程序设计Moodle上的题目汇总（含解答）

1.1

第一版学籍管理系统，只输出接收用户请求的选项界面。

程序运行结果如下：

1.Input

2.Output

3.Order

4.Quit

（注意：使用英文标点，首字母大写，行与行之间**不插入空行**）

|  |
| --- |
| #include <stdio.h>    int main()  {  printf("1.Input\n");  printf("2.Output\n");  printf("3.Order\n");  printf("4.Quit\n");  return 0;  }  1.2  输入两个学生的成绩，按照由小到大顺序输出。输入输出格式如下：  输入：85,70  输出格式：70,85  #include<stdio.h>  int main()  {  int num1,num2;  scanf("%d,%d",&num1,&num2);  printf("%d",num1>num2?num1:num2);  }  1.3 |

输入五个字符，按照逆序输出

输入：o,l,l,e,h 输出：hello

#include<stdio.h>

**int** main()

{

**char** c[5];

**scanf**("%c,%c,%c,%c,%c",&c[0],&c[1],&c[2],&c[3],&c[4]);

**printf**("%c%c%c%c%c",c[4],c[3],c[2],c[1],c[0]);

**return** 0;

}

1.4

输入一个年份，判断其是不是闰年，如果是闰年，则输出“yes”；如果不是，则输出“no”。  
  
输入输出格式如下：  
  
如果输入：2016  
  
输出：yes  
  
如果输入：2015  
  
输出：no

#include<stdio.h>

int main()

{

int year;

scanf("%d",&year);

printf("%s",((year%4==0)&&(year%100!=0)||(year%400==0))?"yes":"no");

return 0;

}

1.5

输入一个字符，输出对应的ASCII码。例如：  
输入：a  
输出：97

#include<stdio.h>

**int** main()

{

**char** a;

**scanf**("%c",&a);

**printf**("%d",a);

}

1.6

输入一个日期，计算其是这一年中的第几天。  
输入输出格式如下：  
如果输入：20160101  
输出：1  
如果输入：8880803  
输出：216

#include<stdio.h>

int main()

{

int i,x,year,mouth,day,flag=0,sum=0;

scanf("%d",&x);

int array[13][2]={{0,0},{31,31},{28,29},{31,31},{30,30},{31,31},{30,30},{31,31},{31,31},{30,30},{31,31},{30,30},{31,31}};

//初始化,得到年月日

year=x/10000;

mouth=x/100%100;

day=x%100;

flag=((year%4==0)&&(year%100!=0)||(year%400==0));

for (i=1;i<mouth;i++)

sum+=array[mouth][flag];

sum+=day;

printf("%d",sum);

return 0;

}

1.7

输入三角形的直角边（输入数据用float类型），分别以%f和%g输出斜边长度。输入输出格式如下：  
输入：  
3 4.1  
输出：  
c=5.080354  
c=5.08035  
注意，输入两个实数之间为一个空格，输出中“c=”是固定的。

#include<stdio.h>

#include<math.h>

**int** main()

{

**float** a,b,c;

**scanf**("%f %f",&a,&b);

    c=**sqrt**(a\*a+b\*b);

**printf**("c=%lf\nc=%g",c,c);

}

2.1

输入五个字符，将其中的字母用原来字母后的第五个字母来代替（比如'a'转换为'f'，'A'转换为'F','u'转换为‘z，以此类推），其他字符不变。输出转换后的结果。例如：

输入：a123v

输出：f123a

#include<stdio.h>

int main()

{

char c[5];

int i;

scanf("%s",c);

for (i=0;i<5;i++)

{

if (c[i]<='z' && c[i]>='a')

c[i]=((c[i]-'a'+5)%26)+'a';

if (c[i]<='Z' && c[i]>='A')

c[i]=((c[i]-'A'+5)%26)+'A';

}

printf("%s",c);

return 0;

}

2.2

输入一个三位数，判断这个数是否是水仙花数，并输出判断结果。

水仙花数是指一个 n 位数（n≥3 ），它的每个位上的数字的 n 次幂之和等于它本身（例如：1^3 + 5^3+ 3^3 = 153）。

例如：

输入：153

输出： yes

输入：111

输出：no

#include<stdio.h>

int main()

{

int x,num\_1,num\_2,num\_3;

scanf("%d",&x);

num\_1=x%10;

num\_2=x/10%10;

num\_3=x/100;

if (x==num\_1\*num\_1\*num\_1+num\_2\*num\_2\*num\_2+num\_3\*num\_3\*num\_3) printf("yes");

else printf("no");

return 0;

}

2.3

输入5个字母，分别统计大写字母和小写字母的个数，要求输入字符用逗号分隔，输入格式如下：

a,A,r,g,R

输出格式如下：

大写字母个数是2

小写字母个数是3

（输入用,分隔，输出之间无空行）

#include<stdio.h>

int main()

{

char c[5];

int i,count\_1=0,count\_2=0;

scanf("%s",c);

for (i=0;i<5;i++)

{

if (c[i]<='z'&&c[i]>='a')

count\_1++;

if (c[i]<='Z'&&c[i]>='A')

count\_2++;

}

printf("大写字母个数是%d\n",count\_2);

printf("小写字母个数是%d",count\_1);

return 0;

}

2.4

输入一个不大于五位的正整数，输出这个数的逆序。格式如下：  
输入：36275  
输出：57263

#include<stdio.h>

**int** main()

{

**int** x,num;

**scanf**("%d",&x);

**while** (x!=0)

    {

        num=x%10;

        x=x/10;

**printf**("%d",num);

    }

**return** 0;

}

2.5

输入一个百分制成绩，要求输出对应的成绩等级。95分以上为A+，90-94为A，80-89为B，70-79为C，60-69为D，60分以下为E.(要求使用switch语句)  
格式如下：  
输入：85  
输出：B

#include<stdio.h>

int main()

{

int x;

scanf("%d",&x);

if (x>=90)S

x=(x+5)/10;

else x=x/10;

switch (x)

{

case 10:printf("A+");break;

case 9:printf("A");break;

case 8:printf("B");break;

case 7:printf("C");break;

case 6:printf("D");break;

default:printf("E");break;

}

return 0;

}

2.6

【题目描述】  
设某银行存款利率如下：  
1年期定期年利率1.5%；  
2年期定期年利率2.1%；  
3年期定期年利率2.75%；  
5年期定期年利率3.0%；  
活期年利率0.35%。  
  
编写程序，输入本金，计算存款5年的下列存款方式到期各获得本息和是多少。  
（1）定期5年  
（2）先存两年，到期后本息再存3年；  
（3）先存3年，到期后本息再存2年；  
（4）存1年，到期本息再存1年，连续存5年；  
（5）活期，每季度结算一次，结算利息算入下期本金。  
数据类型使用double，输入格式为"%lf",输出格式为"%f"，输出数据间一个空格。  
  
【输入输出样例】  
输入：  
1000  
输出:  
1150.000000 1127.965000 1127.965000 1077.284004 1017.646235

#include<stdio.h>

#include<math.h>

int main()

{

int i;

double x,num\_1,num\_2,num\_3,num\_4,num\_5;

scanf("%lf",&x);

//第一种

num\_1=x\*(1+5\*0.03);

//第二种

num\_2=x\*(1+0.021\*2)\*(1+0.0275\*3);

//第三种

num\_3=x\*(1+0.0275\*3)\*(1+0.021\*2);

//4

num\_4=x\*(1+0.015)\*(1+0.015)\*(1+0.015)\*(1+0.015)\*(1+0.015);

//5

for (i=1;i<=20;i++)

x=x\*(1+0.0035\*0.25);

num\_5=x;

printf("%f %f %f %f %f",num\_1,num\_2,num\_3,num\_4,num\_5);

return 0;

}

3.1

输出1!+2!+3!+...+10!的值 。

输出格式如下：

1!+2!+3!+...+10!=求得的值

#include<stdio.h>

int main()

{

int f(int n);

int n=1,sum=0;

for (;n<=10;n++)

{

sum+=f(n);

}

printf("1!+2!+3!+...+10!=%d",sum);

return 0;

}

int f(int n)

{

int i=1,sum=1;

for (;i<=n;i++)

sum\*=i;

return sum;

}

3.2

输出100以内的所有素数，每行输出五个数，数字之间用一个空格分隔。  
输出如下：  
2 3 5 7 11  
13 17 19 23 29  
31 37 41 43 47  
53 59 61 67 71  
73 79 83 89 97

#include<stdio.h>

#include<math.h>

int main()

{

int f(int n);

int i,k=0;

for (i=2;i<=100;i++)

{

if (f(i))

{

printf("%d",i);

k++;

if (k==5)

{

k=0;

printf("\n");

}

else

{

printf(" ");

}

}

}

return 0;

}

int f(int n)

{

int i;

for (i=2;i<=(int)sqrt(n);i++)

if (n%i==0)

return 0;

return 1;

}

6.3

一个数如果等于其因子之和，则称为“完数”。例如，6的因子是1,2,3，而6=1+2+3，则6是完数。输出1000以内的所有完数，数字之间用一个空格隔开。

#include<stdio.h>

#include<math.h>

int main()

{

int i;

for (i=1;i<=1000;i++)

if(f(i))

printf("%d ",i);

return 0;

}

//判断是否为完数

int f(int n)

{

int i,sum=0;

for (i=1;i<n;i++)

if(n%i==0)

sum+=i;

if (n==sum)

return 1;

else

return 0;

}

3.4

输入一个10以内的整数，输出对应的图案。格式如下：  
输入：4  
输出：  
1  
2 3  
4 5 6  
7 8 9 10  
（输出的数字之间一个空格）

#include<stdio.h>

#include<math.h>

int main()

{

int f(int n);

int n,i,j=0,k=1;

scanf("%d",&n);

n=f(n);

for (i=1;i<=n;i++)

{

printf("%d",i);

j++;

if (j==k)

{

printf("\n");

j=0;

k++;

}

else

{

printf(" ");

}

}

printf(" ");

return 0;

}

int f(int n)

{

int i,sum=0;

for (i=1;i<=n;i++)

sum+=i;

return sum;

