O professor Girafales organizou a confecção de um uniforme para as turmas da escola para comemorar o final do ano. Após algumas conversas, ficou decidido com os alunos que eles poderiam escolher a cor do uniforme entre branco ou vermelho. Assim sendo, Girafales precisa de sua ajuda para organizar as listas de quem quer o uniforme em cada uma das turmas, relacionando estas camisetas pela cor, tamanho (P, M ou G) e por último pelo nome.

**Entrada**

Cada caso de teste inicia com um valor **N**, (1 ≤ **N** ≤ 60) inteiro e positivo, que indica a quantidade de uniformes a serem feitas para aquela turma. As próximas **N**\*2 linhas contém informações de cada um dos uniformes (serão duas linhas de informação para cada uniforme). A primeira linha irá conter o nome do estudante e a segunda linha irá conter a cor do uniforme ("branco" ou "vermelho") seguido por um espaço e pelo tamanho do uniforme "P" "M" ou "G". A entrada termina quando o valor de **N** for igual a zero (0) e esta valor não deverá ser processado.

**Saída**

Para cada caso de entrada deverão ser impressas as informações ordenadas pela cor em ordem ascendente, seguido pelos tamanhos em ordem descendente e por último por ordem ascendente de nome, conforme o exemplo abaixo.

| Exemplo de Entrada | Exemplo de Saída |
| --- | --- |
| 9 Maria Jose branco P Mangojata Mancuda vermelho P Cezar Torres Mo branco P Baka Lhau vermelho P JuJu Mentina branco M Amaro Dinha vermelho P Adabi Finho branco G Severina Rigudinha branco G Carlos Chade Losna vermelho P 0 | branco P Cezar Torres Mo branco P Maria Jose branco M JuJu Mentina branco G Adabi Finho branco G Severina Rigudinha vermelho P Amaro Dinha vermelho P Baka Lhau vermelho P Carlos Chade Losna vermelho P Mangojata Mancuda |

***Solução***

import java.io.IOException;

import java.util.\*;

public class YearEndUniform

{

private String name;

private String uniformSize;

private String uniformColor;

public String getUniformColor()

{

return this.uniformColor;

}

public void setUniformColor(String uniformColor)

{

this.uniformColor = uniformColor;

}

public String getName()

{

return this.name;

}

public void setName(String name)

{

this.name = name;

}

public String getUniformSize()

{

return this.uniformSize;

}

public void setUniformSize(String uniformSize)

{

this.uniformSize = uniformSize;

}

public YearEndUniform(String name, String uniformColor, String uniformSize)

{

this.name = name;

this.uniformColor = uniformColor;

this.uniformSize = uniformSize;

}

public void printStudentInfos()

{

System.out.println(this.uniformColor + " " + uniformSize + " " + name);

}

public static void main(String[] args) throws IOException

{

Scanner scanner = new Scanner(System.in);

int N = scanner.nextInt();

scanner.nextLine();

ArrayList<YearEndUniform> arrayUniforms= new ArrayList<>(N);

String name, uniformDetails, uniformColor, uniformSize;

for(int i = 0; i < N; i++)

{

name = scanner.nextLine();

uniformDetails = scanner.nextLine();

uniformColor = uniformDetails.split(" ")[0];

uniformSize= uniformDetails.split(" ")[1];

arrayUniforms.add(new YearEndUniform(name, uniformColor, uniformSize));

}

scanner.close();

Collections.sort(arrayUniforms, new UniformComparator());

for(var item : arrayUniforms)

{

item.printStudentInfos();

}

}

static class UniformComparator implements Comparator<YearEndUniform>

{

@Override

public int compare(YearEndUniform student0, YearEndUniform student1)

{

int res = student0.uniformColor.compareTo(student1.uniformColor);

if(res == 0)

{

res = student0.uniformSize.compareTo(student1.uniformSize);

if(res == 0)

{

res = student0.name.compareTo(student1.name);

} else {

res = -res;

}

}

return res;

}

}

}