1081 检查密码 (15分)

1081 检查密码 (15分)

本题要求你帮助某网站的用户注册模块写一个密码合法性检查的小功能。该网站要求用户设置的密码必须由不少于6个字符组成，并且只能有英文字母、数字和小数点 `.`，还必须既有字母也有数字。

输入格式:

输入第一行给出一个正整数 N（≤ 100），随后 N 行，每行给出一个用户设置的密码，为不超过 80 个字符的非空字符串，以回车结束。

输出格式:

对每个用户的密码，在一行中输出系统反馈信息，分以下5种：

- 如果密码合法，输出 `Your password is wan mei.`；
- 如果密码太短，不论合法与否，都输出 `Your password is tai duan le.`；
- 如果密码长度合法，但存在不合法字符，则输出 `Your password is tai luan le.`；
- 如果密码长度合法，但只有字母没有数字，则输出 `Your password needs shu zi.`；
- 如果密码长度合法，但只有数字没有字母，则输出 `Your password needs zi mu.`。

输入样例:

```
5
123s
zheshi.wodepw
1234.5678
WanMei23333
pass*word.6
```

输出样例:

```
Your password is tai duan le.
Your password needs shu zi.
Your password needs zi mu.
Your password is wan mei.
Your password is tai luan le.
```

```cpp
#include <iostream>

using namespace std;

int main()
{
    int n;
    cin >> n;

    while(n --)
    {
        int flag = 1;
```

```
        int flag_n = 0;
        int flag_m = 0;
        string s;
        cin >> s;
        if(s.size() < 6) cout << "Your password is tai duan le."<<endl;
        else
        {
            for(auto c : s)
            {
                if((c >= 'A' && c <= 'Z') || (c >= 'a' && c <= 'z')) flag_m = 1;
                else if(c >= '0' && c <= '9') flag_n = 1;
                else if(c == '.') ;
                else flag = 0;
            }
        if(!flag) cout << "Your password is tai luan le." << endl;
        if(flag && ! flag_n && flag_m) cout << "Your password needs shu zi."
<<endl;
        if(flag && flag_n && ! flag_m) cout << "Your password needs zi mu."
<<endl;
        if(flag && flag_m && flag_n) cout << "Your password is wan mei." <<endl;
        }

    }
}
```

1082 射击比赛 (20分)

1082 射击比赛 (20分)

本题目给出的射击比赛的规则非常简单，谁打的弹洞距离靶心最近，谁就是冠军；谁差得最远，谁就是菜鸟。本题给出一系列弹洞的平面坐标(x,y)，请你编写程序找出冠军和菜鸟。我们假设靶心在原点(0,0)。

输入格式:

输入在第一行中给出一个正整数 N（≤ 10 000）。随后 N 行，每行按下列格式给出：

```
ID x y
```

其中 `ID` 是运动员的编号（由 4 位数字组成）；`x` 和 `y` 是其打出的弹洞的平面坐标(`x`,`y`)，均为整数，且 0 ≤ |`x`|, |`y`| ≤ 100。题目保证每个运动员的编号不重复，且每人只打 1 枪。

输出格式:

输出冠军和菜鸟的编号，中间空 1 格。题目保证他们是唯一的。

输入样例:

```
3
0001 5 7
1020 -1 3
0233 0 -1
```

输出样例:

```
0233 0001
```

```cpp
#include <iostream>

using namespace std;

int main() {
    int n, id, x, y,maxid,minid;
    int maxdis = -1,mindis = 99999;
    cin >> n;
    while(n --)
    {
        cin >> id >> x >> y;
        int dis = x * x + y * y;
        if (dis > maxdis) maxid = id;
        if (dis < mindis) minid = id;
        maxdis = max(maxdis, dis);
        mindis = min(mindis, dis);
    }
    printf("%04d %04d", minid, maxid);
    return 0;
}
```

1083 是否存在相等的差 (20分)

返回

1083 是否存在相等的差 (20分)