}

3.5

输入两个20以内的正整数，输出其最大公约数和最小公倍数。测试用例如下：

输入：

12

15

输出：

最大公约数为3

最小公倍数为60

输入

2

6

输出

最大公约数为2

最小公倍数为6

（中间无空行，数字和字符之间无空格）

#include<stdio.h>

#include<math.h>

#define max(x,y) x>y?x:y

#define min(x,y) x<y?x:y

int f(int x,int y);

int g(int x,int y);

int main()

{

int x,y,temp\_1,temp\_2;

scanf("%d%d",&x,&y);

temp\_1=f(x,y);

temp\_2=g(x,y);

printf("最大公约数为%d\n最小公倍数为%d",temp\_1,temp\_2);

return 0;

}

//最大公约数

int f(int x,int y)

{

int i=min(x,y);

for (;i>=1;i--)

if (x%i==0 && y%i==0)

return i;

}

//最小公倍数

int g(int x,int y)

{

int i=max(x,y);

for (;1;i++)

if(i%x==0&&i%y==0)

return i;

}

3.6

有一个分数序列，2/1,3/2,5/3,8/5,13/8,21/13...  
，序列中的值为分数对应浮点数向下取整，输入一个20以内的整数n，计算这个数列的前n项之和。  
输入示例：  
2  
输出示例：  
sum=3

#include<stdio.h>

#include<math.h>

double fibonacci(int n,double a1,double a2);

int f(int n);

int main()

{

int n;

scanf("%d",&n);

printf("sum=%d",f(n));

return 0;

}

int f(int n)

{

double j;

int k,i,result=0;

for (k=1;k<=n;k++)

{

i=fibonacci(k,2,3)/fibonacci(k,1,2);

result+=i;

}

return result;

}

double fibonacci(int n,double a1,double a2)

{

int i;

double a3;

if (n==1) return a1;

if (n==2) return a2;

for (i=0;i<n-2;i++)

{

a3=a1+a2;

a1=a2;

a2=a3;

}

return a3;

}

3.7

一个球从100m高度自由落下，每次落地反弹回原高度一半，再落地，再反弹。求它第n次（n<=10）落地共经过多少米，第n次反弹多高。（保留6位小数）  
输入示例：  
10  
输出示例：  
第10次落地时共经过299.609375米  
第10次反弹0.097656米

#include<stdio.h>

#include<math.h>

double f(int n);

double g(int n);

int main()

{

int n;

double sum,temp\_1,temp\_2;

scanf("%d",&n);

temp\_1=f(n);

temp\_2=g(n);

printf("第%d次落地时共经过%.6lf米\n第%d次反弹%.6lf米",n,temp\_1,n,temp\_2);

return 0;

}

double f(int n)

{

int i;

double sum=100,h=100;

for (i=1;i<n;i++)

{

sum+=h;

h=h/2;

}

return sum;

}

double g(int n)

{

int i;

double h=100;

for (i=0;i<n;i++)

h=h/2;

return h;

}

3.8

利用公式 π/4=1-1/3+1/5-1/7+…  
求 π 的近似值，舍去绝对值小于eps的通项。eps由用户输入。数据类型使用双精度。  
输入：双精度实数，输出pai的近似值，保留8位小数。  
注意，级数求和之后再乘以4.  
【输入输出样例】  
输入：  
1e-4  
输出:  
3.14139265

#include<stdio.h>

#include<math.h> //为了使用里面的fabs函数

int main()

{

int sign=1;

double pi=0,n=1,term=1,eps;

scanf("%lf",&eps);

while (fabs(term)>=eps)

{

pi+=term;

n+=2;

sign=-sign;

term=sign/n;

}

pi=4\*pi;

printf("%.8lf",pi);

return 0;

}

3.9

编写程序，输出n行n列的下列形式的左下三角九九乘法表，n由用户输入。下表是n=4的输出。  
4  
1\*1=1  
2\*1=2 2\*2=4  
3\*1=3 3\*2=6 3\*3=9  
4\*1=4 4\*2=8 4\*3=12 4\*4=16  
  
输入：1-9的正整数；  
输出：n行n列左下三角九九乘法表，一行中各列数据间有一个空格，末尾无空格。  
注意不能使用\b消末尾的空格。  
  
提示：输出末尾无空格的10个数：  
int i=1,n=10;  
i=1;  
printf("%d",i);  
for(i=2;i<=n;i++)  
{  
printf(" %d",i);  
}  
printf("\n");

#include<stdio.h>

#include<math.h>

int main()

{

int f(int n);

int n,i,j=0,k=1,m=1;

scanf("%d",&n);

n=f(n);

for (i=1;i<=n;i++)

{

printf("%d\*%d=%d",k,j+1,k\*(j+1));

j++;

if (j==k)

{

printf("\n");

j=0;

k++;

}

else

{

printf(" ");

}

}

printf(" ");

return 0;

}

int f(int n)

{

int i,sum=0;

for (i=1;i<=n;i++)

sum+=i;

return sum;

}

4.1

打印出以下的杨辉三角形 ，要求用户输入行数。

最大为20行

输入示例：

6

输出示例：

1

1 1

1 2 1

1 3 3 1

1 4 6 4 1

1 5 10 10 5 1

（注：输出数据之间用一个空格分隔，每行最后没有空格，行之间不插入空行）

#include<stdio.h>

#include<math.h>

int C(int n,int m);

int main()

{

int x,i,j;

scanf("%d",&x);

for (i=0;i<=x-1;i=i+1)

for (j=0;j<=i;j=j+1)

printf("%d%s",C(j,i),(j!=i)?" ":"\n");

return 0;

}

//杨辉三角函数

int C(int n,int m)

{

int i,k=0,temp\_1=1,temp\_2=1;

if(n==0||m==n)

return 1;

for (i=m;k<n;i--,k++)

temp\_1\*=i;

for (i=1;i<=n;i++)

temp\_2\*=i;

return temp\_1/temp\_2;

}

4.2

输入最多10个字符，以回车符结束输入，输出其中的大写字母。  
输入示例：abcDeF  
输出示例：DF

#include<stdio.h>

int main()

{

char a[10];

int i=0;

scanf("%s",a);

while (a[i]!='\0')

{

if (a[i]<=90&&a[i]>=65)

printf("%c",a[i]);

i++;

}

return 0;

}

4.3

矩阵元素从键盘输入，数值之间用一个空格隔开。输出主对角线元素之和。比如:

输入：

1 2 3 4 5 6 7 8 9

输出：

15

#include<stdio.h>

#include<ctype.h>

#include<math.h>

#define N 100

int main()

{

int array[N];

char c;

int i=0,n=0,sum=0,k=0;

while ((c=getchar())!='\n')

{

if (isdigit(c))

{

ungetc(c,stdin);

scanf("%d",&array[i++]);

n++;

}

}

k=sqrt(n);

for (i=0;i<n;i+=k+1)

{

sum+=array[i];

}

printf("%d",sum);

return 0;

}

4.4

输入n和n个整数，使用选择法排序，输出从大到小的序列。元素个数不超过100.  
输入：正整数n和n个整数。  
输出：从大到小的有序序列，用一个空格隔开，末尾无空格。  
【输入输出样例】  
输入：  
4  
1 2 3 4  
输出:  
4 3 2 1  
说明：  
选择排序是给每个位置选择当前元素最小的，比如给第一个位置选择最小的，在剩余元素里面给第二个元素选择第二小的，依次类推，直到第n-1个元素。

#include<stdio.h>

#include<math.h>

int main()

{

int n,i,j,max;

scanf("%d",&n);

int a[n];

for (i=0;i<n;i++)

scanf("%d",&a[i]);

for (i=0;i<n-1;i++)

{

max=a[i];

for (j=i+1;j<n;j++)

{

if (a[j]>a[i])

{

max=a[j];

a[j]=a[i];

a[i]=max;

}

}

}

for (i=0;i<n;i++)

printf("%d%s",a[i],(i!=n-1)?" ":"");

return 0;

}

4.5

【题目描述】  
用筛选法求n以内（含n，n<=1000）的素数，并逆序输出。输入 n，逆序输出n以内的素数，每10个一行，用空格隔开，行末无空格。  
【输入输出样例】  
输入：  
100  
输出：  
97 89 83 79 73 71 67 61 59 53  
47 43 41 37 31 29 23 19 17 13  
11 7 5 3 2  
提示：筛选法又称筛法，具体做法是：先把n个自然数按次序排列起来。1不是质数，也不是合数，要划去。第二个数2是质数留下来，而把2后面所有能被2整除的数都划去。2后面第一个没划去的数n划去。3后面第一个没划去的数是5，把5留下，再把5后面所有能被5整除的数都划去。这样一直做下去，就会把不超过N的全部合数都筛掉，留下的就是不超过N的全部质数。因为希腊人是把数写在涂腊的板上，每要划去一个数，就在上面记以小点，寻求质数的工作完毕后，这许多小点就像一个筛子，所以就把埃拉托斯特尼的方法叫做“埃拉托斯特尼筛”，简称“筛法”。

#include<stdio.h>

#include<math.h>

int main()

{

int n;

scanf("%d",&n);

int i,k=1,array[n];

//初始化数组

array[0]=0;

for (i=2;i<=n;i++)

array[i-1]=i;

//筛选法找素数

int flag\_1=array[1],flag\_3=array[1];

while (1)

{ //p=2

for (i=k+1;i<n;i++)

if (array[i]!=0 && array[i]%flag\_1==0)

array[i]=0;

//改变flag\_1

for (i=k+1;i<n;i++)

{

if (array[i]!=0)

{

flag\_1=array[i];

k=i;

break;

}

}

//找完了就终止

if (flag\_1==flag\_3)

break;

else

flag\_3=flag\_1;

}

//倒序输出

int flag\_2=0;

for (i=n-1;i>=0;i--)

if (array[i]!=0)

printf("%d%s",array[i],(flag\_2=++flag\_2%10)?" ":"\n");

return 0;

