bán giảm dần, nếu 2 xe có cùng giá bán thì liệt kê theo thứ tự mã xe tăng dần(từ điển). Biết rằng xe máy có mã bắt đầu bằng XM (ví dụ XM001), và ô tô có mã bắt đầu bằng OTO (ví dụ OTO521)

import java.util.ArrayList;

import java.util.Scanner;

import java.util.Comparator;

import java.util.Collections;

class Vehicle {

private String ma, ten, hang, mauSac;

private int giaBan;

public Vehicle(String ma, String ten, String hang, String mauSac, int giaBan) {

this.ma = ma;

this.ten = ten;

this.hang = hang;

this.mauSac = mauSac;

this.giaBan = giaBan;

}

public int getGiaBan(){

return this.giaBan;

}

public String getHang() {

return hang;

}

public String getMa() {

return ma;

}

@Override

public String toString() {

return this.ma + " " + this.ten + " " + this.hang + " " + this.mauSac;

}

}

class XeMay extends Vehicle{

private int tocDo;

public XeMay(int tocDo, String ma, String ten, String hang, String mauSac, int giaBan) {

super(ma, ten, hang, mauSac, giaBan);

this.tocDo = tocDo;

}

@Override

public String toString(){

return super.toString() + " " + this.tocDo + " " + super.getGiaBan();

}

}

class Oto extends Vehicle{

private int maLuc;

public Oto(int maLuc, String ma, String ten, String hang, String mauSac, int giaBan) {

super(ma, ten, hang, mauSac, giaBan);

this.maLuc = maLuc;

}

@Override

public String toString(){

return super.toString() + " " + this.maLuc + " " + super.getGiaBan();

}

}

class SortXeMayByGia implements Comparator<XeMay>{

@Override

public int compare(XeMay o1, XeMay o2) {

if(o1.getGiaBan() != o2.getGiaBan())

return o2.getGiaBan() - o1.getGiaBan();

else

return o1.getMa().compareTo(o2.getMa());

}

}

class SortOtoByGia implements Comparator<Oto>{

@Override

public int compare(Oto o1, Oto o2) {

if(o1.getGiaBan() != o2.getGiaBan())

return o2.getGiaBan() - o1.getGiaBan();

else

return o1.getMa().compareTo(o2.getMa());

}

}

public class Main {

public static void main(String[] args) {

Scanner sc = new Scanner(System.in);

int n = Integer.parseInt(sc.nextLine());

ArrayList<XeMay> arr1 = new ArrayList<>();

ArrayList<Oto> arr2 = new ArrayList<>();

for(int i = 0; i < n; i++){

String ma = sc.nextLine();

String ten = sc.nextLine();

String hang = sc.nextLine();

String mau = sc.nextLine();

int tmp = Integer.parseInt(sc.nextLine());

int gia = Integer.parseInt(sc.nextLine());

if(ma.substring(0, 2).equals("XM")){

XeMay xeMay = new XeMay(tmp, ma, ten, hang, mau, gia);

arr1.add(xeMay);

}

else{

Oto oto = new Oto(tmp, ma, ten, hang, mau, gia);

arr2.add(oto);

}

}

Collections.sort(arr1, new SortXeMayByGia());

Collections.sort(arr2, new SortOtoByGia());

System.out.println("DANH SACH OTO :");

for(Oto x : arr2){

System.out.println(x);

}

System.out.println("DANH SACH XE MAY :");

for(XeMay x : arr1){

System.out.println(x);

}

}

}

**Input Format**

Dòng đầu tiên chứa N : số lượng phương tiện. Các dòng tiếp theo mô tả phương tiện, nếu là xe máy thì gồm 6 dòng : mã xe, tên xe, hãng, màu sắc, tốc độ tối đa, giá bán, nếu là ô tô thì gồm 6 dòng : mã xe, tên xe, hãng, màu sắc, mã lực, giá bán.

**Constraints**

1<=N<=1000;

**Output Format**

In ra danh sách ô tô sau đó in ra danh sách xe máy, các thông tin của phương tiện được in cách nhau một dấu cách

**Sample Input 0**

6

OTO1

TU2

NISSAN

Den

196

1200

XM2

SYM125

SYM

Den

200

30

XM3

SYM125

SYM

Den

200

40

OTO4

TOY555

TOYOTA

Vang

186

600

XM5

Z1000

KAWASAKI

Xanh

180

25

XM6

SUZ221

SUZUKI

Trang

220

30

**Sample Output 0**

DANH SACH OTO :

OTO1 TU2 NISSAN Den 196 1200

OTO4 TOY555 TOYOTA Vang 186 600

DANH SACH XE MAY :