给定 N 张卡片，正面分别写上 1、2、……、N，然后全部翻面，洗牌，在背面分别写上 1、2、……、N。将每张牌的正反两面数字相减（大减小），得到 N 个非负差值，其中是否存在相等的差？

输入格式：

输入第一行给出一个正整数 N（$2 \le N \le 10\ 000$），随后一行给出 1 到 N 的一个洗牌后的排列，第 i 个数表示正面写了 i 的那张卡片背面的数字。

输出格式：

按照“差值 重复次数”的格式从大到小输出重复的差值及其重复的次数，每行输出一个结果。

输入样例：

```
8
3 5 8 6 2 1 4 7
```

输出样例：

```
5 2
3 3
2 2
```

```cpp
#include <iostream>
using namespace std;
int main() {
    int n, t, a[10000] = {0};
    cin >> n;
    for (int i = 1; i <= n; i++) {
        cin >> t;
        a[abs(t-i)]++;
    }
    for (int i = 9999; i >= 0; i--)
        if (a[i] >= 2) cout << i << " " << a[i] << endl;
    return 0;
}
```

1084 外观数列 (20分)

1084 外观数列 (20分)

外观数列是指具有以下特点的整数序列：

```
d, d1, d111, d113, d11231, d112213111, ...
```

它从不等于 1 的数字 `d` 开始，序列的第 n+1 项是对第 n 项的描述。比如第 2 项表示第 1 项有 1 个 `d`，所以就是 `d1`；第 2 项是 1 个 `d`（对应 `d1`）和 1 个 1（对应 11），所以第 3 项就是 `d111`。又比如第 4 项是 `d113`，其描述就是 1 个 `d`，2 个 1，1 个 3，所以下一项就是 `d11231`。当然这个定义对 `d` = 1 也成立。本题要求你推算任意给定数字 `d` 的外观数列的第 N 项。

输入格式：

输入第一行给出 [0,9] 范围内的一个整数 `d`、以及一个正整数 N（≤ 40），用空格分隔。

输出格式：

在一行中给出数字 `d` 的外观数列的第 N 项。

输入样例：

```
1 8
```

输出样例：

```
1123123111
```

```cpp
#include <iostream>
using namespace std;
int main() {
    string s;
    int n, j;
    cin >> s >> n;
    for (int cnt = 1; cnt < n; cnt++) {
        string t;
        for (int i = 0; i < s.length(); i = j) {
            for (j = i; j < s.length() && s[j] == s[i]; j++);
            t += s[i] + to_string(j - i);
        }
        s = t;
    }
    cout << s;
    return 0;
}
```

1085 PAT单位排行 (25分)

1085 PAT单位排行 (25分)

每次 PAT 考试结束后，考试中心都会发布一个考生单位排行榜。本题就请你实现这个功能。

输入格式:

输入第一行给出一个正整数 N（$\leq 10^5$），即考生人数。随后 N 行，每行按下列格式给出一个考生的信息：

```
准考证号 得分 学校
```

其中 `准考证号` 是由 6 个字符组成的字符串，其首字母表示考试的级别：`B` 代表乙级，`A` 代表甲级，`T` 代表顶级；`得分` 是 [0, 100] 区间内的整数；`学校` 是由不超过 6 个英文字母组成的单位码（大小写无关）。注意：题目保证每个考生的准考证号是不同的。

输出格式:

首先在一行中输出单位个数。随后按以下格式非降序输出单位的排行榜：

```
排名 学校 加权总分 考生人数
```

其中 `排名` 是该单位的排名（从 1 开始）；`学校` 是全部按小写字母输出的单位码；`加权总分` 定义为 `乙级总分/1.5 + 甲级总分 + 顶级总分*1.5` 的**整数部分**；`考生人数` 是该属于单位的考生的总人数。

学校首先按加权总分排行。如有并列，则应对应相同的排名，并按考生人数升序输出。如果仍然并列，则按单位码的字典序输出。

输入样例:

```
10
A57908 85 Au
B57908 54 LanX
A37487 60 au
T28374 67 CMU
T32486 24 hypu
A66734 92 cmu
B76378 71 AU
A47780 45 lanx
A72809 100 pku
A03274 45 hypu
```