}

4.6

输入一个n阶（奇数,n<=15）魔方阵是指一个方阵，它的每一行、每一列和对角线之和均相等。  
例如：  
8 1 6  
3 5 7  
4 9 2  
输入示例：  
3  
输出示例：  
8 1 6  
3 5 7  
4 9 2  
提示：  
魔方阵的构造方法是：  
①将1放在第1行的中间位置。  
②从2开始直到n\*n，放在前一个数的前一行、后一列的位置。即若k-1放在i行、j列位置，则k放在i-1行、j+1列位置。行号、列号看作首尾相接的，即第1行的前一行是第n行,第n列的后一列是第1列。  
③若放置k时该位置已有数据或上一个数在第1行、第n列位置，则将k放在k-1的下一行同列的位置。  
注：每行末尾不加空格，行间无空行

#include<stdio.h>

#include<math.h>

#define f(i,n) (i==0)?n-1:i-1

#define g(i,n) (j==n-1)?0:j+1

int main()

{

int n;

scanf("%d",&n);

int array[n][n];

//初始化数组

int q1,q2;

for (q1=0;q1<n;q1++)

for (q2=0;q2<n;q2++)

array[q1][q2]=0;

int temp\_1=n/2;

//确定元素1

array[0][temp\_1]=1;

//确定其他元素

int temp\_2,i=0,j=temp\_1;

for (temp\_2=2;temp\_2<=n\*n;temp\_2++)

{

if (temp\_2==2)

{

array[f(i,n)][g(j,n)]=temp\_2;

i=f(i,n);j=g(j,n);

continue;

}

else

{

if (array[f(i,n)][g(j,n)]!=0 || (i==0 && j==n-1))

{

array[i+1][j]=temp\_2;

i=i+1;j=j;

}

else

{

array[f(i,n)][g(j,n)]=temp\_2;

i=f(i,n);j=g(j,n);

}

}

}

//输出

for (i=0;i<n;i++)

{

for (j=0;j<n;j++)

printf("%d%s",array[i][j],(j!=n-1)?" ":"");

printf("\n");

}

return 0;

}

4.7

编写一个程序求字符串的有效长度（输入字符串中包含的字符个数不超过20）。输入输出如下：  
输入：hello  
输出：5  
输入：hello world  
输出：11  
注意:  
1）字符串的有效长度不包括结束符；  
2）字符串中允许有一个或多个空格，空格是有效的字符。

#include<stdio.h>

int main()

{

char x[20];

int i=0;

gets(x);

while ( x[i++] != '\0' )

;

printf("%d\n",i-1);

return 0;

}

4.8

对于一个已排好序的字符数组，要求输入一个字符，按照字符顺序插入数组中。

已知数组为char ch[20]="bcfmr"，

输入：a

输出：abcfmr

输入：n

输出：bcfmnr

输入：z

输出：bcfmrz

#include<stdio.h>

int main()

{

char char\_1[20]="bcfmr",char\_2;

int i,j;

scanf("%c",&char\_2);

for (i=0,j=0;i<5;i++)

if( char\_1[i] > char\_2)

break;

for (j=5;j>i;j--)

char\_1[j]=char\_1[j-1];

char\_1[i]=char\_2;

printf("%s",char\_1);

return 0;

}

4.9

输入一串由若干单词组成的字符串（单词前后或单词之间都可以有一个或多个空格，字符串总长度不超过50），统计其中单词的个数。  
输入输出示例如下：  
输入：how are you（单词之间用一个空格分隔）  
输出：3  
输入：how are you（单词之间的空格数不同）  
输出：3  
输入： hello world (第一个单词前面有空格，最后一个单词后面有空格)  
输出：2  
  
提示：可以使用下面的方法输入带空格的字符串  
char array[20]  
scanf("%[^\n]",array);

#include<stdio.h>

int main()

{

char char\_1[50];

int flag=1,i=0,len=0,num=0;

gets(char\_1);

while ( char\_1[i++] != '\0' ) ;

len=i-1;

for (i=0;i<len;i++)

{

if ( char\_1[i] == ' ' )

{

flag=1;

}

else

{

if (flag)

{

num++;

flag=0;

}

}

}

printf("%d",num);

return 0;

}

4.10

编程实现字符串比大小（不能直接使用strcmp函数）。输入字符串1和字符串2，如果两个字符串相同，输出0；如果字符串1大于字符串2，输出1；如果字符串1小于字符串2，输出-1。比如：

输入：

hello

world

则输出：-1

输入：

world

hel

则输出1

输入：

hello

hello

则输出：0

说明：字符串中不包括空格、两个字符串不一定等长

#include<stdio.h>

#include<math.h>

int cmp(char \*p1,char \*p2);

int main()

{

char a1[100],a2[100];

scanf("%s",a1);

scanf("%s",a2);

printf("%d",cmp(a1,a2));

return 0;

}

int cmp(char\* p1,char\* p2)

{

for (;\*p1==\*p2;p1++,p2++)

if (\*p1=='\0')

return 0;

return (\*p1-\*p2)/abs(\*p1-\*p2);

}

4.11

假定程序中已经有三个学生信息了，现在需要删除一个新的学生信息。具体要求同上题：

1）必须用数组来存储学生信息，包括学号，班级，姓名，三门课程的成绩；

2）**程序只需要输入一个待删除学生的姓名**，如果该学生不存在，则输出原有三个学生排序后的信息（**要求按照姓名先后输出**）；

3）如果指定删除的学生存在，则将其余学生的信息排序输出（**要求按照姓名先后输出**）

4）已有的三个学生的信息可以在数组初始化时静态赋值，也可以用语句来进行赋值。要求这三个学生的信息必须是：

1001,11,zhang,99.5,88.5,89.5,277.5

1002,22,li,77.9,56.5,87.5,221.9

1003,11,wang,92.5,99.0,60.5,252.0

程序输入如下：

li

1003,11,wang,92.5,99.0,60.5,252.0

1001,11,zhang,99.5,88.5,89.5,277.5

或者：

zhao

 1002,22,li,77.9,56.5,87.5,221.9

1003,11,wang,92.5,99.0,60.5,252.0

1001,11,zhang,99.5,88.5,89.5,277.5

（注意：下划线标注部分为输入。要求输出的学生信息采用统一格式，即每个学生一行，行与行之间无空行；学生的各项信息之间用逗号分隔，采用英文标点）

#include<stdio.h>

#include<string.h>

int cmp(char\* p1,char\* p2);

struct Person

{

int num\_1;

int num\_2;

char name[5];

float num\_3;

float num\_4;

float num\_5;

float num\_6;

};

int main()

{

struct Person x[3]={{1002,22,"li",77.9,56.5,87.5,221.9},{1003,11,"wang",92.5,99.0,60.5,252.0},{1001,11,"zhang",99.5,88.5,89.5,277.5}};

char name\_1[5];

scanf("%s",&name\_1);

int i;

for (i=0;i<3;i++)

if ( cmp(name\_1,x[i].name) )

printf("%d,%d,%s,%.1f,%.1f,%.1f,%.1f\n",x[i].num\_1,x[i].num\_2,x[i].name,x[i].num\_3,x[i].num\_4,x[i].num\_5,x[i].num\_6);

return 0;

}

int cmp(char\* p1,char\* p2)

{

for (;\*p1==\*p2;p1++,p2++)

if (\*p1=='\0')

return 0;

return (\*p1-\*p2)/abs(\*p1-\*p2);

}

4.12

有n个数（n<20）,已按从大到小顺序存放在一个数组中，输入一个数，要求用折半查找法找出该数是数组中的第几个元素的值（从1开始）。如果不在数组中输出0。  
输入：  
第一行为数组元素的个数n。  
第二行是n个数组元素的值，整数。  
第三行是要查找的值。  
输出：  
查找的值在数组中的位置（从序号1开始），找不到时显示0。  
【输入输出样例】  
输入：  
10  
10 9 8 7 6 5 4 3 2 1  
5  
输出:  
6  
提示：折半查找法（二分查找）的基本思想是将n个已经排序的元素分成两部分，取a[n/2]与x做比较，如果x=a[n/2],则找到x,算法中止；如果xa[n/2],则只要在数组a的右半部搜索x。

#include <stdio.h>

#include <stdlib.h>

int find(int \*p,int len,int x);

int main()

{

int len;

scanf("%d",&len);

int \*array=(int \*)malloc(len\*sizeof(int));

int i=0;

for (i=0;i<len;i++)

scanf("%d",array+i);

int x;

scanf("%d",&x);

printf("%d",find(array,len,x));

return 0;

}

int find(int \*p,int len,int x)

{

if (len==1)

if (\*(p+0)==x)

return 0;

int flag1=0,flag2=len,flag;

while ( (flag2-flag1)>0 )

{

flag=(flag2+flag1)/2;

if ( \*(p+flag) == x)

return flag;

if ( \*(p+flag) > x)

flag1=flag;

if ( \*(p+flag) < x)

flag2=flag;

}

return -1;

}

4.13

给定一个 m x n 的矩阵（m和n不超过100），如果这个矩阵是托普利茨矩阵，输出"true"；否则，输出"false" 。如果一个矩阵上每一条从左上到右下的对角线上的元素都相同，则该矩阵是托普利茨矩阵。  
第一行输入矩阵的行数和列数，第二行输入矩阵元素。  
示例1：  
输入：  
3 4  
1 2 3 4 5 1 2 3 9 5 1 2  
输出：  
true  
示例2：  
输入：  
2 2  
1 2 3 4  
输出：  
false

#include<stdio.h>

#include<math.h>

int main()

{

int m,n;

scanf("%d %d",&m,&n);

int array[m][n];

//定义数组

int i,j;

for (i=0;i<m;i++)

for (j=0;j<n;j++)

scanf("%d",&array[i][j]);

//给数组赋值

int flag=1;

for (i=1;i<m;i++)

{

for (j=1;j<n;j++)

{

if (array[i][j]!=array[i-1][j-1])

{

printf("false");

flag=0;

break;

}

}

if (!flag)

{

break;

}

}

if (flag)

{

printf("true");

}

return 0;

}

4.14

给定一个整数数组 nums（大小不超过15） ，找到一个具有最大和的连续子数组（子数组最少包含一个元素），返回其最大和。  
示例:  
输入:  
9  
-2 1 -3 4 -1 2 1 -5 4  
输出: 6  
解释: 输入数组大小9和数组中各个元素，其中连续子数组 [4,-1,2,1] 的和最大，为 6。