XM3 SYM125 SYM Den 200 40

XM2 SYM125 SYM Den 200 30

XM6 SUZ221 SUZUKI Trang 220 30

XM5 Z1000 KAWASAKI Xanh 180 25

**[Kế thừa - Đa hình]. Bài 45. Mua xe**

Một cửa hàng bán oto, xe máy cần quản lý các loại xe máy và xe ô tô. Trong đó xe máy có những thông tin : mã xe, tên xe, hãng, màu sắc, tốc độ tối đa, giá bán. Xe ô tô có những thông tin : mã xe, tên xe, hãng, màu sắc, mã lực, giá bán. Nhập danh sách các phương tiện và tiến hành liệt kê các xe có giá bán trong khoảng tìm kiếm. Biết rằng xe máy có mã bắt đầu bằng XM (ví dụ XM001), và ô tô có mã bắt đầu bằng OTO (ví dụ OTO521)

import java.util.ArrayList;

import java.util.Collections;

import java.util.List;

import java.util.Scanner;

import java.util.Comparator;

class Vehicle {

private String ma, hang, ten, mauSac;

private int giaBan;

public Vehicle() {

}

public Vehicle(String ma, String hang, String ten, String mauSac, int giaBan) {

this.ma = ma;

this.hang = hang;

this.ten = ten;

this.mauSac = mauSac;

this.giaBan = giaBan;

}

public int getGiaBan() {

return giaBan;

}

public String getHang() {

return hang;

}

@Override

public String toString() {

return this.ma + " " + this.ten + " " + this.hang + " " + this.mauSac;

}

public String getMa(){

return this.ma;

}

}

class XeMay extends Vehicle{

private int tocDo;

public XeMay(int tocDo, String ma, String hang, String ten, String mauSac, int giaBan) {

super(ma, hang, ten, mauSac, giaBan);

this.tocDo = tocDo;

}

@Override

public String toString(){

return super.toString() + " " + this.tocDo + " " + super.getGiaBan();

}

}

class Oto extends Vehicle{

private int maLuc;

public Oto(int maLuc, String ma, String hang, String ten, String mauSac, int giaBan) {

super(ma, hang, ten, mauSac, giaBan);

this.maLuc = maLuc;

}

@Override

public String toString(){

return super.toString() + " " + this.maLuc + " " + super.getGiaBan();

}

}

/\*\*

\*

\* @author Andrew

\*/

class SortXeMayByGia implements Comparator<XeMay>{

public int compare(XeMay a, XeMay b){

if(a.getGiaBan() != b.getGiaBan())

return b.getGiaBan() - a.getGiaBan();

return a.getMa().compareTo(b.getMa());

}

}

class SortOtoByGia implements Comparator<Oto>{

public int compare(Oto a, Oto b){

if(a.getGiaBan() != b.getGiaBan())

return b.getGiaBan() - a.getGiaBan();

return a.getMa().compareTo(b.getMa());

}

}

/\*\*

\*

\* @author Andrew

\*/

public class Main {

public static void main(String[] args) {

Scanner sc = new Scanner(System.in);

int n = sc.nextInt();

List<XeMay> arr1 = new ArrayList<>();

List<Oto> arr2 = new ArrayList<>();

sc.nextLine();

for(int i = 0; i < n; i++){

String ma = sc.nextLine();

if(ma.substring(0, 2).equals("XM")){

String ten = sc.nextLine();

String hang = sc.nextLine();

String mau = sc.nextLine();

int tocDo = Integer.parseInt(sc.nextLine());

int giaBan = Integer.parseInt(sc.nextLine());

XeMay m1 = new XeMay(tocDo, ma, hang, ten, mau, giaBan);

arr1.add(m1);

}

else{

String ten = sc.nextLine();

String hang = sc.nextLine();

String mau = sc.nextLine();

int maLuc = Integer.parseInt(sc.nextLine());

int giaBan = Integer.parseInt(sc.nextLine());

Oto o1 = new Oto(maLuc, ma, hang, ten, mau, giaBan);

arr2.add(o1);

}

}

int minPrice = sc.nextInt();

int maxPrice = sc.nextInt();

System.out.println("DANH SACH OTO :");

for(Oto x : arr2){

if(x.getGiaBan() >= minPrice && x.getGiaBan() <= maxPrice)

System.out.println(x);

}

System.out.println("DANH SACH XE MAY :");

for(XeMay x : arr1){

if(x.getGiaBan() >= minPrice && x.getGiaBan() <= maxPrice)

System.out.println(x);

}

}

}

**Input Format**

Dòng đầu tiên chứa N : số lượng phương tiện. Các dòng tiếp theo mô tả phương tiện, nếu là xe máy thì gồm 6 dòng : mã xe, tên xe, hãng, màu sắc, tốc độ tối đa, giá bán, nếu là ô tô thì gồm 6 dòng : mã xe, tên xe, hãng, màu sắc, mã lực, giá bán. Dòng cuối cùng là khoảng giá cần tìm kiếm.