输出样例:

```
5
1 cmu 192 2
1 au 192 3
3 pku 100 1
4 hypu 81 2
4 lanx 81 2
```

```
#include <iostream>
#include <algorithm>
#include <cctype>
#include <vector>
```

```cpp
#include <unordered_map>
using namespace std;
struct node {
    string school;
    int tws, ns;
};
bool cmp(node a, node b) {
    if (a.tws != b.tws)
        return a.tws > b.tws;
    else if (a.ns != b.ns)
        return a.ns < b.ns;
    else
        return a.school < b.school;
}
int main() {
    int n;
    scanf("%d", &n);
    unordered_map<string, int> cnt;
    unordered_map<string, double> sum;
    for (int i = 0; i < n; i++) {
        string id, school;
        cin >> id;
        double score;
        scanf("%lf", &score);
        cin >> school;
        for (int j = 0; j < school.length(); j++)
            school[j] = tolower(school[j]);
        if (id[0] == 'B')
            score = score / 1.5;
        else if (id[0] == 'T')
            score = score * 1.5;
        sum[school] += score;
        cnt[school]++;
    }
    vector<node> ans;
    for (auto it = cnt.begin(); it != cnt.end(); it++)
        ans.push_back(node{it->first, (int)sum[it->first], cnt[it->first]});
    sort(ans.begin(), ans.end(), cmp);
    int rank = 0, pres = -1;
    printf("%d\n", (int)ans.size());
    for (int i = 0; i < ans.size(); i++) {
        if (pres != ans[i].tws) rank = i + 1;
        pres = ans[i].tws;
        printf("%d ", rank);
        cout << ans[i].school;
        printf(" %d %d\n", ans[i].tws, ans[i].ns);
    }
    return 0;
}
```

1086 就不告诉你 (15分)

1086 就不告诉你 (15分)

做作业的时候，邻座的小盆友问你：“五乘以七等于多少？”你应该不失礼貌地围笑着告诉他：“五十三。”本题就要求你，对任何一对给定的正整数，倒着输出它们的乘积。

输入格式:

输入在第一行给出两个不超过 1000 的正整数 A 和 B，其间以空格分隔。

输出格式:

在一行中倒着输出 A 和 B 的乘积。

输入样例:

```
5 7
```

输出样例:

```
53
```

```cpp
#include <iostream>
#include <string>
#include <algorithm>

using namespace std;

int main()
{
    int n,m;
    cin >> n >> m;
    int res = n * m;
    string ans = to_string(res);
    reverse(ans.begin(), ans.end());
    printf("%d", stoi(ans));
}
```

1087 有多少不同的值 (20分)

1087 有多少不同的值 (20分)

当自然数 n 依次取 1、2、3、.....、N 时，算式 $\lfloor n/2 \rfloor + \lfloor n/3 \rfloor + \lfloor n/5 \rfloor$ 有多少个不同的值?
(注：$\lfloor x \rfloor$ 为取整函数，表示不超过 x 的最大自然数，即 x 的整数部分。)

输入格式:

输入给出一个正整数 N $(2 \leq N \leq 10^4)$ 。

输出格式:

在一行中输出题面中算式取到的不同值的个数。

输入样例:

```
2017
```

输出样例:

```
1480
```

```
#include <iostream>
#include <set>
using namespace std;
int main() {
    int n;
    scanf("%d", &n);
    set<int> s;
    for (int i = 1; i <= n; i++)
        s.insert(i / 2 + i / 3 + i / 5);
    printf("%d", s.size());
    return 0;
}
```

1088 三人行 (20分)

1088 三人行 (20分)

子曰：“三人行，必有我师焉。择其善者而从之，其不善者而改之。”

本题给定甲、乙、丙三个人的能力值关系为：甲的能力值确定是 2 位正整数；把甲的能力值的 2 个数字调换位置就是乙的能力值；甲乙两人能力差是丙的能力值的 X 倍；乙的能力值是丙的 Y 倍。请你指出谁比你强应“从之”，谁比你弱应“改之”。

