2018-2019 秋季学期程序设计 I 期末测试

中国人民大学　信息学院

注意事项

1. 本次测试的时间为 150 分钟；编程结果采用机器自动评测。
2. 共 6 题，按评测得分从高到低，在总成绩中实际所占百分比依次为 25、25、15、15、10、10。
3. 提交到在线评测系统中的程序均采用标准输入和标准输出（键盘输入和屏幕输出）。
4. 程序设计语言选用 C 或 C++。
5. 所有题目的时间限制均为 1s。

1. 提货单

提货单

有一份提货单，其数据项目有：商品名（MC）、单价（DJ）、数量（SL）。请计算并输出提货单的总金额。

输入格式

第一行是提货单中记录个数N (N<100)。
接下来N行，每一行是一条记录，包含商品名（长度不超过100的仅包含小写字母的字符串）、单价、数量，单价和数量均为不大于500整数，数据项之间用一个空格隔开。

输出格式

一个整数，表示这份提货单的总金额。

评测样例

- 样例 >>

```
4
book 12 3
pen 2 10
computer 480 1
flower 47 5
```

- << 样例

```
771
```

参考代码

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

typedef struct _GOODS
{
    char Name[128];
    int Price;
    int Count;
}Goods;

int main()
{
```

```
14      Goods Gds[100];
15      int N;
16      scanf("%d", &N);
17      for(int i = 0; i < N; i++)
18      {
19          scanf("%s%d%d", Gds[i].Name, &Gds[i].Price, &Gds[i].Count);
20      }
21      int Sum = 0;
22      for(int i = 0; i < N; i++)
23      {
24          Sum += Gds[i].Price * Gds[i].Count;
25      }
26      printf("%d", Sum);
27      return 0;
28  }
```

2. 获取密码

获取密码

某人有一个保险箱，内存贵重物品。他特别害怕忘记了保险箱密码，就将密码记在纸条上，又担心保险箱密码失窃。为此，他设计一个给保险箱密码加密的方法。保险箱密码为一串数字，加密后还是一串数字，根据下面规则可以得到明码：首先删除第一个数，紧接着将第二个数放到这串数字的末尾；再将新数字串的第一个数删除，并将第二个数放到这串数字的末尾；如此循环，直到剩下最后一个数；将最后这个数也删除；按照刚才删除的顺序，将这些数字连在一起就是明码。

输入格式

输入保险箱密码的密码，即一串十进制数字，长度在6位至9位之间，包括6位和9位。

输出格式

保险箱密码的明码，即长度在6位至9位之间的一串十进制数字，包括6位和9位。

评测样例

- 样例 >>

```
631758924
```

- << 样例

```
615947283
```

数据规模和约定

不含数字0的输入数据占50%。

参考代码

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

int main()
{
    char Raw[10] = {0};
    gets(Raw);
    int Outs[10];
    int OutLen = 0, TmpLen = strlen(Raw);
    char First;
```

```
13      while(TmpLen > 0)
14      {
15          char Tmp[10] = {0};
16          Outs[OutLen++] = Raw[0] - '0';
17          First = Raw[1];
18          strcpy(Tmp, Raw + 2);
19          memset(Raw, 0, 10);
20          strcpy(Raw, Tmp);
21          Raw[strlen(Raw)] = First;
22          TmpLen--;
23      }
24      for(int i = 0; i < OutLen; i++)
25      {
26          printf("%d",Outs[i]);
27      }
28      return 0;
29  }
```

3. 求区间

求区间

给定一个正整数n，求闭区间内连续正整数之和等于n的所有区间[a, b]（a、b均为正整数,且b>a）。

输入格式

一行，包含一个正整数n。

输出格式

若干行，每行一个闭区间，该区间内所有正整数之和为n。按照字典序输出。注意，所有字符均为英文状态下，‘,’后面有一个空格。

评测样例

- 样例 1 ≫

```
15
```

- ≪ 样例 1

```
[1, 5]
[4, 6]
[7, 8]
```

- 样例 2 ≫

```
100
```

- ≪ 样例 2

```
[9, 16]
[18, 22]
```

数据规模和约定

60%的数据 的数据 3≤n≤1000
100%的数据的数据 3≤n≤10^7 (10的7次方)