**Constraints**

1<=N<=1000;

**Output Format**

In ra các phương tiện có giá bán trong khoảng tìm kiếm. Đầu tiên liệt kê các xe oto sau đó liệt kê các xe máy. Các thông tin của phương tiện được in cách nhau một dấu cách

**Sample Input 0**

5

OTO1

TOY555

TOYOTA

Vang

220

600

XM2

Ex

YAMAHA

Trang

200

30

XM3

HON112

HONDA

Xanh

180

30

OTO4

TOY555

TOYOTA

Do

186

600

XM5

Ex

YAMAHA

Xanh

220

40

564 1064

**Sample Output 0**

DANH SACH OTO :

OTO1 TOY555 TOYOTA Vang 220 600

OTO4 TOY555 TOYOTA Do 186 600

DANH SACH XE MAY :

**[Kế thừa - Đa hình]. Bài 46. Tìm kiếm xe**

Một cửa hàng bán oto, xe máy cần quản lý các loại xe máy và xe ô tô. Trong đó xe máy có những thông tin : mã xe, tên xe, hãng, màu sắc, tốc độ tối đa, giá bán. Xe ô tô có những thông tin : mã xe, tên xe, hãng, màu sắc, mã lực, giá bán. Nhập danh sách các phương tiện và tiến hành liệt kê các xe có tên tìm kiếm theo thứ tự xuất hiện trong danh sách. Biết rằng xe máy có mã bắt đầu bằng XM (ví dụ XM001), và ô tô có mã bắt đầu bằng OTO (ví dụ OTO521)

import java.util.ArrayList;

import java.util.Collections;

import java.util.List;

import java.util.Scanner;

import java.util.Comparator;

class Vehicle {

private String ma, hang, ten, mauSac;

private int giaBan;

public Vehicle() {

}

public Vehicle(String ma, String hang, String ten, String mauSac, int giaBan) {

this.ma = ma;

this.hang = hang;

this.ten = ten;

this.mauSac = mauSac;

this.giaBan = giaBan;

}

public int getGiaBan() {

return giaBan;

}

public String getHang() {

return hang;

}

@Override

public String toString() {

return this.ma + " " + this.ten + " " + this.hang + " " + this.mauSac;

}

public String getMa(){

return this.ma;

}

public String getTen(){

return this.ten;

}

}

class XeMay extends Vehicle{

private int tocDo;

public XeMay(int tocDo, String ma, String hang, String ten, String mauSac, int giaBan) {

super(ma, hang, ten, mauSac, giaBan);

this.tocDo = tocDo;

}

@Override

public String toString(){

return super.toString() + " " + this.tocDo + " " + super.getGiaBan();

}

}

class Oto extends Vehicle{

private int maLuc;

public Oto(int maLuc, String ma, String hang, String ten, String mauSac, int giaBan) {

super(ma, hang, ten, mauSac, giaBan);

this.maLuc = maLuc;

}

@Override

public String toString(){

return super.toString() + " " + this.maLuc + " " + super.getGiaBan();

}

}

/\*\*

\*

\* @author Andrew

\*/

class SortXeMayByGia implements Comparator<XeMay>{

public int compare(XeMay a, XeMay b){

if(a.getGiaBan() != b.getGiaBan())

return b.getGiaBan() - a.getGiaBan();

return a.getMa().compareTo(b.getMa());

}

}

class SortOtoByGia implements Comparator<Oto>{

public int compare(Oto a, Oto b){

if(a.getGiaBan() != b.getGiaBan())

return b.getGiaBan() - a.getGiaBan();

return a.getMa().compareTo(b.getMa());

}

}

/\*\*

\*

\* @author Andrew

\*/

public class Main {

public static void main(String[] args) {

Scanner sc = new Scanner(System.in);

int n = sc.nextInt();

List<XeMay> arr1 = new ArrayList<>();

List<Oto> arr2 = new ArrayList<>();

sc.nextLine();

for(int i = 0; i < n; i++){

String ma = sc.nextLine();

if(ma.substring(0, 2).equals("XM")){

String ten = sc.nextLine();

String hang = sc.nextLine();

String mau = sc.nextLine();

int tocDo = Integer.parseInt(sc.nextLine());

int giaBan = Integer.parseInt(sc.nextLine());

XeMay m1 = new XeMay(tocDo, ma, hang, ten, mau, giaBan);

arr1.add(m1);

}

else{

String ten = sc.nextLine();

String hang = sc.nextLine();

String mau = sc.nextLine();

int maLuc = Integer.parseInt(sc.nextLine());

int giaBan = Integer.parseInt(sc.nextLine());

Oto o1 = new Oto(maLuc, ma, hang, ten, mau, giaBan);

arr2.add(o1);

}

}

String s = sc.nextLine();

System.out.println("DANH SACH OTO :");

for(Oto x : arr2){

if(x.getTen().equals(s))

System.out.println(x);

}

System.out.println("DANH SACH XE MAY :");

for(XeMay x : arr1){

if(x.getTen().equals(s))

System.out.println(x);

}

}

}

**Input Format**

Dòng đầu tiên chứa N : số lượng phương tiện. Các dòng tiếp theo mô tả phương tiện, nếu là xe máy thì gồm 6 dòng : mã xe, tên xe, hãng, màu sắc, tốc độ tối đa, giá bán, nếu là ô tô thì gồm 6 dòng : mã xe, tên xe, hãng, màu sắc, mã lực, giá bán. Dòng cuối cùng là tên xe cần tìm kiếm.

