北京航空航天大学计算机系考研复试 06-14 上机真题及答案

复试上机指导

1. 本真题只是提供辅助作用，关键还是研友平时动手能力练习和对算法、数据结构的理解，参加过 ACM 的有一定优势 没参加过的也不用紧张，北航的上机题相对于清华和北大，难度上小很多，多练习的话，问题不大;
2. 上机时，可以快速阅读所有的题目，按照从易到难的次序做题，保证会的一定得分;
3. 熟悉编程环境，熟悉 c 的常用函数;
4. 为了快速测试代码的正确性，尤其是矩阵输入的情况，可以利用标准输入重定向，

   `freopen("c:\\input.txt","r",stdin);`加快测试过程;
5. 注意程序边界条件的测试;
6. 如果你有什么疑问，或者我们提供的材料有问题，欢迎联系我们：bwiunbuaa@163.com 提供北航计算机报考和选导师指导，或者到 kao400.com 给我们留言。

14 年上机题

第一题，阶乘数。输入一个正整数，输出时，先输出这个数本身，跟着一个逗号，再输出这个数的各位数字的阶乘和，等号，阶乘和的计算结果，并判断阶乘和是否等于原数，如果相等输出 Yes，否则输出 No。题目说明输入的正整数以及其各位阶乘和都不会超出 int 型的表示范围。

输入样例 1：

145

输出样例 1：

145,1!+4!+5!=145

Yes

输入样例 2：

1400

输出样例 2：

1400,1!+4!+0!+0!=27

No

第二题，五子棋。输入一个 19*19 的矩阵，只包含数字 0、1、2，表示两人下五子棋的棋牌状态，1、2 分别表示两人的棋子，0 表示空格。要求判断当前状态下是否有人获胜（横向、竖向或者斜线方向连成 5 个同色棋子）。题目说明输入样例保证每条线上至多只有连续 5 个同色棋子，并且保证至多只有 1 人获胜。如果有人获胜，输出获胜者（1 或 2）加一个冒号，接着输出获胜的五连珠的第一个棋子的坐标，从上到下从左到右序号最小的为第一个，序号从 1 开始编号。如果无人获胜，输出 no。

样例略。

第三题，排版题。输入若干行字符，表示某电影的演职员表，每行只有一个冒号，冒号前面是职位，冒号后面是姓名，要求把各行

冒号对齐，删除多余空格后输出。先输入一个数字，表示排版要求的冒号位置，该位置号保证比各行冒号前的最大字符数还要大。再输入若干行字符，最多 50 行，每行最多 100 个字符，除空格、制表符和回车之外都是有效字符，要求每行的冒号处于格式要求的位置，冒号两边与有效单词之间各有一个空格，冒号前面的单词之间只有一个空格（删除多余的空格和制表符），在冒号左边右对齐，前面全由空格填充，冒号后面的单词之间也只有一个空格，在冒号右边左对齐，最后一个单词后不加空格直接换行。

13 年上机题

第一题，给一个真分数的分子分母，输出约分后的分子分母，送分题，25 分；第二题，简单版八皇后，15 分；第三题，给出一个标准输入的正数（开头末尾没有多余的 0），输出其科学计数法表示结果。比如：输入 0.000002，输出 2e-6；输入 123.456，输出 1.23456e2；输入 123456，输出 1.23456e2

12 年上机题

1.【问题描述】

某些整数能分解成若干个连续整数的和的形式，例如

15 = 1 + 2+3+4+5

15 = 4 + 5 + 6

15 = 7 + 8

某些整数不能分解为连续整数的和，例如：16

输入：一个整数 N（N <= 10000）

输出：整数 N 对应的所有分解组合，按照每个分解中的最小整数从小到大输出，每个分解占一行，每个数字之间有一个空格（每行最后保留一个空格）；如果没有任何分解组合，则输出 NONE。

解题思路：

根据题目，任何可以进行分解的整数，必然满足(m+n)(n-m+1)/2 的形式，可以暴力尝试所有 m 和 n 组合，如果满足则输出，否则输出 None。

代码：

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

int main(){
    int n;
    while(scanf("%d",&n) != EOF){
        int begin,end;
        int found = 0;
        for(begin=1;begin<n;begin++){
            for(end=begin+1;end<n;end++){
                // 连续整数求和
                int sum = (begin + end)*(end-begin+1)/2;
            if(sum == n){ // 可以分解，输出结果
                    found = 1;
                    int i;
                    for(i=begin;i<=end;i++){
                        printf("%d ",i);
                    }
                    printf("\n");
                }
            }
        }
        if(found == 0){
            printf("NONE\n");
        }
    }
```

