思考题（1）

问题描述：

编写一个程序，从键盘输入一个字符串（长度不超过100个字符），然后将该字符串按如下要求处理：

（1）将大于原首字符的各字符按原来相互间的顺序关系都集中保存在原首字符的左边；

（2）将小于等于原首字符的各字符（按ASCII码值）升序都集中保存在原首字符的右边。

经过上述处理之后，将会得到一个新的字符串，然后将该字符串打印出来。

思考题（1）

说明：（1）可以使用字符数组存储输入的字符串；（2）由于输入的字符串当中可能包含有空白字符（如空格、Tab键），所以应该使用gets函数来读取这个字符串，不要用scanf函数；（3）输入的字符可能包含大小写或标点符号；（4）提示：可以使用gets、puts、strcpy、strcat等字符串处理函数。

样例输入：

abcdABCD1234

样例输出：

bcda1234ABCD

```c
#include <stdio.h>
#include <string.h>

// 冒泡排序
void sort(char a[], int n)
{
        int i, j;
        char temp;
        for(j = 0; j < n - 1; j++)
        {
                for(i = n - 1; i >= j + 1; i--)
                {
                        if(a[i] < a[i - 1])
                        {
                                temp = a[i];
                                a[i] = a[i - 1];
                                a[i - 1] = temp;
                        }
                }
        }
}
```

```
int main()
{
        char str[101], strLarge[101] = {0}, strSmall[101]={0};
        int i, j, k, l, n;

        gets(str);
        n = strlen(str);

        j = k = 0;
        for(i = 1; i < n; i++)
        {
                if(str[i] > str[0])
                        strLarge[j++] = str[i];
                else
                        strSmall[k++] = str[i];
        }

        strLarge[j] = str[0];
        sort(strSmall, k);

        strcpy(str, strLarge);
        strcat(str, strSmall);
        puts(str);

        return 0;
}
```

数组必须初始化吗？

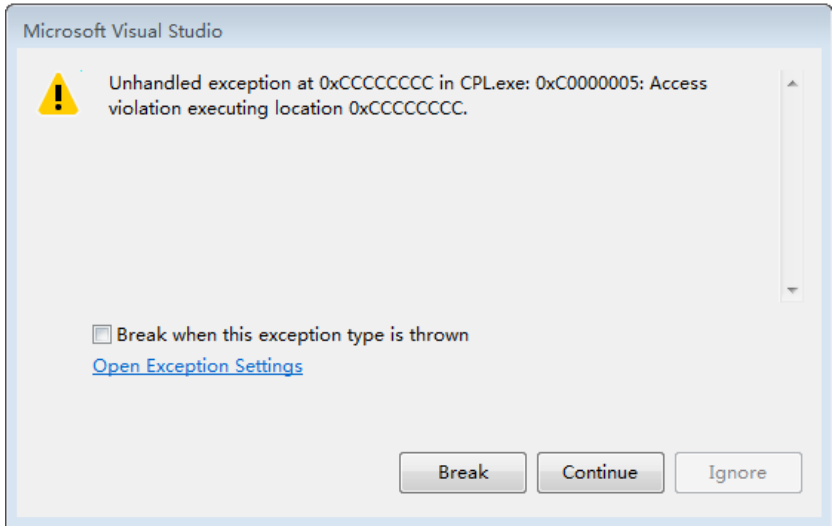
strLarge[101] = {0}, strSmall[101]={0};

abcdABCD1234

bcda1234ABCD

```
int main()
{
        char str[101], strLarge[101], strSmall[101];
        int i, j, k, l, n;

        gets(str);
        n = strlen(str);

        j = k = 0;
        for(i = 1; i < n; i++)
        {
                if(str[i] > str[0])
                        strLarge[j++] = str[i];
                else
                        strSmall[k++] = str[i];
        }

        strLarge[j] = str[0];
        sort(strSmall, k);

        strcpy(str, strLarge);
        strcat(str, strSmall);
        puts(str);

        return 0;
}
```