#include<stdio.h>

#include<math.h>

int main()

{

int n,i,j,k;

scanf("%d",&n);

int array[n];

//定义数组

for (i=0;i<n;i++)

{

scanf("%d",&array[i]);

}

//给数组赋值

int max,num\_1;

max=array[0];

if (n==1)

{

printf("%d",max);

}

else

{

for (i=0;i<n;i++)

{

for (j=n-1;j>=i;j--)

{

num\_1=0;

for (k=i;k<=j;k++)

{

num\_1+=array[k];

}

if(num\_1>max)

{

max=num\_1;

}

}

}

printf("%d",max);

}

return 0;

}

5.1

要求用函数调用来实现完数的判断，在主程序中输入数字，调用函数判断其是否是完数。  
输入：5  
输出：不是完数  
输入：6  
输出：是完数  
  
注：如果一个数恰好等于它的真因子之和，则称该数为“完全数”，比如6的真因子是1,2,3,6=1+2+3，则6是完数

#include<stdio.h>

int main()

{

int x;

scanf("%d",&x);

printf("%s",(f(x))?"是完数":"不是完数");

return 0;

}

int f(int n)

{

int i,sum=0;

for (i=1;i<n;i++)

if(n%i==0)

sum+=i;

return sum==n;

}

5.2

已有3个学生的3门课成绩，分别用函数实现以下功能：  
  
1）计算每个学生的总成绩  
  
2）按照学生总成绩从高到低进行排序  
  
要求：  
  
1）在main函数中分别调用以上函数，按照学生三门课程总成绩从大到小输出学生的相关信息。  
  
2）函数自行定义。三个学生的信息按照如下直接赋值：  
  
1001,11,zhang,99.5,88.5,89.5  
1002,22,li,77.9,56.5,87.5  
1003,11,wang,92.5,99.0,60.5  
  
程序运行结果如下：  
  
1001,11,zhang,99.5,88.5,89.5,277.5  
1003,11,wang,92.5,99.0,60.5,252.0  
1002,22,li,77.9,56.5,87.5,221.9  
（注意：输出学生信息用逗号分隔，换行输出）

#include<stdio.h>

#include<string.h>

struct Person

{

int num\_1;

int num\_2;

char name[5];

float num\_3;

float num\_4;

float num\_5;

float num\_6;

};

int main()

{

float f(float num\_3,float num\_4,float num\_5);

void g(struct Person \*p);

struct Person x[3]={{1001,11,"zhang",99.5,88.5,89.5,f(x[0].num\_3,x[0].num\_4,x[0].num\_5)},{1002,22,"li",77.9,56.5,87.5,f(x[1].num\_3,x[1].num\_4,x[1].num\_5)},{1003,11,"wang",92.5,99.0,60.5,f(x[2].num\_3,x[2].num\_4,x[2].num\_5)}};

g(x);

return 0;

}

float f(float num\_3,float num\_4,float num\_5)

{

float sum;

sum=num\_3+num\_4+num\_5;

return sum;

}

void g(struct Person \*p)

{

int i[3]={0,1,2},temp,j;

if (p[0].num\_6<p[1].num\_6)

{

temp=i[0];

i[0]=i[1];

i[1]=temp;

}

if (p[1].num\_6<p[2].num\_6)

{

temp=i[1];

i[1]=i[2];

i[2]=temp;

}

if (p[0].num\_6<p[1].num\_6)

{

temp=i[0];

i[0]=i[1];

i[1]=temp;

}

for (j=0;j<3;j++)

{

printf("%d,%d,%s,%.1f,%.1f,%.1f,%.1f\n",p[i[j]].num\_1,p[i[j]].num\_2,p[i[j]].name,p[i[j]].num\_3,p[i[j]].num\_4,p[i[j]].num\_5,p[i[j]].num\_6);

}

5.3

编写两个函数，用来计算两个整数的最大公约数和最小公倍数。在主函数中调用并输出计算结果。  
输入：  
12,15  
输出：  
最大公约数是3  
最小公倍数是60

#include<stdio.h>

#define min(x,y) (x>y)?y:x

#define max(x,y) (x<y)?y:x

int f(int x,int y);

int g(int x,int y);

int main()

{

int x,y;

scanf("%d,%d",&x,&y);

printf("最大公约数是%d\n最小公倍数是%d",f(x,y),g(x,y));

return 0;

}

int f(int x,int y)

{

int i=min(x,y);

for (;i>=1;i--)

if (x%i==0&&y%i==0)

return i;

}

int g(int x,int y)

{

int i=max(x,y);

for (;1;i++)

if(i%x==0&&i%y==0)

return i;

}

5.4

【题目描述】  
统计某个给定范围[L, R]内的所有整数中，数字2出现的次数。比如给定范围[2, 22]，数字2在数2中出现了1次，在数12中出现1次，在数20中出现1次，在数21中出现1次，在数22中出现2次，所以数字2在该范围内一共出现了6次。  
【输入输出样例】  
输入两个数作为范围，输出数字2在该范围内的出现次数。  
示例1：  
输入：2 22  
输出 ：6  
示例2：  
输入：3 9  
输出：0

#include<stdio.h>

#include<math.h>

int main()

{

int f(int x);

int i,vary\_1,vary\_2,sum=0;

scanf("%d%d",&vary\_1,&vary\_2);

for (i=vary\_1;i<=vary\_2;i++)

sum+=f(i);

printf("%d",sum);

return 0;

}

int f(int x)

{

int sum=0;

while ( x != 0)

{

if (x%10 == 2)

{

sum++;

}

x=x/10;

}

return sum;

}

5.5

要求用函数调用来实现阶乘的计算，在主程序中输入数字(小于10)，调用函数计算输入的阶乘，并在主程序中输出阶乘。  
示例：  
输入:3  
输出:6

#include<stdio.h>

#include<math.h>

**int** main()

{

**int** f(**int** n);

**int** x;

**scanf**("%d",&x);

**printf**("%d",f(x));

**return** 0;

}

**int** f(**int** n)

{

**int** i=1,sum=1;

**for** (;i<=n;i++)

    {

        sum\*=i;

    }

**return** sum;

}

5.6

编写一个fun函数，其功能为求整数数组p[5]中的最大值和最小值。在主函数中调用fun函数并输出最大值和最小值。  
  
输入：  
  
34  
  
78  
  
23  
  
12  
  
69  
  
输出如下：  
  
max=78  
  
min=12

#include<stdio.h>

#include<math.h>

**int** main ()

{

**int** max(**int** \*array);

**int** min(**int** \*array);

**int** i,array[5];

**for**(i=0;i<5;i++)

    {

**scanf**("%d",&array[i]);

    }

**printf**("max=%d\nmin=%d",max(array),min(array));

**return** 0;

}

**int** max(**int**\* array)

{

**int** i,temp;

    temp=array[0];

**for** (i=1;i<5;i++)

    {

**if** (array[i]>temp)

        {

            temp=array[i];

        }

    }

**return** temp;

}

**int** min(**int**\* array)

{

**int** temp,i;

    temp=array[0];

**for** (i=1;i<5;i++)

    {

**if** (array[i]<temp)

        {

            temp=array[i];

        }

    }

**return** temp;

}

5.7

编写函数，用“冒泡法”对输入的n个整数按从小到大的顺序排列。编写主函数，输入n和n个数组元素，调用函数排序，在主函数中输出。注意，输入、输出可以通过单独的函数实现，在排序函数中不能输出数组元素。  
  
输入：输入n，换行后输入n个整数。（n不超过20）  
输出：从小到大的排列，数字之间有空格，末尾无空格。  
【输入输出样例】  
输入：  
5  
5 4 3 2 1  
输出：  
1 2 3 4 5

|  |
| --- |
| #include <stdio.h>  #include <math.h>  **int** main()  {  **void** sort(**int**\* array,**int** n);  **int** n;  **scanf**("%d",&n);  **int** array[n];  **int** i;  **for** (i=0;i<n;i++)      {  **scanf**("%d",&array[i]);      }      sort(array,n);  **for** (i=0;i<n;i++)      {  **printf**("%d ",array[i]);      }  **return** 0;  }    **void** sort(**int**\* array,**int** n)  {  **int** i,j,temp;  **for** (i=0;i<n;i++)      {  **for** (j=i;j<n;j++)          {  **if** ( array[i]>array[j])              {                  temp=array[i];                  array[i]=array[j];                  array[j]=temp;              }          }      }  } |

5.8

用递归法将一个非负整数n转换成逆序的字符串。例如：输入483，应输出字符串"384"。n的位数不确定，可以是任位数的整数。  
输入：输入非负整数n。（n的位数不超过int上限）  
输出：输出一个字符串。  
  
【输入输出样例】  
输入：  
483  
输出：  
384  
  
注意，本题要实现整数转字符串。投机取巧不得分。  
  
提示1：函数参考原型：  
void myint2str(int a,char s[]，int k);  
将a转换为逆序字符串后放在s中。注意末尾加结束符。k是s中当前可放置字符的下标。  
  
提示2：递归思想 ，如果是1位数，tmp=a+'0'即是字符，存入s[k]，后面再放结束符，结束。如果不是一位数，获取个位数字，转换为字符，放入s[k];注意a=a/10去掉原个位，还是一个整数转字符串的问题，可以递归了。

#include <stdio.h>

#include <math.h>

void myint2str(int a,char \*s,int k);

int main()

{

int a,k=0;

char s[100];

scanf("%d",&a);

myint2str(a,s,k);

printf("%s",s);

return 0;

}

void myint2str(int a,char \*s,int k)

{

char tmp;

if (a>10)

{

s[k]=a%10+'0';

myint2str(a/10,s,k+1);

}

else

{

s[k++]=a+'0';

s[k]='\0';

}

}

5.9

给定一个整数数组 nums 。  
如果一组数字 (i, j)满足 nums[i]==nums[j]，且i小于j,就可以认为这是一组好数对。  
返回好数对的数目。  
编写主函数，输入一个整数n（小于100）和对应的整数数组，调用好数对函数，返回好数对的数目，并在主函数中输出。  
示例：  
输入：  
6  
1 2 3 1 1 3  
输出：  
4  
  