**Constraints**

1<=N<=1000;

**Output Format**

In ra các phương tiện có tên tìm kiếm. Đầu tiên liệt kê các xe oto sau đó liệt kê các xe máy. Các thông tin của phương tiện được in cách nhau một dấu cách

**Sample Input 0**

5

OTO1

HONDAT5

HONDA

Vang

186

850

XM2

HON112

HONDA

Xanh

200

40

XM3

HON112

HONDA

Vang

180

100

OTO4

TOY555

TOYOTA

Vang

150

2400

XM5

Ex

YAMAHA

Trang

220

28

TOY555

**Sample Output 0**

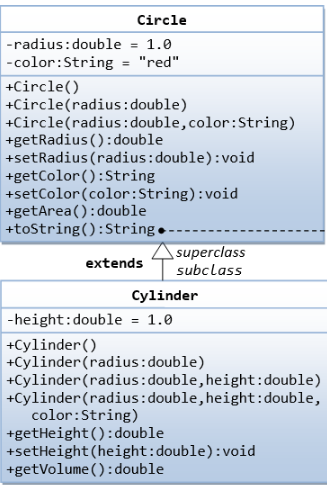
DANH SACH OTO :

OTO4 TOY555 TOYOTA Vang 150 2400

DANH SACH XE MAY :

**[Kế thừa - Đa hình]. Bài 47. Circle và Cylinder**

Cho lớp Circle và lớp Cylinder kế thừa từ lớp Circle, thiết kế 2 lớp trên theo thiết kế dưới đây.



Cho danh sách thông tin về bán kính, chiều cao và màu sắc của các hình trụ, bạn hãy tính và in ra thể tích của hình trụ (Lấy PI là 3.14) đó và sắp xếp giảm dần theo thể tích, nếu 2 hình trụ có cùng thể tính thì sắp xếp theo màu sắc tăng dần về thứ tự từ điển.

**Input Format**

* Dòng 1 là N : số lượng hình trụ
* N dòng tiếp theo mô tả hình trụ gồm : Màu sắc, bán kính, chiều cao

**Constraints**

* 1<=N<=1000
* Màu sắc có một từ, bán kính và chiều cao không quá 1000.

**Output Format**

* In ra danh sách hình trụ sau khi sắp xếp, chiều cao, bán kính và thể tích lấy 2 số sau dấu phẩy.

**Sample Input 0**

9

Red 19 104

White 13 156

White 16 168

Green 13 191

Grey 11 159

White 12 144

Green 11 182

Blue 14 194

White 14 131

**Sample Output 0**

Color : White

Height : 168.00

Radius : 16.00

Volume : 135045.12

-------------------

Color : Blue

Height : 194.00

Radius : 14.00

Volume : 119395.36

-------------------

Color : Red

Height : 104.00

Radius : 19.00

Volume : 117888.16

-------------------

Color : Green

Height : 191.00

Radius : 13.00

Volume : 101356.06

-------------------

Color : White

Height : 156.00

Radius : 13.00

Volume : 82782.96

-------------------

Color : White

Height : 131.00

Radius : 14.00

Volume : 80622.64

-------------------

Color : Green

Height : 182.00

Radius : 11.00

Volume : 69149.08

-------------------

Color : White

Height : 144.00

Radius : 12.00

Volume : 65111.04

-------------------

Color : Grey

Height : 159.00

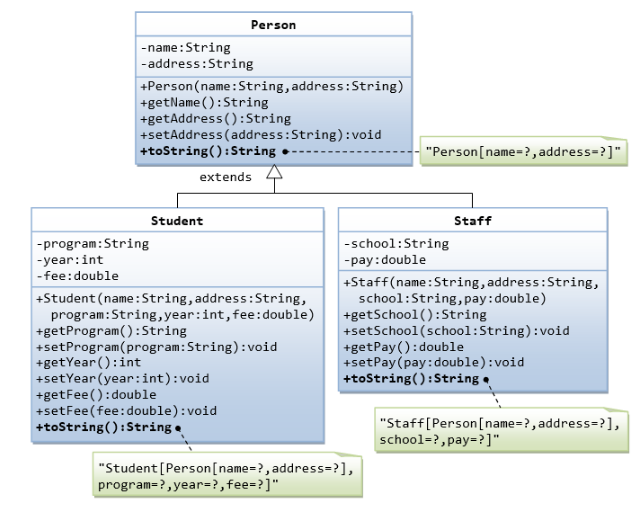
Radius : 11.00

Volume : 60410.46

-------------------

**[Kế thừa - Đa hình]. Bài 48. Person Student Staff**

Cho 3 lớp Person, Student, Staff theo thiết kế dưới đây



Cho danh sách nhân viên và sinh viên của 1 trường đại học, bạn hãy đọc vào danh sách và thực hiện

* In ra danh sách nhân viên sau khi sắp xếp lương giảm dần, nếu có 2 người có cùng lương thì sắp xếp theo tên tăng dần về từ điển.
* In ra danh sách sinh viên theo thứ tự học phí giảm dần, nếu có 2 sinh viên có cùng học phí thì sắp xếp theo tên tăng dần về từ điển.