数组没有初始化，strcpy会导致严重后果



思考题（2）

版本号排序：

软件的版本号只由非负整数和 ‘.’组成，比如：1.3、0.1、3.0.61、3.1.12.5、3.1.12.4。

编写一个程序，程序第一行输入版本号的个数n（1≤n≤100），第二行输入n个字符串，每个字符串表示一个版本号（长度小于100），版本号之间用空格隔开，请将n个版本号从小到大排序并输出。

思考题（2）

样例输入：

5

1.3 0.1 3.0.61 3.1.12.5 3.1.12.4

样例输出：

0.1 1.3 3.0.61 3.1.12.4 3.1.12.5

一种实现方法

```
#define _CRT_SECURE_NO_WARNINGS
#include <stdio.h>
#include "string.h"
#define M 100
#define N 100
bool CompareVersion(char v1[], char v2[]);
void Swap(char a[], char b[]);
void Sort(char str[][N], int n);
void Print(char str[][N], int n);
int main()
{
        int i, n;        char str[M][N];
        scanf("%d", &n);
        for (i = 0; i < n; i++)   scanf("%s", str[i]);
        Sort(str, n);
        Print(str, n);
        return 0;
}
```

```
void Sort(char str[][N], int n) //冒泡排序
{
        char min[N];
        int i, j;
        for (i = 0; i < n - 1; i++)
        {
                for (j = i; j < n - 1; j++)
                {
                        if (CompareVersion(str[j], str[j+1]))
                        {
                                Swap(str[j], str[j+1]);
                        }
                }
        }
}
```

```
void Swap(char a[], char b[])
{
        char temp[N];
        strcpy(temp, a);
        strcpy(a, b);
        strcpy(b, temp);
}

void Print(char str[][N], int n)
{
        //输出
        int i;
        for (i = 0; i < n; i++)
        {
                printf("%s\n", str[i]);
        }
}
```

```
//v1>v2 返回true
bool CompareVersion(char v1[], char v2[])
{
        int len1 = strlen(v1); int len2 = strlen(v2);
        int i = 0; int j = 0;
        while (i < len1 || j < len2)
        {
                int sum1 = 0;
                int sum2 = 0;
                while (i < len1 && v1[i] != '.')
                {
                        sum1 = sum1 * 10 + v1[i] - '0';
                        i++;
                }
                while (j < len2 && v2[j] != '.')
                {
                        sum2 = sum2 * 10 + v2[j] - '0';
                        j++;
                }
                if (sum1 > sum2){
                        return true;
                }
                else if(sum1 < sum2){
                        return false;
                }
                i++; j++;
        }
        return false;
}
```

```
5
1.3 0.1 3.0.61 3.1.12.5 3.1.12.4
0.1 1.3 3.0.61 3.1.12.4 3.1.12.5
```

冒泡排序again

问题描述：

“软件工程（I）”期末考试结束了，将要进行成绩排名。排名的规则如下：

- 以考试成绩为排名依据(分数为整数)；
- 对学生A，若有K个学生的成绩比他/她高，则其排名为K+1；
- 若学生A和B的成绩相同，则他们的排名也相同，但学号小的放在前面。