提示：好数对函数参考原型 int numPairs(int n, int nums[])

#include <stdio.h>

#include <stdlib.h>

int main()

{

int i,n;

scanf("%d",&n);

int \*nums=(int \*)malloc(sizeof(int)\*n);

for (i=0;i<n;i++)

scanf("%d",nums+i);

printf("%d",numPairs(n,nums));

return 0;

}

int numPairs(int n, int\* nums)

{

int i,j,sum=0;

for (i=0;i<n;i++)

for (j=i+1;j<n;j++)

if ( nums[i] == nums[j])

sum++;

return sum;

}

5.10

编写一个函数void copystr(char str1[20],char str2[20], int m)，用于实现将str1字符串的前m个字符全部复制到另一个字符串str2中。在主函数中输入字符串和m，调用这个函数，输出复制后的结果。  
运行格式如下  
  
输入:  
hello world  
5  
输出：  
hello  
  
输入:  
hello world  
7  
输出：  
hello w  
  
（注意：字符串中允许有空格，不要直接调用strncpy函数）

#include<stdio.h>

void copystr(char \*array\_1,char \*array\_2,int m);

int main()

{

char array\_1[20],array\_2[20];

scanf("%[^\n]",array\_1);

int m;

scanf("%d",&m);

copystr(array\_1,array\_2,m);

printf("%s",array\_2);

return 0;

}

void copystr(char \*p1,char \*p2,int m)

{

int i=0;

for (;i<m;i++)

\*(p2+i)=\*(p1+i);

\*(p2+m)='\0';

}

5.11

题目描述：  
定义一个函数：求输入的非空字符串是否是回文字符串。“回文串”是一个正读和反读都一样的字符串。  
注意：统计字符串中回文子串的个数需要通过主函数外另外的函数专门实现。  
测试用例：  
输入描述：输入数据的第一行为一个正整数（<=10），表示待判断字符串的数目，之后每行数据为一个字符串,字符串长度（<=10）。  
输出描述：每个输入的字符串对应输出一个整数，代表该字符串中回文子串的个数。  
示例：  
输入：  
3  
a  
abc  
aaa  
输出：  
1  
3  
6  
示例解释：  
"a"有1个回文子串："a"；  
"abc"有3个回文子串: "a", "b", "c"；  
"aaa"有6个回文子串: "a","a","a","aa", "aa", "aaa"

#include <stdio.h>

#include <math.h>

#include <string.h>

void num(char (\*array)[10],int \*array\_2,int n);

int is\_palindrome(char \*array);

int num\_one(char \*array);

int main()

{

int n;

scanf("%d",&n);

char array[n][10];

int i,array\_2[n];

//输入字符

for (i=0;i<n;i++)

scanf("%s",&array[i]);

num(array,array\_2,n);

//输出数字

for (i=0;i<n;i++)

printf("%d\n",array\_2[i]);

return 0;

}

//分别统计字符串的子字符的回文字符个数，并记录在array\_2中

void num(char (\*array)[10],int \*array\_2,int n)

{

int i,j,k=0; //k用来统计array\_2

for (i=0;i<n;i++)

array\_2[i]=num\_one(array[i]);

}

//统计一个字符串子字符的回文字符串个数

int num\_one(char \*array)

{

int len=strlen(array);

char temp[10];

int i,j,k,m=0;

int sum=0;

for (i=0;i<len;i++)

{

for (j=i;j<len;j++)

{

for (k=i;k<=j;k++)

{

temp[m++]=array[k];

}

temp[m]='\0';

if (is\_palindrome(temp))

{

sum++;

}

m=0;

}

}

return sum;

}

//判断一个字符串是否为回文字符

int is\_palindrome(char \*array)

{

int len=strlen(array),i=0;

for (i=0;i<len/2;i++)

if ( array[i] != array[len-i-1] )

return 0;

return 1;

}

5.12

编写程序，对输入的若干单词按词典排序，不区分大小写。要求字符串的比较、复制、大小写转换、排序等工作均用函数实现（注意是自己编写函数，不能使用系统的字符串处理库函数）。  
在主函数中输入单词，调用函数排序，在主函数中输出结果。字符串个数不超过100个，每个单词的长度不超过20。  
输入格式：  
两行：第1行为单词个数n，第2行为n个单词，用空格隔开。  
输出格式：  
n行，每行一个单词。  
输入样例：  
5  
sin Cos log exp sqrt  
输出样例：  
Cos  
exp  
log  
sin  
sqrt

#include<stdio.h>

#include<math.h>

#define min(a,b) ((a)<(b))?(a):(b)

//声明函数

void reverse(char \*array);

int compare(char \*array\_1,char \*array\_2);

void copy(char \*array\_1,char \*array\_2);

int stlen(char \*array);

void transfer(char (\*array)[20],char (\*array\_1)[20],int n);

int main()

{

int n,i;

scanf("%d",&n);

char array[n][20],array\_1[n][20];

for (i=0;i<n;i++)

scanf("%s",array[i]);

//把输入array的复制到array\_1中

for (i=0;i<n;i++)

copy(array[i],array\_1[i]);

//把输入的字符转换为小写字母

for (i=0;i<n;i++)

reverse (array[i]);

//排序

transfer(array,array\_1,n);

//输出字符

for (i=0;i<n;i++)

printf("%s\n",array\_1[i]);

return 0;

}

//大小写转化函数

void reverse(char \*p)

{

int i;

for (i=0;i<stlen(p);i++)

if ( 'A' <=p[i] && p[i] <='Z' )

p[i]=p[i]-'A'+'a';

}

//大返回1，小返回-1，一样返回0

int compare(char \*p1,char \*p2)

{

for(;\*p1==\*p2;p1++,p2++)

if(\*p1=='\0')

return 0;

return (\*p1-\*p2)/abs(\*p1-\*p2);

}

//复制函数

void copy(char \*p1,char \*p2)

{

while((\*p2++=\*p1++));

\*p2='\0';

}

//统计字符长度

int stlen(char \*p)

{

int i=0;

while (p[i++]);

return i-1;

}

//转换字符数组的位置

void transfer(char (\*array)[20],char (\*array\_1)[20],int n)

{

int i,j;

char temp[20];

for (i=0;i<n;i++)

{

for (j=i;j<n;j++)

{

if ( compare ( array[i] , array[j] ) == 1 )

{

copy(array[i],temp);

copy(array[j],array[i]);

copy(temp,array[j]);

//同步操作

copy(array\_1[i],temp);

copy(array\_1[j],array\_1[i]);

copy(temp,array\_1[j]);

}

}

}

}

5.13

编写函数，使用递归实现字符串的逆序。编写主函数，输入字符串，调用函数逆序，输出字符串。要求1不使用库函数，2采用递归，3不在逆序函数中输出。  
输入：不带空格的字符串,长度小于200。  
输出：逆序的字符串。  
【输入输出样例】  
输入：  
abcdef  
输出:  
fedcba  
  
提示：逆序函数的参考原型：  
void mystrrev(char s[],int i,int j);  
其中i,j为要交换的两个字符的下标，起始值为0,n-1。

#include <stdio.h>

#include <string.h>

void mystrrev(char \*s,int i,int j);

int main()

{

char s[200];

scanf("%[^\n]",s);

int n=strlen(s);

mystrrev(s,0,n-1);

printf("%s",s);

return 0;

}

void mystrrev(char \*s,int i,int j)

{

char temp;

if ( i<j )

{

temp=s[i];

s[i]=s[j];

s[j]=temp;

mystrrev(s,i+1,j-1);

}

}

5.14

题目描述：  
对于给定的字符序列，从左至右将所有的数字字符取出拼接成一个无符号整数（字符序列长度小于100，拼接出的整数小于2^31,），计算并输出该整数的最大素因子（最大素因子：该数所有因子中的最大素数，素数的最大素因子因子为自身）。  
注意：从字符串中提取数字、计算该数字的最大素因子均需通过对应的函数实现。  
测试用例：  
输入描述：有多组数据，输入数据的第一行为一个正整数(<=10)，表示字符序列的数目，每组数据为一行字符序列。  
输出描述：对每个字符序列，取出所得整数的最大素因子，若字符序列中没有数字或者找出的整数为0，则输出0，每个整数占一行输出。  
示例：  
输入：  
2  
sdf0ejg3.f?9f  
abcde  
输出：  
13  
0  
  
示例解释：第一个字符串提取出的数为39,39的素因子有3,13,最大素因子为13。第二个字符串提取出的数为0，其最大素因子为0。

#include <stdio.h>

#include <math.h>

#include <string.h>

//声明函数

int is\_prime\_number(int x);

int the\_max\_prime\_number(int x);

void find\_number(char (\*array\_1)[100],int \*array\_2,int n);

int main()

{

int n,i,j,k=0;

scanf("%d",&n);

char array\_1[n][100];

int array\_2[20];

//录入数据

for (i=0;i<n;i++)

scanf("%s",array\_1[i]);

//选出数字

find\_number(array\_1,array\_2,n);

//输出

for (i=0;i<n;i++)

printf("%d\n",the\_max\_prime\_number(array\_2[i]));

return 0;

}

//找最大素数因子

int the\_max\_prime\_number(int x)

{

int i;

for (i=x;i>0;i--)

if ( x%i==0 && is\_prime\_number(i) )

return i;

return 0;

}

//判断是否为素数

int is\_prime\_number(int x)

{

int i;

for (i=x-1;i>=sqrt(x);i--)

if (x%i==0)

return 0;

return 1;

}

//选出数字

void find\_number(char (\*array\_1)[100],int \*array\_2,int n)

{

int sum=0,i,j,k=0;

for (i=0;i<n;i++)

{

for (j=0;j<strlen(array\_1[i]);j++)

if (isdigit(array\_1[i][j]))

sum=10\*sum+(array\_1[i][j]-'0'); //数字的acs码减48等于原本的数字！！！！！！

array\_2[k++]=sum;

sum=0;

}

}

5.15

编写一个函数，将一个字符串（字符串长度不大于40）中最长的单词及其长度输出。（注意最长单词可以有多个）  
【输入输出样例】  
示例1：  
输入：  
My mother is Alice  
输出：  
mother  
6  
示例2：  
输入：  
Nice to meet you  
输出：  
Nice  
meet  
4