**Input Format**

* Dòng 1 là N và M : số lượng sinh viên và nhân viên
* Các dòng tiếp theo mô tả sinh viên, mỗi sinh viên gồm 5 dòng : Tên, địa chỉ, ngành học, năm học, học phí
* Các dòng tiêp theo mô tả nhân viên, mỗi nhân viên gồm 4 dòng : Tên, địa chỉ, tên trường, lương

**Constraints**

* 1<=N,M<=2000

**Output Format**

* In ra danh sách nhân viên sau đó in ra danh sách sinh viên theo mẫu, phần học phí và lương in ra với độ chính xác 2 chữ số sau dấu phẩy

**Sample Input 0**

5 9

-----------------

Philip Smith

PennsylvaniaRhode Island

Health Professions

4

131000

-----------------

Rick Williams

Wisconsin

Visual and Performing Arts

3

100000

-----------------

Benjamin Erickson

New Mexico

Engineering

1

110000

-----------------

Claude Anderson

Delaware

Business

1

135000

-----------------

Samuel Matthews

IllinoisIndiana

Health Professions

1

109000

-----------------

Ramon Wheeler

Iowa

Harvard

122000

-----------------

Dan Flores

North Carolina

Harvard

187000

-----------------

Dan Flores

Maryland

Harvard

192000

-----------------

Samuel Matthews

Georgia

Harvard

104000

-----------------

Aidan Simmons

IllinoisIndiana

Stanford

198000

-----------------

Liam Smith

New York

Harvard

178000

-----------------

Charlie Burns

North Dakota

Harvard

194000

-----------------

Conner Martin

Massachusetts

Stanford

139000

-----------------

Clark Green

Connecticut

Harvard

156000

-----------------

**Sample Output 0**

Student List :

-----------------

Full Name : Claude Anderson

Address : Delaware

Program : Business

Year : 1

Fee : 135000 $

-----------------

Full Name : Philip Smith

Address : PennsylvaniaRhode Island

Program : Health Professions

Year : 4

Fee : 131000 $

-----------------

Full Name : Benjamin Erickson

Address : New Mexico

Program : Engineering

Year : 1

Fee : 110000 $

-----------------

Full Name : Samuel Matthews

Address : IllinoisIndiana

Program : Health Professions

Year : 1

Fee : 109000 $

-----------------

Full Name : Rick Williams

Address : Wisconsin

Program : Visual and Performing Arts

Year : 3

Fee : 100000 $

-----------------

Staff List :

-----------------

Full Name : Aidan Simmons

Address : IllinoisIndiana

School : Stanford

Pay : 198000 $

-----------------

Full Name : Charlie Burns

Address : North Dakota

School : Harvard

Pay : 194000 $

-----------------

Full Name : Dan Flores

Address : Maryland

School : Harvard

Pay : 192000 $

-----------------

Full Name : Dan Flores

Address : North Carolina

School : Harvard

Pay : 187000 $

-----------------

Full Name : Liam Smith

Address : New York

School : Harvard

Pay : 178000 $

-----------------

Full Name : Clark Green

Address : Connecticut

School : Harvard

Pay : 156000 $

-----------------

Full Name : Conner Martin

Address : Massachusetts

School : Stanford

Pay : 139000 $

-----------------

Full Name : Ramon Wheeler

Address : Iowa

School : Harvard

Pay : 122000 $

-----------------

Full Name : Samuel Matthews

Address : Georgia

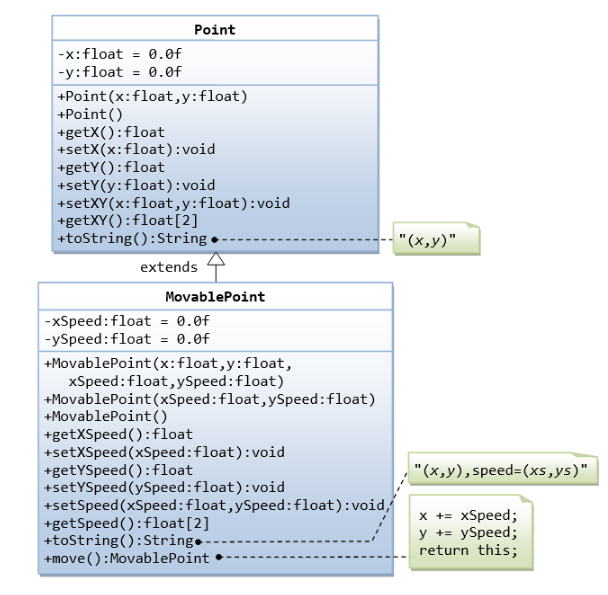
School : Harvard

Pay : 104000 $

-----------------

**[Kế thừa - Đa hình]. Bài 49. Point & Movable Point**

Cho thiết kế lớp Point và Movable Point như sau :



Cho thông tin các điểm của lớp MovablePoint có tọa độ x, y, xSpeed, ySpeed, số lần di chuyển. Hãy cập nhật tọa độ của từng điểm sau 1 số lần di chuyển.