```
    return 0;
}
```

2.【问题描述】

小岛面积

```
1 1 1 1 1 1
1 1 0 0 0 1
1 0 0 0 1 0
1 1 0 1 1 1
0 1 0 1 0 0
1 1 1 1 1 1
```

上面矩阵的中的 1 代表海岸线，0 代表小岛。求小岛面积（即被 1 中包围的 0 的个数）。注意：仅求这样的 0，该 0 所在行中被两个 1 包围，该 0 所在列中被两个 1 包围。

输入：

第一行输入一个整数 N，表示输入方阵的维数

输入一个 N 维方阵

输出：

小岛面积

样例输入：

```
6
1 1 1 1 1 1
1 1 0 0 0 1
1 0 0 0 1 0
1 1 0 1 1 1
0 1 0 1 0 0
1 1 1 1 1 1
```

样例输出：

```
8
```

解题思路：

理解题目本身意思，可以发现对于矩阵中的 0 是否属于内陆，取决于该 0 所处的行和列上，如果 0 满足，如下条件则 0 为内陆，否则不是。

- 0 所在的行，0 的左边和右边必须有 1
- 0 所在的列，0 的上面和下面必须有 1

所以，解题思路就是，遍历所有的行和列，记录改行或列，最左面和最右面（或者最上面和最下面）1 的坐标，然后当遇到 0，判断是否处于记录的值的中间，是，则是内陆，面积加 1，否则不加。

代码：

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

#define DEBUG_BUAA_122
```

```
int main(){
    int N; //矩阵维数
    int island[100][100];    //输入岛数据的方阵
    int data[100][4];

    #ifdef DEBUG_BUAA_122
        freopen("/Users/bwiunbuaa/tmp/oj/buaa_122.in", "r", stdin);
    #endif /* JOBDU_H_ */
    while(scanf("%d",&N) != EOF){
        int i,j;
        //初始化
        memset(data,-1,100*4*sizeof(int));
        //读入数据
        for(i=0;i<N;i++){
            for(j=0;j<N;j++){
                scanf("%d",&island[i][j]);
            }
        }

        for(i=0;i<N;i++){
            for(j=0;j<N;j++){

                if(island[i][j] == 1){   //小岛边界

                    if(data[i][0] == -1){
                        data[i][0] = j;
                    }
                    if(data[j][2] == -1){
                        data[j][2] = i;
                    }
                    data[i][1] = j;
                    data[j][3] = i;
                }
            }
        }
        int area = 0;
        for(i=0;i<N;i++){
            for(j=0;j<N;j++){
                if(island[i][j] == 0){   //小岛边界
                    if(i > data[j][2] && i < data[j][3] && j > data[i][0]
&& j < data[i][1]){
                        area++;
                        //printf("i=%d,j=%d\n",i,j); for debug
```

```
                }
            }
        }
    }
    printf("%d\n",area);
}
return 0;
}
```

3.【问题描述】

统计关键字出现位置

输入：

一行标准 c 语言代码（字符个数小于 300），统计出该字符串中关键字的 if，while，for 所在的位置，按照关键字出现的顺序依次输出。注意双引号内的不需要统计。

输入：一行标准 c 语言代码，字符个数小于 300

输出：

关键字 if，while，for 对应的位置，按照关键字出现的顺序依次输出。输出格式为：关键字，后跟冒号，然后是出现的位置。扫描到关键字就输出，每个输出占一行。

样例输入：

```
#include <stdio.h> int main() {int i = 0; if(i == 0) printf("YES"); return 0;}
#include <stdio.h> int main() {int ifwhile = 0; int forif = 1;char if_for_while = 'a';char *str = "while"; while(ifwhile == 0) {ifwhile = 1;forif = 0;} if(forif == 0) {if_for_while = 'b';} if(ifwhile == 1) {if_for_while = 'c';} return 0;}
```