样例输入

```
5
1006 95
1002 94
1007 100
1000 95
1001 100
```

样例输出

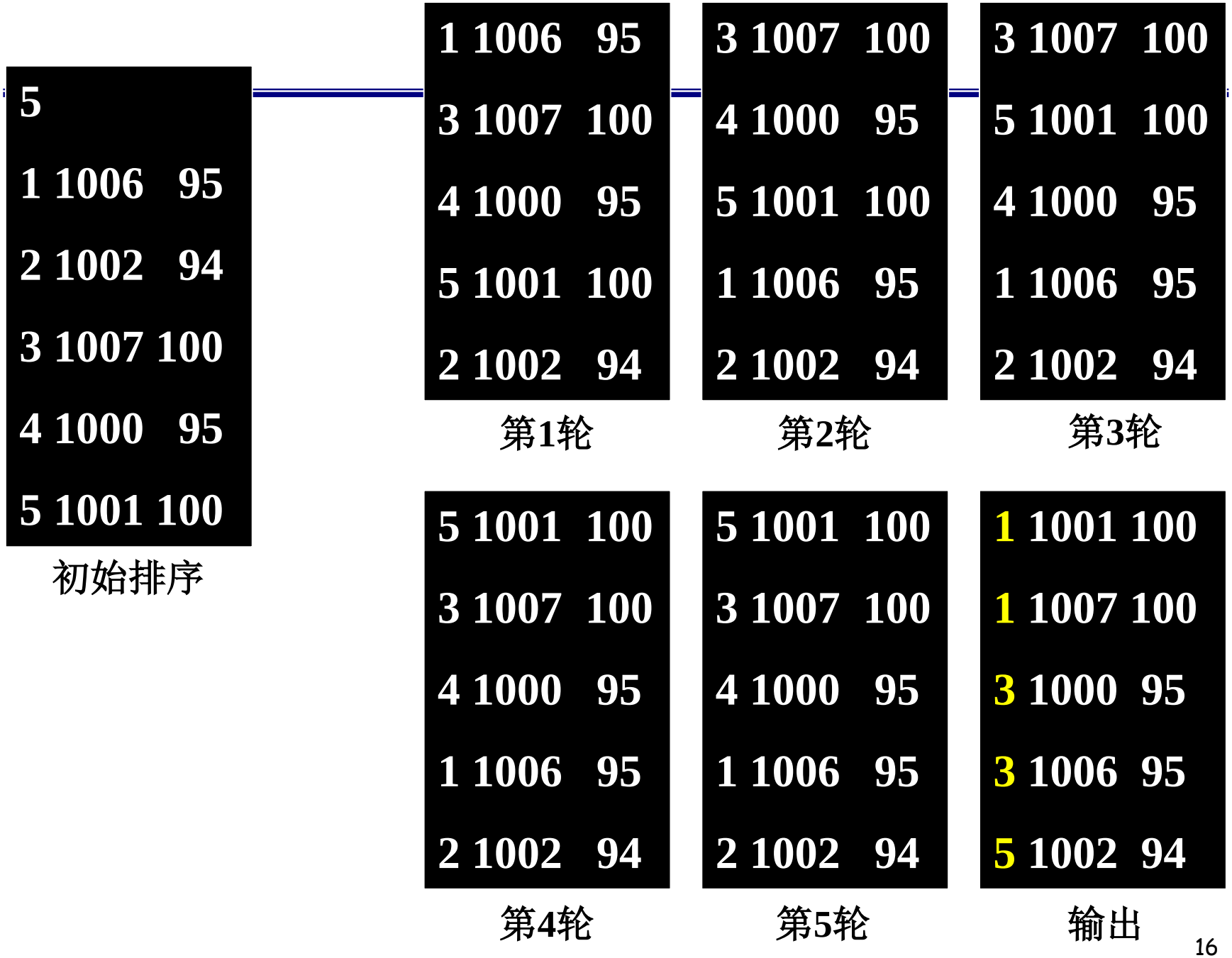
```
1 1001 100
1 1007 100
3 1000 95
3 1006 95
5 1002 94
```

实现方法(2)

```
#include<stdio.h>

int score[110],stu[110],id[110];

int cmp(int x,int y)
{
        if(score[x]!=score[y])
        {
                return score[x]>score[y];
        }
        return stu[x]<stu[y];
}
```