输入格式:

输入在一行中给出三个数，依次为：M（你自己的能力值）、X 和 Y。三个数字均为不超过 1000 的正整数。

输出格式:

在一行中首先输出甲的能力值，随后依次输出甲、乙、丙三人与你的关系：如果其比你强，输出 `Cong`；平等则输出 `Ping`；比你弱则输出 `Gai`。其间以 1 个空格分隔，行首尾不得有多余空格。

注意：如果解不唯一，则以甲的最大解为准进行判断；如果解不存在，则输出 `No Solution`。

输入样例 1:

```
48 3 7
```

输出样例 1:

```
48 Ping Cong Gai
```

```cpp
#include <iostream>
#include <cmath>
using namespace std;
int m, x, y;
void print(double t) {
    if (m == t) printf(" Ping");
    else if (m < t) printf(" Cong");
    else printf(" Gai");
}
int main() {
    scanf("%d %d %d", &m, &x, &y);
    for (int i = 99; i >= 10; i--) {
        int j = i % 10 * 10 + i / 10;
        double k = abs(j - i) * 1.0 / x;
        if (j == k * y) {
            cout << i;
            print(i); print(j); print(k);
            return 0;
        }
    }
    cout << "No Solution";
    return 0;
}
```

1089 狼人杀-简单版 (20分)

1089 狼人杀-简单版 (20分)

以下文字摘自《灵机一动·好玩的数学》：“狼人杀”游戏分为狼人、好人两大阵营。在一局“狼人杀”游戏中，1 号玩家说：“2 号是狼人”，2 号玩家说：“3 号是好人”，3 号玩家说：“4 号是狼人”，4 号玩家说：“5 号是好人”，5 号玩家说：“4 号是好人”。已知这 5 名玩家中有 2 人扮演狼人角色，有 2 人说的不是实话，有狼人撒谎但并不是所有狼人都在撒谎。扮演狼人角色的是哪两号玩家？

本题是这个问题的升级版：已知 N 名玩家中有 2 人扮演狼人角色，有 2 人说的不是实话，有狼人撒谎但并不是所有狼人都在撒谎。要求你找出扮演狼人角色的是哪几号玩家？

输入格式:

输入在第一行中给出一个正整数 N ($5 \le N \le 100$)。随后 N 行，第 i 行给出第 i 号玩家说的话（$1 \le i \le N$），即一个玩家编号，用正号表示好人，负号表示狼人。

输出格式:

如果有解，在一行中按递增顺序输出 2 个狼人的编号，其间以空格分隔，行首尾不得有多余空格。如果解不唯一，则输出最小序列解 —— 即对于两个序列 $A = a[1], ..., a[M]$ 和 $B = b[1], ..., b[M]$，若存在 $0 \le k < M$ 使得 $a[i] = b[i]$ $(i \le k)$，且 $a[k+1] < b[k+1]$，则称序列 A 小于序列 B。若无解则输出 `No Solution`。

输入样例 1:

```
5
-2
+3
-4
+5
+4
```

输出样例 1:

```
1 4
```

输入样例 2:

```
6
+6
+3
+1
-5
-2
+4
```

输出样例 2（解不唯一）:

```
1 5
```

```cpp
#include <iostream>
#include <vector>
#include <cmath>
using namespace std;
int main() {
    int n;
    cin >> n;
    vector<int> v(n+1);
    for (int i = 1; i <= n; i++) cin >> v[i];
    for (int i = 1; i <= n; i++) {
        for (int j = i + 1; j <= n; j++) {
            vector<int> lie, a(n + 1, 1);
            a[i] = a[j] = -1;
            for (int k = 1; k <= n; k++)
                if (v[k] * a[abs(v[k])] < 0) lie.push_back(k);
            if (lie.size() == 2 && a[lie[0]] + a[lie[1]] == 0) {
                cout << i << " " << j;
                return 0;
            }
        }
    }
    cout << "No Solution";
    return 0;
}
```

1090 危险品装箱 (25分)

1090 危险品装箱 (25分)