样例输出：

```
if:43
while:88
if:133
if:170
```

解题思路：

首先把输入字符串切分为一个个的单词，然后对每个单词进行匹配。注意 c 语言中的 strtok 函数，不能够返回位置（当分隔符连续时，比如+=,计算出前导的分割符有几个），所以不能满足本题的要求，必须自己写获取单词的函数（反正也不复杂啦）。因为考虑到引号的问题，我们的解决方案是首先对输入字符串进行预处理，将“ ”内的每个字符都替换为#，然后再进行单词分割（切词），最后进行匹配。具体步骤：

- 输入源码字符串预处理，将”” 内的每个字符替换为#
- 一次获取源码中的单词，看是否为查找的关键词，如果是则输出

代码：

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

```
#define DEBUG_BUAA_123
#define MAX_WORD_LEN 128
#define MAX_LEN 300
#define KW_NUM 3

/**
 * 将一段源代码中间的，位于双引号“”的以#代替
 */
void replaceQuota(char buf[]){
    int i;
    int len = strlen(buf);
    int quotaL = 0,quotaR = 0;
    for(i=0;i<len;i++){
        if(buf[i] == '"'){
            if(quotaL == 0){  //左引号
                quotaL = 1;
            }else{ //右引号
                quotaL = quotaR = 0;
            }
        }
        else{
            if(quotaL == 1){  //该字符属于引号之间的，替换为#
                buf[i] = '#';
            }
        }
    }
}

// 判断字符是否为分隔符
int isdelim(char ch){
    switch(ch){
        case ' ':
        case '=':
        case ',':
        case ';':
        case '<':
        case '>':
        case '(':
        case ')':
        case '{':
        case '}':return 1;
        default:return 0;
    }
```

```
}

/**
 * 从input数组下标pos处开始一个单词,返回的单词放在word数组中
 * 返回下次查找开始的位置
 */
int getWord(char input[],int pos,char word[]){
    int len = strlen(input);
    int i;
    int isbegin = 0;
    int wpos = 0;
    for(i=pos;i<len;i++){
        if(isdelim(input[i])){
            if(isbegin == 0){ //略过该分隔符
                continue;
            }
            else{  //单词获取结束
                break;
            }
        }
        else{
            isbegin = 1;
            word[wpos++] = input[i];
        }
    }
    word[wpos]='\0';  //结束标志，勿忘，切记
    return (i>=len) ? -1:i;  //如果到文件结尾了，返回-1
}


int main(){
    char input[300];
    char word[MAX_WORD_LEN];

    #ifdef DEBUG_BUAA_123
        freopen("/Users/bwiunbuaa/tmp/oj/buaa_123.in", "r", stdin);
    #endif /* JOBDU_H_ */

    char delim[] = " ,();\",+-*/=<>?:";    //单词分隔符

    while(gets(input) != NULL){
        //首先对input进行预处理，将引号中间的字符都转化为#
        replaceQuota(input);
        int beg_pos = 0;
```

```
        do
        {
            beg_pos = getWord(input,beg_pos,word);
            int wlen = strlen(word);
            //判断是否为指定的关键字
            if(strcmp("if",word) == 0){
                printf("if:%d\n",beg_pos-wlen+1);//位置以1开始，而不是0，所以要加1
            }
            else if(strcmp("while",word) == 0){
                printf("while:%d\n",beg_pos-wlen+1);
            }
            else if(strcmp("for",word) == 0){
                printf("for:%d\n",beg_pos-wlen+1);
            }
        }while(beg_pos >= 0);
    }
    return 0;
}
```

11 年上机题

1. 【问题描述】

孪生数定义： 如果 A 的约数（因数，包含 1，但不包含 A 本身）之和等于 B ， B 的约数（因数）之和等于 A ， A 和 B 称为孪生数（A 和 B 不相等）。试找出正整数 M 和 N 之间的孪生数。

输入：

从控制台输入两个正整数 M 和 N（1<=M<N<=20000），中间用一个空格分隔。

输出：

在标准输出上输出符合题目描述的 M 和 N 之间的全部孪生数对（包括 M 和 N）。每行输出一对孪生数，用一个空格隔开，小的先输出；各行孪生数按照第一个数从小到大的顺序输出，一对孪生数只输出一次。 如果没有符合要求的孪生数对，则输出字符串“NONE”。

输入样例

20 300

200 250

输出样例

220 284

NONE

样例说明

样例 1 输入的区间为[20,300]，其间有一对孪生数对，即：220（1+2+4+5+10+11+20+22+44+55+110=284）和 284（1+2+4+71+142=220）。样例 2 输入的区间是[200,250]，其间没有孪生数对，所以输出字符串：NONE。