实现方法(1)

```
int main() {
        int n;
        scanf("%d",&n);
        for(int i=1;i<=n;i++) {
                scanf("%d%d",&stu[i],&score[i]);
                id[i]=i;
        }
        for(int i=1;i<=n;i++) {
                for(int j=1;j<=n-1;j++) {
                        if(cmp(id[j],id[j+1])==0) {
                                int t=id[j];
                                id[j]=id[j+1];
                                id[j+1]=t;
                        }
                }
        }
        int rk=0,Truerk=0;
        for(int i=1;i<=n;i++) {
                Truerk++;
                if(i==1||score[id[i]]!=score[id[i-1]]) {
                        rk=Truerk;
                }
                printf("%d %d %d\n",rk,stu[id[i]],score[id[i]]);
        }
        return 0;
}
```


5
1 1006 95
2 1002 94
3 1007 100
4 1000 95
5 1001 100
初始排序
1 1006 95
3 1007 100
4 1000 95
5 1001 100
2 1002 94
第1轮
3 1007 100
4 1000 95
5 1001 100
1 1006 95
2 1002 94
第2轮
3 1007 100
5 1001 100
4 1000 95
1 1006 95
2 1002 94
第3轮
5 1001 100
3 1007 100
4 1000 95
1 1006 95
2 1002 94
第4轮
5 1001 100
3 1007 100
4 1000 95
1 1006 95
2 1002 94
第5轮
1 1001 100
1 1007 100
3 1000 95
3 1006 95
5 1002 94
输出

实现方法(1)

```
#include <stdio.h>
#include <stdlib.h>
#include <math.h>

#define isequal(a, b)  (fabs(double(a - b)) < 10e-3) // 处理浮点数成绩

struct student
{
    int NO;      // 排名(序号)
    int ID;      // 学号
    int score;   // 成绩(可以处理实数)
};

void swap(student *s1, student *s2);
void sortScore(student *s, int n);
void sortID(student *s, int m);
```

```
void swap(student *s1, student *s2)
{
    student temp;

    temp.ID = s1->ID;
    temp.score = s1->score;

    s1->ID = s2->ID;
    s1->score = s2->score;

    s2->ID = temp.ID;
    s2->score = temp.score;
}
```