集装箱运输货物时，我们必须特别小心，不能把不相容的货物装在一只箱子里。比如氧化剂绝对不能跟易燃液体同箱，否则很容易造成爆炸。

本题给定一张不相容物品的清单，需要你检查每一张集装箱货品清单，判断它们是否能装在同一只箱子里。

输入格式:

输入第一行给出两个正整数：N ($\le 10^4$) 是成对的不相容物品的对数；M (≤ 100) 是集装箱货品清单的单数。

随后数据分两大块给出。第一块有 N 行，每行给出一对不相容的物品。第二块有 M 行，每行给出一箱货物的清单，格式如下：

```
K G[1] G[2] ... G[K]
```

其中 `K` (≤ 1000) 是物品件数，`G[i]` 是物品的编号。简单起见，每件物品用一个 5 位数的编号代表。两个数字之间用空格分隔。

输出格式:

对每箱货物清单，判断是否可以安全运输。如果没有不相容物品，则在一行中输出 `Yes`，否则输出 `No`。

输入样例：

```
6 3
20001 20002
20003 20004
20005 20006
20003 20001
20005 20004
20004 20006
4 00001 20004 00002 20003
5 98823 20002 20003 20006 10010
3 12345 67890 23333
```

输出样例：

```
No
Yes
Yes
```

```
#include <iostream>
#include <vector>
#include <map>
using namespace std;
int main() {
    int n, k, t1, t2;
    map<int,vector<int>> m;
    scanf("%d%d", &n, &k);
    for (int i = 0; i < n; i++) {
        scanf("%d%d", &t1, &t2);
        m[t1].push_back(t2);
        m[t2].push_back(t1);
    }
    while (k--) {
        int cnt, flag = 0, a[100000] = {0};
        scanf("%d", &cnt);
        vector<int> v(cnt);
        for (int i = 0; i < cnt; i++) {
            scanf("%d", &v[i]);
            a[v[i]] = 1;
        }
        for (int i = 0; i < v.size(); i++)
            for (int j = 0; j < m[v[i]].size(); j++)
                if (a[m[v[i]][j]] == 1) flag = 1;
        printf("%s\n",flag ? "No" :"Yes");
    }
    return 0;
}
```

1091 N-自守数 (15分)

1091 N-自守数 (15分)

如果某个数 K 的平方乘以 N 以后，结果的末尾几位数等于 K，那么就称这个数为“N-自守数”。例如 $3 \times 92^2 = 25392$，而 25392 的末尾两位正好是 92，所以 92 是一个 3-自守数。

本题就请你编写程序判断一个给定的数字是否关于某个 N 是 N-自守数。

输入格式:

输入在第一行中给出正整数 M（≤ 20），随后一行给出 M 个待检测的、不超过 1000 的正整数。

输出格式:

对每个需要检测的数字，如果它是 N-自守数就在一行中输出最小的 N 和 NK^2 的值，以一个空格隔开；否则输出 `No`。注意题目保证 $N < 10$。

输入样例:

```
3
92 5 233
```

输出样例:

```
3 25392
1 25
No
```

```
#include <iostream>
#include <string>
using namespace std;
int main() {
    int m;
    cin >> m;
    while (m--) {
        int k, flag = 0;
        cin >> k;
        for (int n = 1; n < 10; n++) {
            int mul = n * k * k;
            string smul = to_string(mul), sk = to_string(k);
            string smulend = smul.substr(smul.length() - sk.length());
            if (smulend == sk) {
                printf("%d %d\n", n, mul);
                flag = 1;
                break;
            }
        }
        if (flag == 0) printf("No\n");
    }
    return 0;
}
```

1092 最好吃的月饼 (20分)