评分标准

该题要求输出区间中的所有孪生数对，共有 5 个测试点，提交程序文件名为 example1.c 或 example1.cpp。

解题思路：

这题目很简单，比较每对数的约数和，看是否满足条件，如果满足则输出。

代码：

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

//计算x约数的和
int yinzisum(int x)
{
    int i=1,sum=0;
    if(x == 1) return 0;
```

```
    while(i<x)
    {
        if(x%i==0)
            sum=sum+i;
        i++;
    }
    return sum;
}

int main(){
    int x,y,flag=0,i,j;
    scanf("%d%d",&x,&y);
    int min,max;
    if(x > y){
        max=x;
        min=y;
    }else{
        max=y;
        min=x;
    }
    int* array=(int*)malloc(sizeof(int)*(max-min+1));
    for(i=0;i<max-min+1;i++)
        array[i]=yinzisum(min+i);
    for(i=0;i<max-min+1;i++)
        for(j=i+1;j<max-min+1;j++)
            if(array[i]==j+min && array[j]==i+min)
            {
                printf("%d %d\n",i+min,j+min);
                flag=1;
            }
    if(flag==0)
        printf("NONE\n");
    return 1;
}
```

2. 【问题描述】

先输入两个矩阵 A 和 B，然后输入替换位置（左上角），编写程序将矩阵 A 中从替换位置开始的子矩阵（与 B 同样大小）替换为 B，并输出替换后的矩阵。

【输入形式】

从控制台先输入矩阵 A 的行数和列数（行数和列数均大于等于 1，小于等于 20），然后在新的行上输入矩阵 A 的各行数字（以一个空格分隔的整数）。再以同样的方式输入矩阵 B。最后输入替换位置（用一个空格分隔的两个整数表示，行数和列数都从 1 开始计数，因此两个整数都大于等于 1）。若替换位置超出了矩阵 A 的行数或列数，则原样输出矩阵 A。

【输出形式】

在标准输出上分行输出替换后的矩阵，每行中各数字之间以一个空格分隔。

【输入样例 1】

```
5 6
10 2 34 -1 800 90
2 76 56 -200 23 1
35 0 0 98 8 3000
2000 100 -1 1 2 0
8 7 85 963 496 8
2 3
9 9 9
9 9 9
3 3
```

【输出样例 1】

```
10 2 34 -1 800 90
2 76 56 -200 23 1
35 0 9 9 9 3000
2000 100 9 9 9 0
8 7 85 963 496 8
```

【样例 1 说明】

输入的矩阵 A 为 5 行 6 列，矩阵 B 是 2 行 3 列，替换位置为第 3 行的第 3 列，即：将 A 中第 3 行第 3 列开始的、行数为 2 列数为 3 的子矩阵替换为 B。

【输入样例 2】

```
3 4
10 2 34 -1
2 76 56 -200
35 0 0 98
2 3
9 9 9
9 9 9
2 3
```

【输出样例 2】

```
10 2 34 -1
2 76 9 9
35 0 9 9
```

【样例 2 说明】

输入的矩阵 A 为 3 行 4 列，矩阵 B 是 2 行 3 列，替换位置为第 2 行的第 3 列，即：将 A 中第 2 行第 3 列开始的、行数为 2 列数为 3 的子矩阵替换为 B。但该子矩阵超出了 A 的范围，所以只实现了部分替换。

【评分标准】

该题要求输出替换后的矩阵，共有 5 个测试点，提交程序文件名为 example2.c 或 example2.cpp。

解题思路：

很简单,略。

代码：

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

void main ()
{
    int x0,y0,i,j,x1,y1,x2,y2;

    printf("输入矩阵行列数[行,列]:");
    scanf("%d%d",&x0,&y0);
    int** array0=(int**)malloc(sizeof(int*)*x0);
    for(i=0;i<x0;i++)
        array0[i]=(int*)malloc(sizeof(int)*y0);
    for(i=0;i<x0;i++)
        for(j=0;j<y0;j++)
            scanf("%d",&array0[i][j]);

    printf("输入矩阵行列数[行,列]:");
    scanf("%d%d",&x1,&y1);
    int** array1=(int**)malloc(sizeof(int*)*x1);
    for(i=0;i<x1;i++)
        array1[i]=(int*)malloc(sizeof(int)*y1);
    for(i=0;i<x1;i++)
        for(j=0;j<y1;j++)
            scanf("%d",&array1[i][j]);

    printf("输入替换坐标[行,列]:");
    scanf("%d%d",&x2,&y2);
    for(i=0;i<x0 && i<x1;i++)
        for(j=0;j<y0 && j<y1;j++)
            array0[i+x2-1][j+y2-1]=array1[i][j];
    for(i=0;i<x0;i++)
    {
        for(j=0;j<y0;j++)
            printf("%d ",array0[i][j]);
        putchar('\n');
    }
}
```