参考代码

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

int main()
```

```c
{
    int N;
    scanf("%d", &N);
    for(int i = 1; i <= N / 2; i++)
    {
        for(int j = i; j <= N / 2 + 1; j++)
        {
            if((i + j) * (j - i + 1) == 2 * N)
            {
                printf("[%d, %d]\n", i, j);
            }
            if((i + j) * (j - i + 1) > 2 * N)
                break;
        }
    }
    return 0;
}
```

4. 名次查询

名次查询

信息学院一年级学生期末考试后，需对成绩进行评估排名。排名规则是：1）按成绩从高到低排序；2）同分名次相同；3）名次从1开始，某分数所在的名次由超过该分数的总人数决定。e.g. 若超过90分的总人数是30，则90分对应名次是31。
请编程完成具体的学号和成绩输入后，查询指定学号的名次。注意，本题中学号以字符串表示，最长不超过20个字符，且不会重复。

输入格式

共有n+1行，第1行一个整数n和一个学号；前者表示学生人数n < 200，后者为待查的学号，空格分隔，在主函数中完成输入；
第2行起，每行两个数据，第1个为学号（字符串），第2个为整型的分数，空格分隔。这n行数据在子函数中完成输入。

输出格式

1个整数，为名次。

评测样例

- 样例 1 ≫

```
7 112
112 89
211 89
131 80
113 92
158 92
119 60
141 80
```

- ≪ 样例 1

```
3
```

- 样例 2 ≫

```
5 211
112 85
211 85
131 85
113 85
158 85
```

- ≪ 样例 2

```
1
```

数据规模和约定

本题为完成函数题。该函数完成n对学生学号与成绩的输入，并根据指定的学号查询其名次，将结果存放在指定地址内。数据保证被查询学号有且仅有一次。请你按main函数中的调用方式及如下的函数参数说明，完成所需的Query子函数。

程序代码框架

```
//
// Query函数参数说明
// int n, 表示学生总人数
// char *id 指向待查学生学号的字符指针
// int *prk 存放查询结果的地址
//================程序开始==================
#include <stdio.h>
#include <stdlib.h>
#include <string.h>

//TODO ...

// Query函数，它本身可再调用其它子函数。
//void Query(int n, char *id, int *prk);
//@{你的代码}

int main(){

    char stu_id[21];        //学生学号，以字符表示
    int rank = 0;           //名次
    int n;              //总人数

    scanf("%d%s", &n, stu_id);
    Query(n, stu_id, &rank);

    printf("%d\n", rank);
    return 0;
}
//================程序结束==================

```

参考代码

```
//
// Query函数参数说明
// int n, 表示学生总人数
// char *id 指向待查学生学号的字符指针
// int *prk 存放查询结果的地址
//================程序开始==================
#include <stdio.h>
#include <stdlib.h>
#include <string.h>

typedef struct _STU
{
    char Num[21];
    int Grade;
    int Rank;
}Students;

int Cmp(const void * A, const void *B)
{
    Students * StuA = (Students *) A;
```

```
21      Students * StuB = (Students *) B;
22      return -(StuA[0].Grade - StuB[0].Grade);
23  }
24
25  // Query函数，它本身可再调用其它子函数。
26  void Query(int n, char *id, int *prk)
27  {
28      Students Stus[200];
29      for(int i = 0; i < n; i++)
30          scanf("%s%d", Stus[i].Num, &Stus[i].Grade);
31
32      qsort(Stus, n, sizeof(Students), Cmp);
33      Stus[0].Rank = 1;
34      if(strcmp(id, Stus[0].Num) == 0)
35      {
36          *prk = Stus[0].Rank;
37          return;
38      }
39      for(int i = 1; i < n; i++)
40      {
41          if(Stus[i].Grade == Stus[i - 1].Grade)
42          {
43              Stus[i].Rank = Stus[i - 1].Rank;
44          }
45          else Stus[i].Rank = i + 1;
46
47          if(strcmp(id, Stus[i].Num) == 0)
48          {
49              *prk = Stus[i].Rank;
50              return;
51          }
52      }
53
54  }
55
56  int main(){
57
```