输入格式：

输入首先给出两个正整数 N（≤ 1000）和 M（≤ 100），分别为月饼的种类数（于是默认月饼种类从 1 到 N 编号）和参与统计的城市数量。

接下来 M 行，每行给出 N 个非负整数（均不超过 1 百万），其中第 i 个整数为第 i 种月饼的销量（块）。数字间以空格分隔。

输出格式：

在第一行中输出最大销量，第二行输出销量最大的月饼的种类编号。如果冠军不唯一，则按编号递增顺序输出并列冠军。数字间以 1 个空格分隔，行首尾不得有多余空格。

输入样例：

```
5 3
1001 992 0 233 6
8 0 2018 0 2008
36 18 0 1024 4
```

输出样例：

```
2018
3 5
```

```cpp
#include <iostream>
#include <vector>
using namespace std;
int a[1005][105], sum[1005];
int main() {
    int m, n, maxn = 0, total = 0;
    vector<int> ans;
    cin >> m >> n;
    for (int i = 1; i <= n; i++) {
        for (int j = 1; j <= m; j++) {
            cin >> a[i][j];
            sum[j] += a[i][j];
            maxn = max(maxn, sum[j]);
        }
    }
    cout << maxn << endl;
    for (int i = 1; i <= m; i++)
        if (sum[i] == maxn) ans.push_back(i);
    for (int i = 0; i < ans.size(); i++) {
        if (i != 0) cout << " ";
        cout << ans[i];
    }
    return 0;
}
```

1093 字符串A+B (20分)

1093 字符串A+B (20分)

给定两个字符串 A 和 B，本题要求你输出 $A+B$，即两个字符串的并集。要求先输出 A，再输出 B，但**重复的字符必须被剔除。**

输入格式:

输入在两行中分别给出 A 和 B，均为长度不超过 10^6的、由可见 ASCII 字符 (即码值为32~126)和空格组成的、由回车标识结束的非空字符串。

输出格式:

在一行中输出题面要求的 A 和 B 的和。

输入样例:

```
This is a sample test
to show you_How it works
```

输出样例:

```
This ampletowyu_Hrk
```

```cpp
#include <iostream>
using namespace std;
int main() {
    string s1, s2, s;
    int hash[200] = {0};
    getline(cin, s1);
    getline(cin, s2);
    s = s1 + s2;
    for (int i = 0; i < s.size(); i++) {
        if (hash[s[i]] == 0) cout << s[i];
        hash[s[i]] = 1;
    }
    return 0;
}
```

1094 谷歌的招聘 (20分)

自然常数 e 是一个著名的超越数，前面若干位写出来是这样的：e = 2.718281828459045235360287471352662497757247093699959574966967627724076630353547

其中粗体标出的 10 位数就是答案。

本题要求你编程解决一个更通用的问题：从任一给定的长度为 L 的数字中，找出最早出现的 K 位连续数字所组成的素数。

输入格式:

输入在第一行给出 2 个正整数，分别是 L（不超过 1000 的正整数，为数字长度）和 K（小于 10 的正整数）。接下来一行给出一个长度为 L 的正整数 N。

输出格式:

在一行中输出 N 中最早出现的 K 位连续数字所组成的素数。如果这样的素数不存在，则输出 `404`。注意，原始数字中的前导零也计算在位数之内。例如在 200236 中找 4 位素数，0023 算是解；但第一位 2 不能被当成 0002 输出，因为在原始数字中不存在这个 2 的前导零。

输入样例 1:

```
20 5
23654987725541023819
```

输出样例 1:

```
49877
```

输入样例 2:

```
10 3
2468024680
```

```
#include <iostream>
#include <string>
using namespace std;
bool isPrime(int n) {
    if (n == 0 || n == 1) return false;
    for (int i = 2; i * i <= n; i++)
        if (n % i == 0) return false;
    return true;
}
int main() {
    int l, k;
    string s;
    cin >> l >> k >> s;
    for (int i = 0; i <= l - k; i++) {
        string t = s.substr(i, k);
        int num = stoi(t);
        if (isPrime(num)) {
            cout << t;
            return 0;
        }
    }
```

```
    cout << "404\n";
    return 0;
}
```

1095 解码PAT准考证 (25分)

1095 解码PAT准考证 (25分)

