**Bài 1: Kế thừa lớp Person**

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.

Mã nguồn tham khảo :

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();

}

@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) {

Scanner sc = new Scanner(System.in);

int n = sc.nextInt();

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

sc.nextLine();

String ten = sc.nextLine();

String s = sc.nextLine();

String lop = sc.nextLine();

double diem = sc.nextDouble();

String ngaySinh = "", diaChi = "";

for(char x : s.toCharArray()){

if(Character.isDigit(x) || x == '/'){

ngaySinh += x;

}

else diaChi += x;

}

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

sinhVien.chuanHoa();

System.out.println(sinhVien);

}

}

}

**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.
* Cần tách ngày sinh và địa chỉ của sinh viên.

**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**

7

Nguyen AnH MaNH

27/8/2004Nam Dinh

CNTT2

2.70

pham Phuong TuaN

21/9/2004Nam Dinh

CNTT1

2.70

Vu Ngoc NAM

7/9/2004Nam Dinh

CNTT2

3.05

Vu AnH LoNG

17/12/2004Ha Noi

CNTT1

2.80

Nguyen Phuong NAM

23/6/2004Ha Nam

HTTT3

2.80

Luong AnH NAM

16/1/2004Thai Binh

DTVT1

2.80

Nguyen Phuong HaI

14/1/2004Nam Dinh

DTVT1

3.05

**Sample Output 0**

0001 Nguyen Anh Manh 27/08/2004 Nam Dinh CNTT2 2.70

0002 Pham Phuong Tuan 21/09/2004 Nam Dinh CNTT1 2.70

0003 Vu Ngoc Nam 07/09/2004 Nam Dinh CNTT2 3.05

0004 Vu Anh Long 17/12/2004 Ha Noi CNTT1 2.80

0005 Nguyen Phuong Nam 23/06/2004 Ha Nam HTTT3 2.80

0006 Luong Anh Nam 16/01/2004 Thai Binh DTVT1 2.80

0007 Nguyen Phuong Hai 14/01/2004 Nam Dinh DTVT1 3.05

**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

**Bài 3. 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

**Bài 4. 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

**Bài 5. 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

**Bài 6. 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

**Bài 7. 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

**Bài 8. 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

**Bài 9. 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 :

**Bài 10. 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**

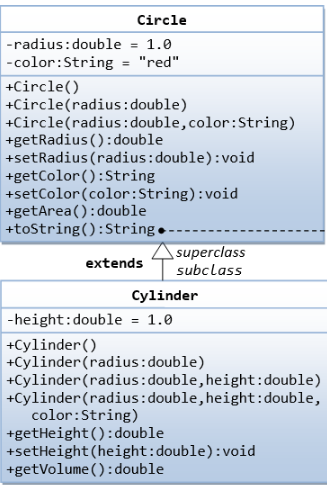
DANH SACH OTO :

OTO4 TOY555 TOYOTA Vang 150 2400

DANH SACH XE MAY :

**Bài 11. 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

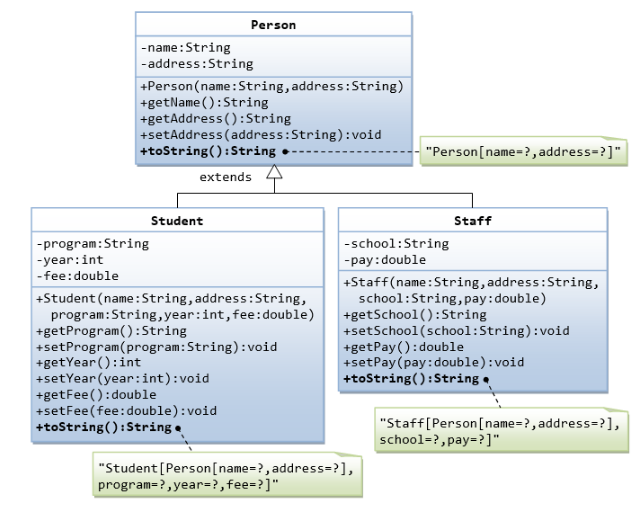
Radius : 11.00

Volume : 60410.46

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

**Bài 12. 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

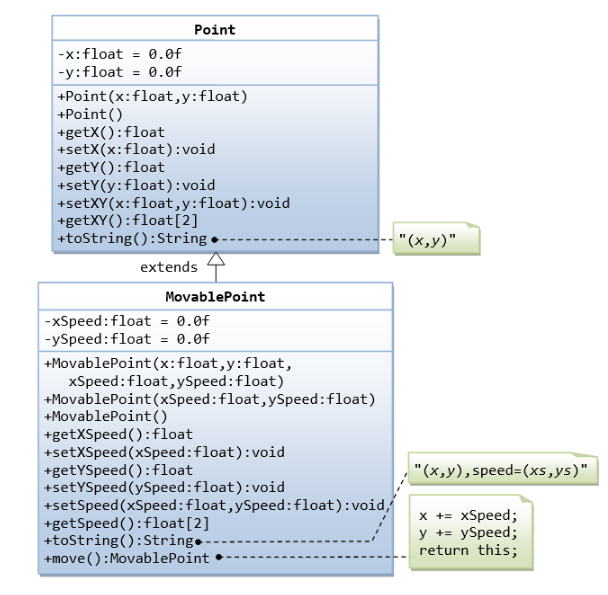
School : Harvard

Pay : 104000 $

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

**Bài 13. 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

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