3. 【问题描述】

从键盘输入包含扩展符'-'的字符串，将其扩展为等价的完整字符，例如将 a-d 扩展为 abcd，

并输出扩展后的字符串。

要求：只处理[a-z]、[A-Z]、[0-9]范围内的字符扩展，即只有当扩展符前后的字符同时是小写字母、大写字母或数字时才进行扩展，其它情况不进行扩展，原样输出。例如：a-R、D-e、0-b、4-B 等字符串都不进行扩展。

【输入形式】

从键盘输入包含扩展符的字符串

【输出形式】

输出扩展后的字符串

【输入样例 1】

ADEa-g-m02

【输出样例 1】

ADEabcdefghijklm02

【输入样例 2】

cdeT-bcd

【输出样例 2】

cdeT-bcd

【样例说明】

将样例 1 的输入 ADEa-g-m02 扩展为：ADEabcdefghijklm02；样例 2 的输入 cdeT-bcd 中，扩展符前的字符为大写字母，扩展符后的字符为小写字母，不在同一范围内，所以不进行扩展。

【评分标准】

结果完全正确得 15 分，共 5 个测试点，每个测试点 3 分，提交程序文件 expand.c 或 expand.cpp。

解题思路：

很简单，略。

代码：

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

int getindex(char ch)
{
    int index=-1;
    if(ch>='a' && ch<='z')
        index=0;
    else if(ch>='0' && ch<='9')
        index=1;
    else if(ch>='A' && ch<='Z')
        index=2;
    return index;
}
```

```c
int main()
{
    char str[300],ch;
    ch=getchar();
    int i=0,j;
    while(ch!='\n')
    {
        str[i]=ch;
        ch=getchar();
        i++;
    }
    str[i]='\0';
    for(i=0;str[i]!='\0';i++)
        if(str[i]=='-')
        {
            if(getindex(str[i-1])==getindex(str[i+1]) &&
getindex(str[i-1])!=-1)
            {
                ch=str[i-1]+1;
                while(ch<str[i+1])
                {
                    putchar(ch);
                    ch++;
                }
            }
        }
        else
            putchar(str[i]);
    return 0;
}
```

2010 年上机真题（回忆版）

1. 利用泰勒公式求 cos(x)=1-x^2/2!+x^4/4!-……公式已给，重要的就是注意细节（比如阶乘的存储最好用 double 类型），二级 C 语言的难度。
2. 归并两个有序字符串，要求输出不能有重复字符（数据结构上做过 N 遍的 Merge 函数）

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

void main()
{
    char str0[200],str1[200],str[400],ch;
    ch=getchar();
    int i=0,j,k;
    while(ch!='\n')
    {
        str0[i]=ch;
        ch=getchar();
        i++;
    }
    str0[i]='\0';
    i=0;
    ch=getchar();
    while(ch!='\n')
    {
        str1[i]=ch;
        ch=getchar();
        i++;
    }
    str1[i]='\0';
    i=j=k=0;
    while(str0[i]!='\0' && str1[j]!='\0')
    {
        if(str0[i]<str1[j])
            if(str0[i]==str[k-1])
                i++;
            else
            {
                str[k]=str0[i];
```

```
            k++;
            i++;
        }

    else if(str0[i]>str1[j])
        if(str1[j]==str[k-1])
            j++;
        else
        {
            str[k]=str1[j];
            j++;
            k++;
        }
    else
        if(str0[i]==str[k-1])
        {
            i++;
            j++;
        }
        else
        {
            str[k]=str0[i];
            i++;
            j++;
            k++;
        }
}
if(str0[i]=='\0'){
    while(str1[j]!='\0'){
        if(str1[j]!=str[k-1])
            str[k++]=str1[j++];
        else
            j++;
    }

}
else{
    while(str0[i]!='\0')
    if(str0[i]!=str[k-1])
        str[k++]=str0[i++];
    else
        i++;
}
```

```
    int length = strlen(str);
    for(i=0;i<length;i++){
        printf("%c",str[i]);
    }
}
```