#include <stdio.h>

#include <math.h>

#include <string.h>

int f(char\* array,int len);

void g(char\* array,int max,int len);

int main ()

{

char array[40];

int max;

gets(array);

max=f(array,strlen(array));

g(array,max,strlen(array));

printf("%d",max);

return 0;

}

//找最多数字

int f(char \*array,int len)

{

int i=0,flag=0,sum=0,max=0;

for (i=0;i<len;i++)

{

if (isalpha(array[i]))

sum++;

if ( array[i]==' ')

{

max=sum>max?sum:max;

sum=0;

}

}

return sum>max?sum:max;

}

//输出最多的字符串

void g(char \*array,int max,int len)

{

int i=0,k=0,sum=0,j=0;

char array\_2[40];

for (i=0;i<len+1;i++)

{

if (isalpha(array[i]))

{

sum++;

array\_2[k++]=array[i];

}

if ( array[i]==' ' || array[i]=='\0' )

{

array\_2[k]='\0';

if ( sum == max )

printf("%s\n",array\_2);

k=0;

sum=0;

}

}

}

6.1

编写一个函数void swap(int \*p1,int \*p2)。在主函数中输入两个整数，调用swap函数，并在主函数中输出交换后的整数。  
输入示例：  
3,56  
输出示例：  
56,3

|  |
| --- |
| #include<stdio.h>  **int** main()  {  **void** swap(**int** \*p1,**int** \*p2);  **int** x,y;  **scanf**("%d,%d",&x,&y);  **int** \*p1=&x,\*p2=&y;      swap(p1,p2);  **printf**("%d,%d",x,y);  **return** 0;   }    **void** swap(**int** \*p1,**int** \*p2)  {  **int** temp;      temp=\*p1;      \*p1=\*p2;      \*p2=temp;  } |

6.2

编写一个函数void fun(int \*p, int n)，其功能为对整数数组p[5]中的数字进行排序。排序结果在主函数中输出。 输入： 34 78 23 12 69 输出如下： 78 69 34 23 12

注意：输出数字之间用一个空格分隔

#include <stdio.h>

#include <stdlib.h>

int main()

{

void fun(int \*p, int n);

int \*a=(int \*)malloc(sizeof(int)\*5);

int i;

for (i=0;i<5;i++)

scanf("%d",a+i);

fun(a,5);

for (i=0;i<5;i++)

printf("%d%s",\*(a+i),(i==4)?"":" ");

return 0;

}

void fun(int \*p, int n)

{

int i,j,temp;

for (i=0;i<n;i++)

{

for (j=i;j<n;j++)

{

if ( \*(p+i) < \*(p+j) )

{

temp=\*(p+j);

\*(p+j)=\*(p+i);

\*(p+i)=temp;

}

}

}

}

6.3

编写一个函数，实现字符串的反转，要求使用指针实现。  
在主函数中输入字符串（大小不超过100），输出反转后的字符串  
示例：  
输入：abc.edi  
输出：ide.cba

#include <stdio.h>

#include <string.h>

int main()

{

void reverse(char \*array);

char array[100];

scanf("%s",array);

reverse(array);

printf ("%s",array);

}

void reverse (char \*array)

{

int len=strlen(array);

int i;

char temp;

for (i=0;i<len/2;i++)

{

temp=\*(array+i);

\*(array+i)=\*(array+len-i-1);

\*(array+len-i-1)=temp;

}

}

6.4

【题目描述】  
使用指针实现矩阵的转置，要求动态分配数组内存空间。第一行输入两个数，分别代表矩阵的行数和列数，从第二行开始以行优先顺序依次输入矩阵元素。  
【输入输出样例】  
输入：  
3 4  
1 2 3 4  
5 6 7 8  
9 10 11 12  
输出：  
1 5 9  
2 6 10  
3 7 11  
4 8 12  
提示： 动态数组定义可参考如下代码  
int \*p;  
p = (int \*) malloc(n \* m \* sizeof(int)); //p指向动态数组的起始位置,m和n为二维数组的行数和列数

#include <stdio.h>

#include <stdlib.h>

**int** main()

{

**int** row,column;

**scanf**("%d%d",&row,&column);

    //动态分配

**int** \*array=(**int** \*) **malloc**(row \* column \* **sizeof**(**int**));

**int** i,j;

    //输出

**for** (i=0;i<row;i++)

    {

**for** (j=0;j<column;j++)

        {

**scanf**("%d",array+i+j\*row);

        }

    }

    //输出

**for** (i=0;i<row\*column;i++)

    {

**printf**("%d%s",\*(array+i),((i+1)%row==0)?"\n":" ");

    }

**return** 0;

 }

6.5

若有n个数，循环右移一位指每个数移到相邻的右边一个位置，最右边的数移到最左边。如数1,2,3,4,5循环右移一位的结果为5,1,2,3,4；右移3位的结果为3,4,5,1,2。  
编写函数，实现数组元素的循环右移k位，要求用指针实现。  
  
题目考察点：指针与一维数组  
  
要求：主函数输入若干元素和右移位数，调用函数右移，在主函数中输出结果。元素个数不超过100,要求用指针实现。  
  
输入：三行，第1行为数组长度n；第2行为若干用空格分隔的整数；第3行一个整数，表示右移的位数。  
输出:一行，移位的结果，用一个空格隔开，末尾无空格。  
【输入输出样例】  
输入：  
5  
1 2 3 4 5  
3  
输出：  
3 4 5 1 2

#include <stdio.h>

#include <string.h>

#include<stdlib.h>

//声明函数

void move (int \*array,int len,int n);

int main()

{

int len;

scanf("%d",&len);

//动态分配

int \*array=(int\*)malloc(sizeof(int)\*len);

int i;

for (i=0;i<len;i++)

{

scanf ("%d",&array[i]);

}

//录入次数

int n;

scanf("%d",&n);

//使用函数

move(array,len,n);

//输出

for (i=0;i<len;i++)

{

printf("%d%s",array[i],(i==len-1)?"":" ");

}

return 0;

}

void move (int \*array,int len,int n)

{

int i=0,j=0,temp=0;

for (i=0;i<n;i++)

{

temp=\*(array+len-1);

for (j=len-1;j>=1;j--)

{

\*(array+j)=\*(array+j-1);

}

\*(array+0)=temp;

}

}

6.6

有一段文字，其中有若干整数。编写函数，找出其中的整数，存入一个数组中。函数原型为：  
int findint(char \*s,int \*a);  
其中s表示带数字的字符串，a表示存整数的数组，返回值为字符串中的整数个数。字符串中只有正整数。  
  
编写主函数，输入字符串，调用函数识别整数，逆序输出它们。例如，输入字符串“The Wandering Earth exhibition at the China Science and Technology Museum in Beijing, April 4, 2019.”识别数字，输出“2019,4”。  
  
输入：带空格字符串  
输出：若干整数，用逗号隔开。  
【输入输出样例】  
输入：  
he Wandering Earth exhibition at the China Science and Technology Museum in Beijing, April 4, 2019.  
输出：  
2019,4  
  
题目考察点：指针与字符数组  
题目延伸：（1）如果字符串中有负整数，如何修改程序。（2）如果字符串中是实数，如何编写程序。

#include <stdio.h>

#include <string.h>

#include <ctype.h>

#define N 100 //定义字符串数组的整数数组的最大值

//全局定义函数

int findint(char \*s,int \*a);

void reverse\_printf(int \*a,int n);

int main()

{

char array\_1[N];

int array\_2[N];

scanf("%[^\n]",array\_1); //录入字符串

char \*s=array\_1;

int \*a=array\_2; //定义指针

int len=findint(s,a);

a=array\_2; //重新指向初地址

reverse\_printf(a,len); //逆序输出

return 0;

}

//s为字符串数组，a为整数数组，返回值为字符串中的整数个数

int findint(char \*s,int \*a)

{

int temp=0,flag=0,sum=0,i;

while (\*s)

{

if ( isdigit(\*s) )

{

temp=temp\*10+\*s-'0';

flag=1;

}

else

{

if (flag)

{

\*a++=temp;

sum++;

}

temp=0;

flag=0;

}

s++;

}

//补充最后一种情况

if (temp!=0)

{

\*a++=temp;

sum++;

}

return sum;

}

//逆序输出函数

void reverse\_printf(int \*a,int n)

{

a+=n-1;

while ( --n!=-1 )

printf("%d%s",\*(a--),(n==0)?"":",");

}

6.7

写一个函数int mystrcmp（char \*p1, char \*p2）用来实现两个字符串的比较。在主函数中输入两个字符串，输出对应的结果。字符串长度不超过20。  
输入：  
hello  
world  
则输出：-1  
输入：  
world  
hello  
则输出1  
输入：  
hello  
hello  
则输出：0

6.8

编写一个函数，实现字符串的前n个字符的复制。  
要求使用指针实现，函数定义如下：  
char \*mystrncpy(char \*str1,char \*str2,int n)，将str2的前n个字符复制到str1中，函数返回为字符指针。  
在主函数中输入字符串和n（字符串可以包含空格），输出复制后的字符串，如果n超过字符串中包含的字符个数（不算空格），则提示出错。  
输入：  
hello  
2  
输出：  
he  
输入：  
hello  
6  
输出：  
error  
输入：  
hell dsff  
9  
输出：  
error  
输入：  
hello world  
8  
输出：  
hellowor  
  
注：字符串中含有空格的话，则跳过空格赋值（参见上面最后一个例子）。

#include<stdio.h>

#include<string.h>

char \*mystrncpy(char \*str1,char \*str2,int n);

void function\_1(char \*p);

int main()

{

char array\_1[100];

char array\_2[6]="error\0";

scanf("%[^\n]",array\_1);

function\_1(array\_1);

int n;

scanf("%d",&n);

char \*p=mystrncpy(array\_1,array\_2,n);

printf("%s",p);

return 0;

}

void function\_1(char \*p)

{

int i=0,j=0;

for (i=0;i<strlen(p);i++)

if (\*(p+i)!=' ')

\*(p+j++)=\*(p+i);

\*(p+j)='\0';