**Input Format**

* Dòng 1 là N : số điểm của lớp MovablePoint
* Mỗi điểm gồm 5 thông số : x, y, xSpeed, ySpeed, số lần di chuyển

**Constraints**

* 1<=N<=1000

**Output Format**

* In ra danh sách các điểm trong lớp MovablePoint sau khi di chuyển, các thông tin lấy 2 chữ số sau dấu phẩy

**Sample Input 0**

6

-18.00 15.00 9.00 1.00 4

19.00 13.00 8.00 7.00 2

16.00 9.00 5.00 7.00 2

-15.00 2.00 6.00 7.00 5

14.00 4.00 10.00 2.00 5

13.00 14.00 10.00 3.00 3

**Sample Output 0**

X : 18.00

Y : 19.00

X Speed : 9.00

Y Speed : 1.00

-------------------

X : 35.00

Y : 27.00

X Speed : 8.00

Y Speed : 7.00

-------------------

X : 26.00

Y : 23.00

X Speed : 5.00

Y Speed : 7.00

-------------------

X : 15.00

Y : 37.00

X Speed : 6.00

Y Speed : 7.00

-------------------

X : 64.00

Y : 14.00

X Speed : 10.00

Y Speed : 2.00

-------------------

X : 43.00

Y : 23.00

X Speed : 10.00

Y Speed : 3.00

-------------------

**[Kế thừa - Đa hình]. Bài 50. Shape abstract class**

Xây dựng 4 lớp : Shape, Circle, Rectangle, Square theo thiết kế dưới đây



Nhập danh sách các đối tượng thuộc lớp Circle, Rectangle, Square sau đó tính toán chu vi và diện tích của từng hình và in ra kết quả trên màn hình. Lấy PI = 3.14

**Input Format**

* Dòng 1 là N : Số lượng hình
* Mỗi hình được mô tả bởi 1 dòng : Chữ cái đầu tiên là C, R, S tương ứng với mô tả hình Circle, Rectangle, Square.
* Nếu là hình tròn thì mô tả tiếp theo sẽ là bán kính, màu sắc, trạng thái được tô màu.
* Nếu là hình chữ nhật mô tả tiếp theo là độ dài 2 cạnh, màu sắc, trạng thái được tô màu.
* Nếu là hình vuông thì mô tả tiếp theo là độ dài cạnh, màu sắc, trạng thái được tô màu.

**Constraints**

* 1<=N<=1000

**Output Format**

* In ra danh sách hình tròn, hình chữ nhật, hình vuông theo thứ tự xuất hiện trong danh sách. Chu vi và bán kính được lấy 2 số sau dấu phẩy.

**Sample Input 0**

6

S 9.00 Blue false

S 7.00 Green false

S 1.00 Yellow false

C 6.00 Blue false

C 5.00 Yellow false

R 6.00 5.00 Yellow false

**Sample Output 0**

Circle :

-----------------------

Radius : 6.00

Color : Blue

Filled : false

Perimeter : 37.68

Area : 113.04

-----------------------

-----------------------

Radius : 5.00

Color : Yellow

Filled : false

Perimeter : 31.40

Area : 78.50

-----------------------

Rectangle :

-----------------------

Width : 5.00

Length : 6.00

Color : Yellow

Filled : false

Perimeter : 22.00

Area : 30.00

-----------------------

Square :

-----------------------

Side : 9.00

Color : Blue

Filled : false

Perimeter : 36.00

Area : 81.00

-----------------------

-----------------------

Side : 7.00

Color : Green

Filled : false

Perimeter : 28.00

Area : 49.00

-----------------------

-----------------------

Side : 1.00

Color : Yellow

Filled : false

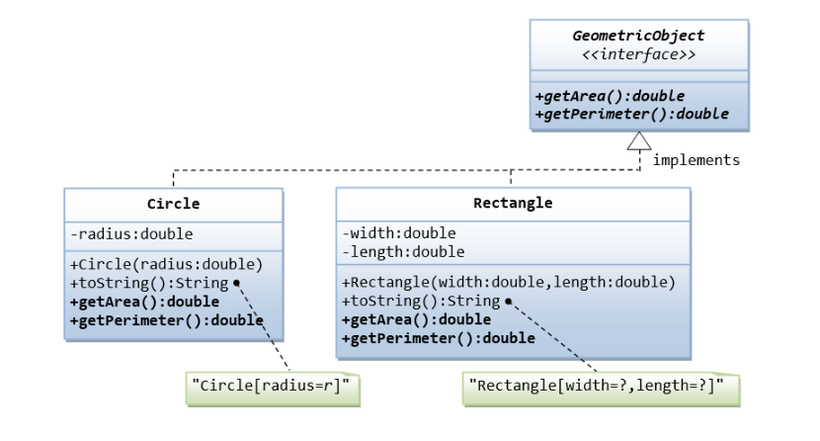
Perimeter : 4.00

Area : 1.00

-----------------------

**[Kế thừa - Đa hình]. Bài 51. GeometricObject Interface**

Thiết kế lớp Circle và Rectangle implement interface geometricObject có chứa 2 phương thức tính diện tích và chu vi như sau :