3. 两个整数数组（无序，可有重复元素），判断两个整数数组是否完全相同（重复元素的话，重复次数也要相同）

代码:

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

void main()
{
    int n,i,ii,j,k,l;
    scanf("%d",&n);
    int* array0=(int*)malloc(sizeof(int)*n);
    for(i=0;i<n;i++)
        scanf("%d",array0+i);
    int* array1=(int*)malloc(sizeof(int)*n);
    for(i=0;i<n;i++)
        scanf("%d",array1+i);
    for(i=0;i<n;i++)
    {
        j=l=0;
        for(ii=0;ii<n;ii++)
            if(array0[i]==array0[ii])
                j++;
        for(k=0;k<n;k++)
            if(array1[k]==array0[i])
                l++;
        if(j!=l)
        {
            printf("not equal!\n");
            exit(0);
        }
    }
    printf("equal! \n");
}
```

09 年上机题

1、【问题描述】

立方根的逼近迭代方程是 y(n+1) = y(n)*2/3 + x/(3*y(n)*y(n)),其中 y0=x.求给定的 x 经过 n 次迭代后立方根的值。

输入：

输入有多组数据。

每组一行，输入 x n。

输出：

迭代 n 次后的立方根，double 精度,保留小数点后面六位。

样例输入：

3000000 28

样例输出：

144.224957

代码：

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

int buaa_func_091()
{
        double x,y;
    int i,n;
    while(scanf("%lf%d",&x,&n) != EOF)
    {
        y=x;
        for(i=1;i<=n;i++)
        y=y*2/3+x/(3*y*y);
        printf("%.6f\n",y);
    }
    return 0;
}
```

2、数组排序

输入一个数组的值,求出各个值从小到大排序后的次序。

输入:输入的第一个数为数组的长度,后面的数为数组中的值,以空格分割

输出:各输入的值按从小到大排列的次序。

sample

input:
4
-3 75 12 -3
output:
1 3 2 1

代码：

```
#include<stdio.h>
# define N 10000
int del(int a[],int n);
int bubblesort(int a[],int n);
int locate(int a[],int b,int n);
int main()
{
	int a[N],b[N],i,j,n,num,z;
	while(scanf("%d",&n)!=EOF)
	{
		for(i=0;i<n;i++)
		{
			scanf("%d",&a[i]);
			b[i]=a[i];
		}
		bubblesort(a,n);
		num=del(a,n);
	for(z=0;z<n-1;z++)
			printf("%d ",locate(a,b[z],num));
		printf("%d\n",locate(a,b[n-1],num));
	}
	return 0;
}
int del(int a[],int n)
{
	int i,j,k;
	for(i=0,j=i+1;j<n;j++)
	{
		if(a[i]!=a[j])
		{
			if((k=j-i-1)!=0)
			while(j<n)
				{
					a[j-k]=a[j];
					j++;
				}
				i++;
```

```
                                        j=i;
                                        n=n-k;
                    }
            }
            if(a[n-2]==a[n-1])
                                n--;
            return n;
}
int bubblesort(int a[],int n)
{
    int i,j,temp;
    for(i=0;i<n;i++)
            for(j=i;j<n;j++)
                    if(a[i]>a[j])
                    {
                temp=a[i];
                                a[i]=a[j];
                                a[j]=temp;
                    }
                return 0;
}
int locate(int a[],int b,int n)
{
            int i,j,k;
     for(i=0;i<n;i++)
                    if(a[i]==b)
                                return i+1;
}
```

3、字符串的查找删除

给定文件 filein.txt 按要求输出 fileout.txt。

输入：无空格的字符串

输出：将 filein.txt 删除输入的字符串(不区分大小写),输出至 fileout.txt

sample

输入:in

输出:将 filein.txt 中的 In、IN、iN、in 删除,每行中的空格全部提前至行首,输出至 fileout.txt

filein.txt 中的值为:

```
#include <stdio.h>
int main()
{
printf(" Hi ");
}
```

输出的 fileout.txt 为

```
#clude<stdio.h>
```

```
tma()
{
prtf("Hi");
}
```

代码：

```
#include <stdio.h>
#include <string.h>
#include <ctype.h>
int main(){
	char a[100];
	int i,n,j;
	char c;
	scanf("%s",a);
	n=strlen(a);
	i=0;
	c=getchar();
	while((c=getchar())!=EOF){
		if(tolower(c)==tolower(a[i])){
			i++;
			if(i>=n)
				i=0;
		}
		else{
			if(i==0){
				if(c!=' ')
					putchar(c);
			}
			else{
				for(j=0;j<i;j++)
					putchar(a[j]);
				i=0;
				if(c!=' ')
					putchar(c);
			}
		}
	}
}
```