}

char \*mystrncpy(char \*str1,char \*str2,int n)

{

if (n>strlen(str1))

return str2;

else

{

\*(str1+n)='\0';

return str1;

}

}

6.9

编写函数，将一个5\*5的矩阵中最大的元素放在中心，4个角分别放4个最小的元素（顺序为从左到右，从上到下依次从小到大存放）。  
编写主函数，输入5\*5的矩阵，调用函数进行上述变换，在主函数中输出结果。  
输入：5行整数，表示5\*5矩阵  
输出：5行整数，每行中的元素用一个英文空格隔开，末尾无空格。  
【输入输出样例】  
输入：  
31 34 33 32 25  
30 29 28 27 26  
35 24 23 22 21  
20 19 18 17 16  
15 14 13 12 11  
  
输出:  
11 34 33 32 12  
30 29 28 27 26  
23 24 35 22 21  
20 19 18 17 16  
13 31 15 25 14  
  
题目考察点：指针与二维数组  
注意，元素的交换只交换最大和中心元素，最小和四角元素，其他元素位置不变。假设元素互不相同。

#include<stdio.h>

void reverse(int \*array);

void swap(int \*p1,int \*p2);

int main()

{

int array[5][5];

int i,j,k=0;

int array\_2[25];

for (i=0;i<5;i++)

{

for(j=0;j<5;j++)

{

scanf("%d",&array[i][j]);

array\_2[k++]=array[i][j];

}

}

reverse(array\_2);

for (i=0;i<5;i++)

{

for(j=0;j<5;j++)

{

if (array[i][j]==array\_2[0])

{

swap(&array[i][j],&array[0][0]);

}

}

}

for (i=0;i<5;i++)

{

for(j=0;j<5;j++)

{

if (array[i][j]==array\_2[1])

{

swap(&array[i][j],&array[0][4]);

}

}

}

for (i=0;i<5;i++)

{

for(j=0;j<5;j++)

{

if (array[i][j]==array\_2[2])

{

swap(&array[i][j],&array[4][0]);

}

}

}

for (i=0;i<5;i++)

{

for(j=0;j<5;j++)

{

if (array[i][j]==array\_2[3])

{

swap(&array[i][j],&array[4][4]);

}

}

}

for (i=0;i<5;i++)

{

for(j=0;j<5;j++)

{

if (array[i][j]==array\_2[24])

{

swap(&array[i][j],&array[2][2]);

}

}

}

for (i=0;i<5;i++)

{

for(j=0;j<5;j++)

{

printf("%d ",array[i][j]);

}

printf("\n");

}

}

void reverse(int\* array)

{

int i,j,temp;

for (i=0;i<25;i++)

{

for (j=i;j<25;j++)

{

if ( array[i]>array[j])

{

temp=array[i];

array[i]=array[j];

array[j]=temp;

}

}

}

}

void swap(int \*p1,int \*p2)

{

int temp=\*p1;

\*p1=\*p2;

\*p2=temp;

}

6.10

题目描述：  
有一种简单的字符串压缩算法，对于字符串中连续出现的同一个字符，用该字符加上连续出现的次数来表示（连续出现次数小于3时不压缩，原样输出），例如，字符串aaaaabbbabaaaaaaaaaaaaabbbb可压缩为a5b3aba13b4。请设计一个程序，将字符串压缩并输出。  
测试用例：  
输入说明：只有一组数据，数据为一个字符串（长度不大于100，只包含字母）。  
输出说明：在一行上输出压缩后的字符串（长度不超过50），最后换行。  
示例1：  
输入：  
aaaaabbbabaaaaaaaaaaaaabbbb  
输出：  
a5b3aba13b4  
示例2：  
输入：  
a  
输出：  
a  
示例3：  
输入：  
aabbbb  
输出：  
aab4

#include <stdio.h>

#include <string.h>

void compress(char \*p,int \*n);

int main()

{

char array[100];

scanf("%s",array);

int len[50];

//多设置一个数组也是无奈之举，我想不到更好的办法了。

compress(array,len);

int i,j=strlen(array);

//这样子的输出真的是无奈之举

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

{

if (\*(len+i)==1)

{

printf("%c",\*(array+i));

}

else if (\*(len+i)==2)

{

printf("%c%c",\*(array+i),\*(array+i));

}

else

{

printf("%c%d",\*(array+i),\*(len+i));

}

}

printf("\n");//题目说结束要换行

return 0;

}

void compress(char \*p,int \*n)

{

int i,j,k,sum=0;

char temp=\*p;

for (i=0,j=1,k=0;\*(p+i)!='\0';i++)

{

if ( \*(p+i)==temp )

{

sum++;

}

else

{

\*(n+k++)=sum;

sum=1;

temp=\*(p+i);

\*(p+j++)=\*(p+i);

}

}

\*(n+k)=sum;

\*(p+j)='\0';

}

6.11

编写一个函数，删除一个字符串（不超过100）中指定的所有某个字符（默认指定的字符存在于字符串中），要求使用指针实现。  
示例：  
输入：  
abdbca  
a  
输出：bdbc  
输入格式说明：字符串和字符之间有换行

#include<stdio.h>

#define N 100

void delete\_char(char \*p,char c);

int main()

{

char array[N];

gets(array);

char c;

scanf("%c",&c);

char \*p=array;

delete\_char(p,c);

printf("%s",array);

return 0;

}

void delete\_char(char \*p,char c)

{

int i,j;

for (i=0,j=0;\*(p+i)!='\0';i++)

{

if ( \*(p+i)!=c )

\*(p+j++)=\*(p+i);

}

\*(p+j)='\0';

}

6.12

题目描述：  
输入n个互不相同的整数，n由用户指定，将其中最小的数与第一个数对换，把最大的数与最后一个数对换。要求对数组的定义初始化采用动态数组的形式，所有的操作均采用指针的方式。  
测试用例：  
输入描述：输入数据的第一行为一个正整数n，表示数组中元素的个数；第二行输入n个整数。  
示例1：  
输入：  
5  
5 4 3 2 1  
输出：  
1 4 3 2 5  
示例2：  
输入：  
3 2 5 -1 1  
输出：  
-1 2 1 3 5

#include <stdio.h>

#include <string.h>

#include <stdlib.h>

**void** swap(**int** \*p,**int** len);

**int** main()

{

**int** len=0;

**scanf**("%d",&len);

    //动态分配

**int** \*array=(**int** \*)**malloc**(len\***sizeof**(**int**));

    //录入

**int** i;

**for** (i=0;i<len;i++)

    {

**scanf**("%d",array+i);

    }

    //交换

    swap(array,len);

    //输出

**for** (i=0;i<len;i++)

    {

**printf**("%d%s",\*(array+i),(i==len-1)?"":" ");

    }

**return** 0;

}

//交换函数

**void** swap(**int** \*p,**int** len)

{

**int** temp=0,max=\*p,min=\*p,i=0,flag\_max=0,flag\_min=0;

**for** (i=0;i<len;i++)

    {

**if** (max<\*(p+i))

        {

            max=\*(p+i);

            flag\_max=i;

        }

**if** (min>\*(p+i))

        {

            min=\*(p+i);

            flag\_min=i;

        }

    }

**if** (flag\_min!=0 && flag\_max!=0)

    {

        temp=\*(p+0);

        \*(p+0)=min;

        \*(p+flag\_min)=temp;

    }

**if** (flag\_min!=0 && flag\_max==0)

    {

        temp=\*(p+0);

        \*(p+0)=min;

        \*(p+flag\_min)=temp;

        flag\_max=flag\_min;

    }

**if** (flag\_max!=len-1)

    {

        temp=\*(p+len-1);

        \*(p+len-1)=max;

        \*(p+flag\_max)=temp;

    }

}

6.13

编写程序，输入正整数n和n个学生的学号和三门课程成绩，计算每个人的平均分，按平均分从小到大排序，输出排序结果。学号的长度不超过10位；三门课程成绩为整数；平均成绩为实数，输出平均成绩保留两位小数；学生人数不超过50人。  
提示：请用多个函数+指针实现。  
【输入输出样例】  
输入：  
5  
001 80 80 80  
002 60 70 80  
003 80 90 93  
004 50 60 70  
005 60 70 83  
输出：  
004 50 60 70 60.00  
002 60 70 80 70.00  
005 60 70 83 71.00  
001 80 80 80 80.00  
003 80 90 93 87.67  
  
题目考察点：指针与二维数组

6.13

编写程序，输入正整数n和n个学生的学号和三门课程成绩，计算每个人的平均分，按平均分从小到大排序，输出排序结果。学号的长度不超过10位；三门课程成绩为整数；平均成绩为实数，输出平均成绩保留两位小数；学生人数不超过50人。  
提示：请用多个函数+指针实现。  
【输入输出样例】  
输入：  
5  
001 80 80 80  
002 60 70 80  
003 80 90 93  
004 50 60 70  
005 60 70 83  
输出：  
004 50 60 70 60.00  
002 60 70 80 70.00  
005 60 70 83 71.00  
001 80 80 80 80.00  
003 80 90 93 87.67  
  
题目考察点：指针与二维数组

6.14

【题目描述】  
使用指针实现字符串拼接函数strcat()。（两个字符串长度均不超过20）  
【输入输出样例】  
输入：  
hello  
world  
输出：  
helloworld

#include<stdio.h>

#define N 20

void strlink(char \*p1, char \*p2);

int main()

{

char array\_1[N],array\_2[N];

scanf("%s",array\_1);

scanf("%s",array\_2);

strlink(array\_1,array\_2);

printf("%s",array\_1);

return 0;

}

void strlink(char \*p1, char \*p2)

{

while (\*p1++);

p1--;

while (\*p1++=\*p2++);

}

7.1

定义表示平面点的结构体类型，成员有x坐标和y坐标，数据类型用float。定义计算两个点的距离的函数，计算两个平面点的距离，返回float。主函数中定义两个平面点的结构体变量，输入点的坐标，调用函数求距离，显示它们的距离(输出%f)。  
输入：四个实数，前两个表示一个点的x,y坐标；后两个表示另一个点的x,y坐标.  
输出：两个点的距离。  
注意，距离函数的参数是结构体类型！！！  
【输入输出样例】  
输入：  
2 2 1 1  
输出：  
1.414214