Nhập vào 1 danh sách hình tròn và HCN và tính toán chu vi, diện tích của các hình đã cho. Diện tích hình tròn tính theo công thức PI \* PI \* r

**Input Format**

* + Dòng 1 là N : Số lượng hình
* Mỗi hình được mô tả bởi 1 dòng : Chữ cái đầu tiên là C, R tương ứng với mô tả hình Circle, Rectangle.
* Nếu là hình tròn thì mô tả tiếp theo sẽ là bán kính
* Nếu là hình chữ nhật mô tả tiếp theo là độ dài 2 cạnh

**Constraints**

* 1<=N<=1000

**Output Format**

* In ra danh sách hình tròn, hình chữ nhật theo thứ tự xuất hiện trong danh sách. Chu vi và bán kính được lấy 2 số sau dấu phẩy.

**Sample Input 0**

8

C 9.00

R 2.00 1.00

R 10.00 9.00

C 5.00

R 2.00 5.00

C 2.00

C 7.00

C 7.00

**Sample Output 0**

Circle :

-----------------------

Radius : 9.00

Perimeter : 56.52

Area : 254.34

-----------------------

-----------------------

Radius : 5.00

Perimeter : 31.40

Area : 78.50

-----------------------

-----------------------

Radius : 2.00

Perimeter : 12.56

Area : 12.56

-----------------------

-----------------------

Radius : 7.00

Perimeter : 43.96

Area : 153.86

-----------------------

-----------------------

Radius : 7.00

Perimeter : 43.96

Area : 153.86

-----------------------

Rectangle :

-----------------------

Width : 1.00

Length : 2.00

Perimeter : 6.00

Area : 2.00

-----------------------

-----------------------

Width : 9.00

Length : 10.00

Perimeter : 38.00

Area : 90.00

-----------------------

-----------------------

Width : 2.00

Length : 5.00

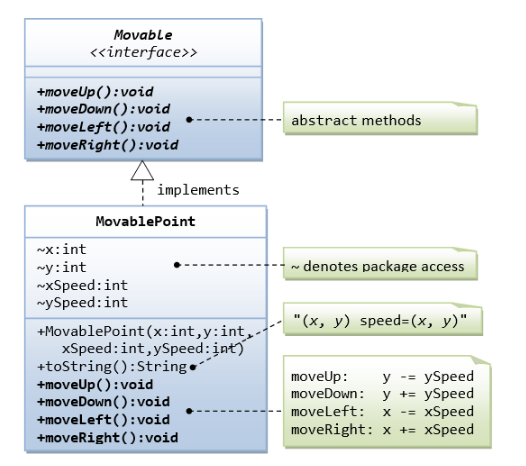
Perimeter : 14.00

Area : 10.00

-----------------------

**[Kế thừa - Đa hình]. Bài 52. Movable Interface**

Cho lớp Movable Point và Movable Interface như sau :



Cho một danh sách các điểm với tọa độ x, y và các lượt di chuyển, hãy xác định vị trí cuối cùng của các điểm ban đầu trong danh sách và in ra màn hình.

**Input Format**

* Dòng 1 là N : số lượng điểm ban đầu
* Mỗi điểm gồm : Dòng 1 là x, y, xSpeed, ySpeed, Dòng 2 là số lần di chuyển M, M dòng tiếp theo mô tả di chuyển