08 年上机题

1. 素数
输入一个整数，要求输出所有从 1 到这个整数之间个位为 1 的素数，如果没有则输出-1（30 分）

```
#include <stdio.h>
int main()
{
	int n,i,j,flag;

	while((scanf("%d",&n))!=EOF)
	{
		flag=0;
		for(i=2;i<n;i++)
		{
			for(j=2;j<i;j++)
			{
				if(i%j==0)
					break;
			}
			if(j==i&&(i-1)%10==0)
			{
				if(!flag)
				{
					printf("%d",i);
				flag=1;
				}
				else
					printf(" %d",i);

			}
		}
				if(flag)
				printf("\n");
		else if(!flag)
			printf("-1\n");
	}
    return 0;

}
```

2. 旋转矩阵
任意输入两个 9 阶以下矩阵，要求判断第二个是否是第一个的旋转矩阵，如果是，输出旋转角度（0、90、180、270），如果不是，输出-1。

要求先输入矩阵阶数，然后输入两个矩阵，每行两个数之间可以用任意个空格分隔。行之间用回车分隔，两个矩阵间用任意的回车分隔。（60 分）

```
#include<stdio.h>

int judge(int a[9][9],int b[9][9],int n){

    int i,j,count=0;;

if(a[0][0]==b[0][0]&&a[0][n-1]==b[0][n-1]&&a[n-1][0]==b[n-1][0]&&a[n-1][n-1]==b[n-1][n-1]){
        for(i=0;i<n;i++){
            for(j=0;j<n;j++){
                if(a[i][j]==b[i][j]) count++;
            }
        }
        if(count==n*n) return 0;
        else return -1;
    }

    else if(a[0][0]==b[0][n-1]&&a[0][n-1]==b[n-1][n-1]&&a[n-1][0]==b[0][0]&&a[n-1][n-1]==b[n-1][0]){
        for(i=0;i<n;i++){
            for(j=0;j<n;j++){
                if(a[i][j]==b[j][n-i-1]) count++;
            }
        }
        if(count==n*n) return 90;
        else return -1;
    }

    else if(a[0][0]==b[n-1][n-1]&&a[0][n-1]==b[n-1][0]&&a[n-1][0]==b[0][n-1]&&a[n-1][n-1]==b[0][0]){
        for(i=0;i<n;i++){
            for(j=0;j<n;j++){
                if(a[i][j]==b[n-i-1][n-j-1]) count++;
            }
        }
        if(count==n*n) return 180;
        else return -1;
    }

    else
```

```
if(a[0][0]==b[n-1][0]&&a[0][n-1]==b[0][0]&&a[n-1][0]==b[n-1][n-1]&&a[n-1][n
-1]==b[0][n-1]){
            for(i=0;i<n;i++){
                for(j=0;j<n;j++){
                    if(a[i][j]==b[n-j-1][i]) count++;
                }
            }
            if(count==n*n) return 270;
            else return -1;
    }

    else return -1;

    }

int main(){

    int n,i,j,a[9][9],b[9][9];
    while(scanf("%d",&n)!=EOF){

          for(i=0;i<n;i++)
              for(j=0;j<n;j++)
                  scanf("%d",&a[i][j]);

          for(i=0;i<n;i++)
              for(j=0;j<n;j++)
                  scanf("%d",&b[i][j]);
          printf("%d\n",judge(a,b,n));
    }

    return 0;
    }
```