```
58      char stu_id[21];          //学生学号，以字符表示
59      int rank = 0;             //名次
60      int n;                //总人数
61
62      scanf("%d%s", &n, stu_id);
63      Query(n, stu_id, &rank);
64
65      printf("%d\n", rank);
66      return 0;
67  }
68  //=================程序结束====================
```

5. 单词统计

单词统计

在信息检索中，单词的文档频率（Document Frequency，简称DF）是评估单词重要性的重要参数。针对一组文档，某个单词w的DF定义为该单词出现的文档的个数。注意：不是单词在这组文档中出现的总次数，而是包含该单词的文档的个数。若一篇文档包含某个单词多次，DF不重复计数。例如：如果w在10篇文档中出现过，其DF的值为10。
此外，由于同一个单词可能会有不同的大小写形态，需要将单词都归一化为小写形态后再进行DF的统计。
请你设计一个简易的单词统计程序，输入D篇文档，统计文档频率DF最大的前k个归一化为小写后的单词。对于结果中DF相同的单词，按其字典序排序。

输入格式

第一行为两个数字D和K，中间以空格分隔，
以下D行每行一个英文文档（字符长度小于1000），单词之间以空格分隔

输出格式

共K行，每行先输出归一化的单词，再输出其文档频率，中间以空格分隔。

评测样例

- 样例 1 >>

```
2 3
We live in this world when we love it
What you are you do not see what you see is your shadow
```

- << 样例 1

```
are 1
do 1
in 1
```

- 样例 2 >>

```
3 6
I know how poor I am when I see you said the Work to the Word
One word keep for me in thy silence O World when I am dead I
have loved
Thought feeds itself with its own words and grows
```

- << 样例 2

```
am 2
i 2
when 2
word 2
and 1
dead 1
```

- 样例 3 >>

```
3 5
I cannot choose the best The best chooses me
He could have chosen ME
They chose to take responsibility for their education and set
goals
```

- << 样例 3

```
me 2
and 1
best 1
cannot 1
choose 1
```

数据规模和约定

【数据规模】
文档个数D <= 110；单词总数小于等于1000；每个单词的长度不超过30个字符。
【特别说明】
第一行数字输入后的“回车换行符”可能被当做一个字符串接受，请注意特殊处理，例如可以用getchar()将回车符跳过后再继续。

参考代码

```
//================程序开始==================
#include <stdio.h>
#include <stdlib.h>
#include <string.h>
#include <stdbool.h>

void my_strlwr(char Str[])
{
    int Len = strlen(Str);
    for(int i = 0; i < Len; i++)
        if(Str[i] >= 'A' && Str[i] <= 'Z')
            Str[i] += 32;
}

typedef struct _RECORD
{
    char Word[32];
    int Times;
    int Latest;
}Record;

Record RcdTb[1024];
int NowWdCnt = 0;

int Cmp(const void * A, const void * B)
{
    Record * RcdA = (Record *)A;
    Record * RcdB = (Record *)B;
    if((RcdA -> Times) != (RcdB -> Times))
        return -((RcdA -> Times) - (RcdB -> Times));
```

```
31      else return strcmp(RcdA -> Word, RcdB -> Word);
32  }
33
34  void InputWord(int NowD, char * In)
35  {
36      char * Token = strtok(In, " ");
37      bool IsMatched = false;
38      while(Token != NULL)
39      {
40          IsMatched = false;
41          for(int i = 0; i < NowWdCnt; i++)
42          {
43              if(strcmp(Token, RcdTb[i].Word) == 0)
44              {
45                  IsMatched = true;
46                  if(NowD > RcdTb[i].Latest)
47                  {
48                      RcdTb[i].Times++;
49                      RcdTb[i].Latest = NowD;
50                  }
51                  break;
52              }
53          }
54          if(!IsMatched)
55          {
56              strcpy(RcdTb[NowWdCnt].Word, Token);
57              RcdTb[NowWdCnt].Times = 1;
58              RcdTb[NowWdCnt++].Latest = NowD;
59          }
60          Token = strtok(NULL, " ");
61      }
62  }
63  int main(){
64      int D, K;
65      scanf("%d%d", &D, &K);
66      getchar();
67      for(int i = 0; i < 1024; i++)
```

```
68      {
69          RcdTb[i].Latest = -1;
70          memset(RcdTb[i].Word, 0, sizeof(RcdTb[0].Word));
71          RcdTb[i].Times = 0;
72      }
73      for(int i = 0; i < D; i++)
74      {
75          char Tmp[1024] = {0};
76          gets(Tmp);
77          my_strlwr(Tmp);
78          InputWord(i, Tmp);
79      }
80      //printf("%d\n", NowWdCnt);
81      qsort(RcdTb, NowWdCnt, sizeof(Record), Cmp);
82      for(int i = 0; i < K; i++)
83          printf("%s %d\n", RcdTb[i].Word, RcdTb[i].Times);
84      return 0;
85  }
86  //================程序结束==================
```

6. 连连看游戏 （讲解过，此处略）