PAT 准考证号由 4 部分组成：

- 第 1 位是级别，即 `T` 代表顶级；`A` 代表甲级；`B` 代表乙级；
- 第 2~4 位是考场编号，范围从 101 到 999；
- 第 5~10 位是考试日期，格式为年、月、日顺次各占 2 位；
- 最后 11~13 位是考生编号，范围从 000 到 999。

现给定一系列考生的准考证号和他们的成绩，请你按照要求输出各种统计信息。

输入格式：

输入首先在一行中给出两个正整数 N（$\leq 10^4$）和 M（≤ 100），分别为考生人数和统计要求的个数。

接下来 N 行，每行给出一个考生的准考证号和其分数（在区间 $[0, 100]$ 内的整数），其间以空格分隔。

考生信息之后，再给出 M 行，每行给出一个统计要求，格式为：`类型 指令`，其中

- `类型` 为 1 表示要求按分数非升序输出某个指定级别的考生的成绩，对应的 `指令` 则给出代表指定级别的字母；
- `类型` 为 2 表示要求将某指定考场的考生人数和总分统计输出，对应的 `指令` 则给出指定考场的编号；
- `类型` 为 3 表示要求将某指定日期的考生人数分考场统计输出，对应的 `指令` 则给出指定日期，格式与准考证上日期相同。

输出格式：

对每项统计要求，首先在一行中输出 `Case #: 要求`，其中 `#` 是该项要求的编号，从 1 开始；`要求` 即复制输入给出的要求。随后输出相应的统计结果：

- `类型` 为 1 的指令，输出格式与输入的考生信息格式相同，即 `准考证号 成绩`。对于分数并列的考生，按其准考证号的字典序递增输出（题目保证无重复准考证号）；
- `类型` 为 2 的指令，按 `人数 总分` 的格式输出；
- `类型` 为 3 的指令，输出按人数非递增顺序，格式为 `考场编号 总人数`。若人数并列则按考场编号递增顺序输出。

如果查询结果为空，则输出 `NA`。

输入样例：

```
8 4
B123180908127 99
B102180908003 86
A112180318002 98
T107150310127 62
A107180908108 100
T123180908010 78
B112160918035 88
A107180908021 98
1 A
2 107
3 180908
2 999
```

输出样例：

```
Case 1: 1 A
A107180908108 100
A107180908021 98
A112180318002 98
Case 2: 2 107
3 260
Case 3: 3 180908
107 2
123 2
102 1
Case 4: 2 999
NA
```

```cpp
#include <iostream>
#include <vector>
#include <unordered_map>
#include <algorithm>
using namespace std;
struct node {
    string t;
    int value;
};
bool cmp(const node &a, const node &b) {
    return a.value != b.value ? a.value > b.value : a.t < b.t;
}
int main() {
    int n, k, num;
    string s;
    cin >> n >> k;
    vector<node> v(n);
    for (int i = 0; i < n; i++)
        cin >> v[i].t >> v[i].value;
    for (int i = 1; i <= k; i++) {
        cin >> num >> s;
        printf("Case %d: %d %s\n", i, num, s.c_str());
        vector<node> ans;
        int cnt = 0, sum = 0;
        if (num == 1) {
            for (int j = 0; j < n; j++)
```

```
                if (v[j].t[0] == s[0]) ans.push_back(v[j]);
        } else if (num == 2) {
            for (int j = 0; j < n; j++) {
                if (v[j].t.substr(1, 3) == s) {
                    cnt++;
                    sum += v[j].value;
                }
            }
            if (cnt != 0) printf("%d %d\n", cnt, sum);
        } else if (num == 3) {
            unordered_map<string, int> m;
            for (int j = 0; j < n; j++)
                if (v[j].t.substr(4, 6) == s) m[v[j].t.substr(1, 3)]++;
            for (auto it : m) ans.push_back({it.first, it.second});
        }
        sort(ans.begin(), ans.end(),cmp);
        for (int j = 0; j < ans.size(); j++)
            printf("%s %d\n", ans[j].t.c_str(), ans[j].value);
        if (((num == 1 || num == 3) && ans.size() == 0) || (num == 2 && cnt ==
0)) printf("NA\n");
    }
    return 0;
}
```