3. 字符串匹配

从 string.in 中读入数据，然后用户输入一个短字符串。要求查找 string.in 中和短字符串的所有匹配，输出行号、匹配字符串到 string.out 文件中。匹配时不区分大小写，并且可以有一个用中括号表示的模式匹配。如“aa[123]bb”，就是说 aa1bb、aa2bb、aa3bb 都算匹配。（60 分）

```
#include <stdio.h>
#include <string.h>
#define MAXN 1000 +10
char a[MAXN][MAXN];
char   s[MAXN];

int cmpch(char a,char b){
```

```
                    if (a>='0'&&a<='9'&&a==b)
                    {return 1;}
                    else if (a>='a'&&a<='z'&& (a==b||(a+'A'-'a')==b) )
                    {return 1;}
                    else if (a>='A'&&a<='Z'&& (a==b||(a+'a'-'A')==b))
                    {return 1;}
                    else return 0;

}

int main(){

        int n,i,j,m,k,mark,len;
        while(scanf("%d",&n)==1){
          for(i=0;i<n;i++){

          scanf("%s",a[i]);
          }
          scanf("%s",s);
          m=strlen(s);

         for(i=0;i<n;i++){
                    len=strlen(a[i]);
                for(j=0,k=0;j<len,k<m;j++,k++){

                    if(cmpch(a[i][j],s[k]))    continue;
                    else{
                              if(s[k]!='[')    goto out;
                              else{
                                        mark=0;
                                        k=k+1;
                                        while(s[k]!=']'){

                                                  if(cmpch(a[i][j],s[k])) mark=1;
                                                  k++;
                                        }
                                          if(mark==0) goto out;

                              }
                    }



                    }
```

```
out:
                                if((j==len)&&(k==m))    printf("%d %s\n",i+1,a[i]);
            }
            /* for(i=0;i<n;i++){

              printf("%s\n",a[i]);
              }
              printf("%s\n",s);*/


          }
return 0;
}
```

07 年真题

1.【问题描述】
从输入的字符串中，统计空格，回车，TAB 出现的次数
代码：

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

```c
#include <stdlib.h>

int main(){
	int space_count,enter_count,tab_count;
	space_count = enter_count=tab_count=0;

	char ch,buf[1024];
	ch=getchar();
	int i=0,j;
	while(ch!='\\')		//以"\"作为输入标志
	{
		buf[i]=ch;
		ch=getchar();
		i++;
	}
	buf[i]='\0';

	int length = strlen(buf);
	for(i=0;i<length;i++){
		ch = buf[i];
		if(ch == ' '){
			space_count++;
		}
		else if(ch == '\n'){
			enter_count++;
		}
		else if(ch == '\t'){
			tab_count++;
		}
	}

	printf("%d,%d,%d",space_count,enter_count,tab_count);
	return 0;
}
```

2. 【问题描述】
将两个升序字符串合并成一个升序字符串，相同的字母，出现一次。
解答：归并排序的 merge 过程，参见 10 年上机 第 2 题
3. 【问题描述】
两个多项式相加。（计算机二级水平）

06 年上机题

1.【问题描述】
排序与数组内容结合
第一题　写一个函数 set_same(int a[],int len1,int b[],int len2) 判断数组 a 和 b 所含元素是否都相同 就是说
如果把 a 和 b 所含元素按成两个集合　判断是两个集合是否相等 既然是集合 那就不考虑重复元素和顺序了
代码

```
#include <stdio.h>
#include <stdlib.h>
#include "fsh08.h"

int main(){
	int n,m,i,j;
		scanf("%d",&n);
		int* array0=(int*)malloc(sizeof(int)*n);
		for(i=0;i<n;i++)
			scanf("%d",array0+i);
		scanf("%d",&m);
		int* array1=(int*)malloc(sizeof(int)*m);
		for(i=0;i<m;i++)
			scanf("%d",array1+i);

		int is_equal = 1;
		for(i=0;i<n;i++)
		{
			for(j=0;j<m;j++){
				if(array0[i]==array1[j]){
					break;
				}
			}
			if(j == m){    // 找到相同的元素
				is_equal = 0;
				break;
			}
		}
		if(is_equal == 0){
			printf("not equal! \n");
		}
		else{
			for(i=0;i<m;i++)
			{
```

```
                for(j=0;j<n;j++){
                        if(array1[i]==array0[j]){
                                break;
                        }
                }
                if(j == n){    // 找到相同的元素
                        is_equal = 0;
                        break;
                }
        }
        if(is_equal == 0){
                printf("not equal! \n");
        }
        else{
                printf("equal! \n");
        }

    }
    return 0;
}
```

2.【问题描述】

输入几个学生的姓名和成绩　然后按照规定的格式 按照成绩的高低 顺序输出（数据结构中的排序算法，冒泡排序，插入排序，快速排序等等，请各位研友自己敲代码，熟练掌握）