**Constraints**

* 1<=N<=1000
* -1000<=x,y<=1000
* 0<=xSpeed,ySpeed<=100

**Output Format**

* In ra kết quả của bài toán

**Sample Input 0**

8

0 7 9 5

4

Down

Left

Down

Down

8 2 4 7

7

Right

Up

Down

Up

Up

Right

Left

2 6 7 7

6

Left

Down

Left

Up

Up

Up

7 5 0 5

7

Right

Left

Up

Left

Right

Up

Up

7 4 7 7

3

Right

Right

Right

5 9 3 0

4

Down

Down

Up

Up

4 3 1 9

7

Up

Up

Up

Left

Left

Left

Right

5 2 9 1

3

Up

Right

Up

**Sample Output 0**

-9 22

12 -12

-12 -8

7 -10

28 4

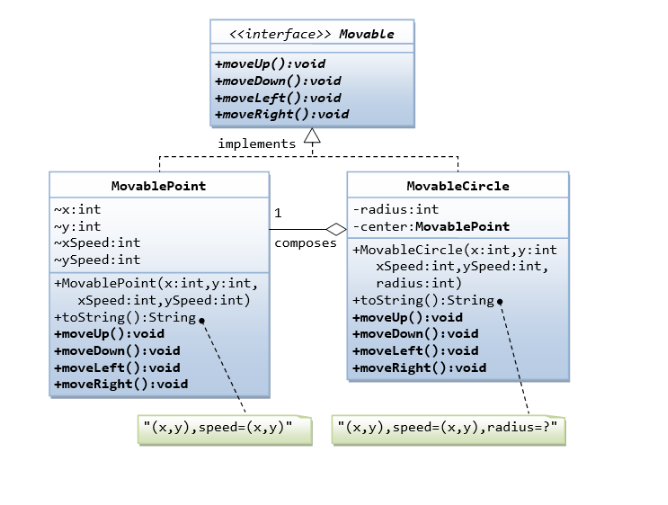
5 9

2 -24

14 0

**[Kế thừa - Đa hình]. Bài 53. Movable Interface**

Cho Interface Movable, lớp MovablePoint, MovableCircle được thiết kế như sau :



Cho danh sách các hình tròn kèm theo bán kính và tọa độ tâm của hình tròn, thực hiện di chuyển tâm hình tròn theo các hướng up, down, left, right sau đó in ra tọa độ tâm hình tròn sau di chuyển.

**Input Format**

* Dòng 1 là N : số lượng hình tròn ban đầu
* Mỗi điểm gồm : Dòng 1 là tọa độ tâm hình tròn, tốc độ di chuyển theo trục x và y, bán kính, Dòng 2 là số lần di chuyển M, M dòng tiếp theo mô tả di chuyển

**Constraints**

* 1<=N<=1000
* -1000<=x,y<=1000
* 0<=xSpeed,ySpeed<=100

**Output Format**

* In ra kết quả của bài toán

**Sample Input 0**

9

6 7 3 5 12

3

Right

Left

Up

4 1 5 7 15

6

Up

Down

Down

Right

Left

Up

1 8 9 4 77

6

Down

Down

Left

Right

Left

Up

8 2 4 8 56

6

Right

Down

Up

Left

Left

Left

2 9 3 8 95

5

Left

Right

Left

Right

Left

2 2 0 4 45

4

Down

Left

Down

Right

1 0 8 3 23

7

Up

Left

Down

Right

Left

Left

Left

6 4 9 2 25

7

Up

Down

Down

Right

Down

Down

Left

9 7 5 0 19

7

Up

Right

Left

Down

Left

Right

Down

**Sample Output 0**

6 2

4 1

-8 12

0 2

-1 9

2 10

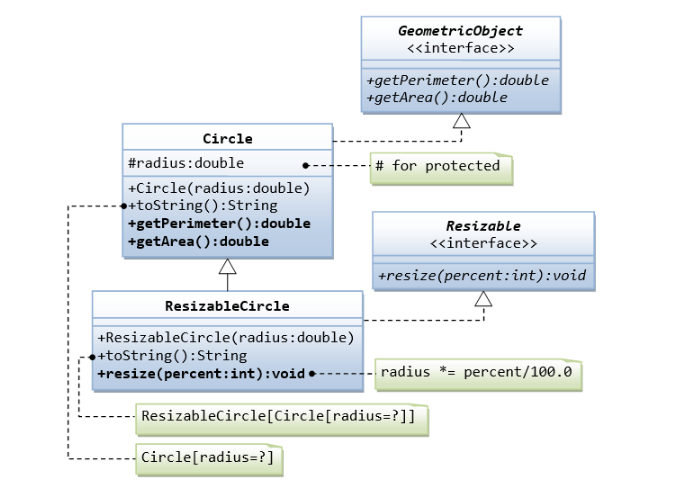
-23 0

6 10

9 7

## [Kế thừa - Đa hình]. Bài 54. Resizable Interface

Cho các lớp và interface theo thiết kế dưới đây :



Thực hiện thay đổi bán kính của các hình tròn đã cho và in ra bán kính, chu vi, diện tích sau các lần thay đổi đó. Lấy PI = 3.14

**Input Format**

* Dòng 1 là N : số lượng hình tròn
* Thông tin về mỗi hình tròn bao gồm : Dòng 1 là bán kính và M - số lần thay đổi bán kính, dòng tiếp theo là M số tương ứng với tỉ lệ (phần trăm) thay đổi của bán kính.

**Constraints**

* 1<=N, M<=1000

**Output Format**

* In ra thông số bán kính, chu vi, diện tích sau các lần thay đổi bán kính, kết quả lấy 2 số sau dấu phẩy

**Sample Input 0**

5

12.00 3

184 155 2

9.00 3

198 182 157

5.00 3

156 140 161

12.00 3

64 104 163

9.00 3

200 115 71

**Sample Output 0**

---------------------

Radius : 0.68

Perimeter : 4.30

Area : 1.47

---------------------

---------------------

Radius : 50.92

Perimeter : 319.77

Area : 8141.18

---------------------

---------------------

Radius : 17.58

Perimeter : 110.41

Area : 970.57

---------------------

---------------------

Radius : 13.02

Perimeter : 81.76

Area : 532.22

---------------------

---------------------

Radius : 14.70

Perimeter : 92.30

Area : 678.25

---------------------