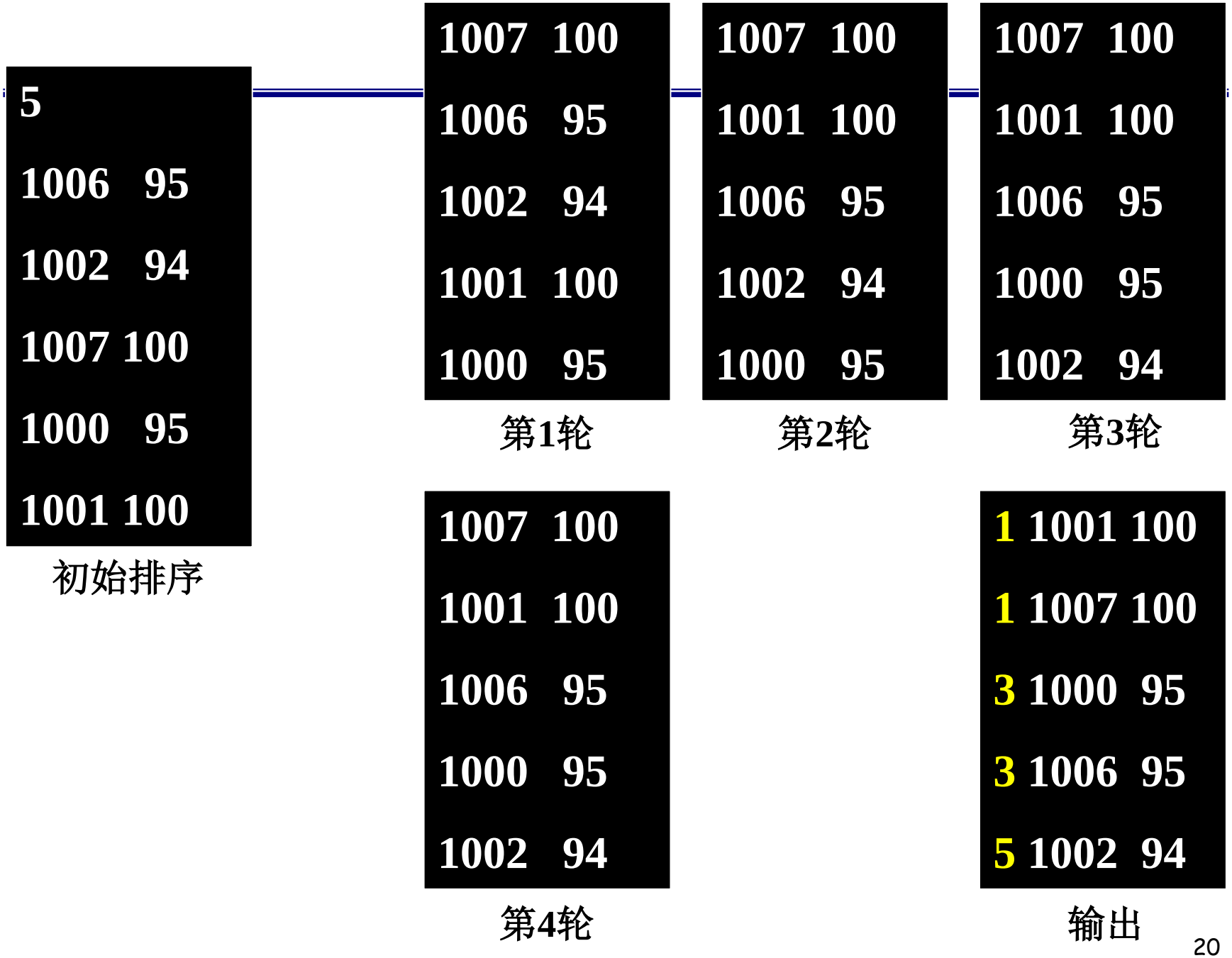
实现方法(2)

```
void sortScore(student *s, int n)
{
    int i, j;

    for (j=0; j<n-1; j++)
    {
        for (i=n-1; i>j; i--)
        {
            if(s[i].score > s[i-1].score)
                swap(&s[i], &s[i-1]);
        }
    }
}

void sortID(student *s, int m)
{
    int i, j;

    for (j=0; j<m-1; j++)
    {
        for (i=m-1; i>j; i--)
        {
            if(s[i].ID < s[i-1].ID)
                swap(&s[i], &s[i-1]);
        }
    }
}
```

```c
int main()
{
    int i, j, m, n;
    student *stu;
    scanf("%d", &n);
    stu = (student *)malloc(n * sizeof(student));
    for (i=0; i<n; i++)
        scanf("%d %d", &stu[i].ID, &stu[i].score);
    // 先按成绩排序
    sortScore(stu, n);

    int start = 0;
    for (i = 0;  i < n;  i = i+1)
    {
        start = i;
        m = 1;
        while (isequal(stu[i].score, stu[i+1].score))
        {
            i++;
            m++;
        }

        // 如果成绩相同再按学号进行排序
        if(m > 1)
            sortID(&stu[start], m);

        // 最后再计算排名
        for (j=0; j<m; j++)
            stu[start+j].NO = start + 1;
    }

    for (i = 0;  i < n;  i = i+1)
        printf("%d %4d %d\n", stu[i].NO, stu[i].ID, stu[i].score);

    free(stu);
    return 0;
}
```

```
5
1006 95
1002 94
1007 100
1000 95
1001 100
```

```
1 1001 100
1 1007 100
3 1000 95
3 1006 95
5 1002 94
```


5
1006 95
1002 94
1007 100
1000 95
1001 100
初始排序
1007 100
1006 95
1002 94
1001 100
1000 95
第1轮
1007 100
1001 100
1006 95
1002 94
1000 95
第2轮
1007 100
1001 100
1006 95
1000 95
1002 94
第3轮
1007 100
1001 100
1006 95
1000 95
1002 94
第4轮
1 1001 100
1 1007 100
3 1000 95
3 1006 95
5 1002 94
输出