#include<stdio.h>

#include<math.h>

struct DOT

{

float x1;

float y1;

float x2;

float y2;

};

struct DOT dot1;

float f(struct DOT \*p);

int main()

{

scanf("%f",&dot1.x1);

scanf("%f",&dot1.y1);

scanf("%f",&dot1.x2);

scanf("%f",&dot1.y2);

struct DOT \*p=&dot1;

printf("%f",f(p));

return 0;

}

float f(struct DOT \*p)

{

return sqrt((p->x1-p->x2)\*(p->x1-p->x2)+(p->y1-p->y2)\*(p->y1-p->y2));

}

7.2

编写程序，从键盘输入 n (n<1000)个职员的工号（长为5位的char类型数据，工号互不相同，唯一表示一名职员）、姓名（char 类型数据，不超过20）及其性别（char类型)，统计其中性别(sex)为'M'的人数。  
输入：第一行有一个整数n，表示以下有 n 个人信息将会输入。保证n不大于1000。  
以后的n行中，每行数据为一个人的学号，姓名和性别。  
【输入输出样例】  
输入：  
3  
10001 Li M  
10002 Zhang F  
10003 Fun M  
输出：  
2

#include<stdio.h>

struct staff

{

char num[5];

char name[20];

char sex[1];

};

int f(struct staff \*p,int n);

int main()

{

int n;

scanf("%d",&n);

struct staff array[n];

int i;

for (i=0;i<n;i++)

{

scanf("%s",array[i].num);

scanf("%s",array[i].name);

scanf("%s",array[i].sex);

}

printf("%d",f(array,n));

return 0;

}

int f(struct staff \*array,int n)

{

int i,num=0;

for (i=0;i<n;i++)

if ( array[i].sex[0] == 'M' )

num++;

return num;

}

7.3

【题目描述】  
从键盘输入n（n<10）本书的名称（书名长度不超过50，可以包含空格）和定价（不考虑定价相同的情况）并存入结构体数组中，从中查找定价最高的最低的书的名称和定价，并输出。  
输入：第一行输入书籍的个数，从第二行开始输入书籍信息（对每本书籍，第一行输入名称，第二行输入定价(浮点数)）。  
输出：第一行为定价最高的图书名称和定价，第二行为定价最低的图书名称和定价（名称和定价之间空一格，定价保留两位小数）。  
【输入输出样例】  
输入：  
3  
book1  
89.0  
book 2  
139.0  
book3  
78.0  
输出：  
book 2 139.00  
book3 78.00

#include <stdio.h>

struct BOOK

{

char name[50];

float price;

};

int main()

{

int n;

scanf("%d",&n);

struct BOOK book1[n];

int i;

for (i=0;i<n;i++)

{

getchar();

scanf("%[^\n]",book1[i].name);

scanf("%f",&book1[i].price);

}

float max=book1[0].price,min=book1[0].price;

int flag1=0,flag2=0;

for (i=0;i<n;i++)

{

if (max<book1[i].price)

{

flag1=i;

max=book1[i].price;

}

if (min>book1[i].price)

{

flag2=i;

min=book1[i].price;

}

}

printf("%s %0.2f\n",book1[flag1].name,book1[flag1].price);

printf("%s %0.2f",book1[flag2].name,book1[flag2].price);

return 0;

}

7.4

定义表示日期的结构体类型，成员包括年、月、日，都用整数表示。编写函数，计算一个结构体类型的日期是本年中的第几天（要考虑闰年），如2018年1月1日是当年（2018）年的第1天。  
编写主函数，定义结构体变量。输入结构体变量的年、月、日，调用函数计算天数并显示。  
【输入输出样例】  
输入：  
2019 4 15  
输出：  
105  
提示：主函数如下：  
int main()  
{  
struct Date d1;  
scanf("%d %d %d",&d1.year,&d1.month,&d1.day);  
printf("%d\n",daysofyear(d1));  
return 0;  
}

|  |
| --- |
| #include<stdio.h>  **struct** Date  {  **int** year;  **int** month;  **int** day;  };    **int** main()  {  **struct** Date d1;  **scanf**("%d %d %d",&d1.year,&d1.month,&d1.day);  **printf**("%d\n",days\_of\_year(d1));  **return** 0;  }    **int** days\_of\_year(**struct** Date d1)  {  **int** flag=0,sum=0,i=0;  **if** ((d1.year%4==0)&&(d1.year%100!=0)||(d1.year%400==0)) flag=1;  **for** (i=0;i<d1.month;i++)      {  **if** (i==1||i==3||i==5||i==7||i==8||i==10||i==12)          {              sum+=31;          }  **if** (i==4||i==6||i==9||i==11)          {              sum+=30;          }  **if** (i==2&&flag==1)          {              sum+=29;          }  **if** (i==2&&flag!=1)          {              sum+=28;          }      }      sum+=d1.day;  **return** sum;  } |

7.5

设有5名歌手(编号为1-5)参加歌咏比赛，另有6名评委打分，每位歌手的得分从键盘输入：依次输入第1个歌手的6位评委打分(10分制，分数为整型，分数之间使用空格分隔)，第2个歌手的6位评委打分…以此类推。计算出每位歌手的最终得分(扣除一个最高分和一个最低分后的平均分，最高分或最低分如有重复，则只分别去掉一次。最终得分保留2位小数)，最后按最终得分由高到低的顺序输出每位歌手的编号及最终得分（如果平均分相同，则按照序号先后输出）。输入输出示例如下；  
输入：  
4 6 8 3 5 1  
3 5 6 8 1 4  
5 6 7 3 8 2  
5 7 9 1 0 4  
6 7 9 2 1 5  
输出：  
003 5.25  
005 5.00  
001 4.50  
002 4.50  
004 4.25  
  
  
【提示】结构体定义如下：  
struct Singer  
{  
int number;  
int score[6];  
float average;  
} singer;

#include <stdio.h>

struct Singer

{

int number;

float score[6];

float average;

};

float f(struct Singer p);

void sort(struct Singer \*p);

int main()

{

// printf("Please input singer's score:\n");

struct Singer singer[5];

int i,j;

//录入

for (i=0;i<5;i++)

singer[i].number=i+1;

for (i=0;i<5;i++)

for (j=0;j<6;j++)

scanf("%f",&singer[i].score[j]);

//得平均数

for (i=0;i<5;i++)

singer[i].average=f(singer[i]);

//排序输出

sort(singer);

for (i=0;i<5;i++)

printf("00%d %0.2f\n",singer[i].number,singer[i].average);

return 0;

}

float f(struct Singer p)

{

float max=p.score[0],min=p.score[0];

int i;

float sum=0;

for (i=0;i<6;i++)

{

max=max>p.score[i]?max:p.score[i];

min=min<p.score[i]?min:p.score[i];

sum+=p.score[i];

}

sum=sum-max-min;

sum=sum/4;

return sum;

}

//逆序排序

void sort(struct Singer \*p)

{

int i,j;

for (i=0;i<5;i++)

{

for (j=i;j<5;j++)

{

if(p[i].average<p[j].average)

{

struct Singer temp;

temp=p[i];

p[i]=p[j];

p[j]=temp;

}

}

}

}

7.6

【题目描述】  
用结构体定义复数，通过函数调用实现两个复数的乘除运算，结果保留2位小数。  
第一行输入第一个复数的实部和虚部，第二行输入第二个复数的实部和虚部，实部和虚部之间用一个空格隔开。输出的第一行为乘法结果，输出的第二行为除法结果。  
【输入输出样例】  
输入：  
-3.5 4.2  
6.5 -2.3  
输出：  
-13.09+35.35i  
-0.68+0.40i  
【提示】  
复数结构体参考如下定义：  
struct Complex{  
double a;  
double b;  
};  
两个复数的乘法和除法函数声明如下  
Complex mulC(Complex C1, Complex C2)  
Complex divC(Complex C1, Complex C2)

#include<stdio.h>

#include<math.h>

**struct** DOT

{

**float** x1;

**float** y1;

**float** x2;

**float** y2;

};

**struct** DOT dot1;

**void** f1(**struct** DOT p);

**void** f2(**struct** DOT p);

**int** main()

{

**scanf**("%f",&dot1.x1);

**scanf**("%f",&dot1.y1);

**scanf**("%f",&dot1.x2);

**scanf**("%f",&dot1.y2);

    f1(dot1);

**printf**("\n");

    f2(dot1);

**return** 0;

 }

**void** f1(**struct** DOT p)

{

**printf**("%0.2f%s%0.2fi",(p.x1\*p.x2-p.y1\*p.y2),((p.y1\*p.x2+p.x1\*p.y2)>0)?"+":"",(p.y1\*p.x2+p.x1\*p.y2));

}

**void** f2(**struct** DOT p)

{

**printf**("%0.2f%s%0.2fi",(p.x1\*p.x2+p.y1\*p.y2)/(p.y2\*p.y2+p.x2\*p.x2),((p.y1\*p.x2+p.x1\*p.y2)>0)?"+":"",(p.y1\*p.x2-p.x1\*p.y2)/(p.y2\*p.y2+p.x2\*p.x2));

}

7.7

题目描述：  
编写程序，从键盘输入 n (n<100)个学生的学号（长为10位的char类型数据，学号互不相同，唯一表示一名学生）、姓名（char 类型数据）及其对应成绩（int类型），按成绩从高到低排序，并输出排序后的学生信息。若两个学生成绩相同，则按学号升序排序。  
测试用例：  
输入描述：有多组数据，输入数据的第一行为一个正整数，表示学生个数，之后每行数据为学生学号、姓名及其对应成绩。  
示例：  
输入：  
4  
10000002 Zhangsan 85  
10000001 Lisi 90  
10000000 Bob 85  
10000003 Alice 45  
输出：  
10000001 Lisi 90  
10000000 Bob 85  
10000002 Zhangsan 85  
10000003 Alice 45

#include<stdio.h>

#include<math.h>

struct Student

{

char num[10];

char name[20];

int result;

};

void sort(struct Student \*p);

void sort2(struct Student \*p);

int main()

{

int n;

scanf("%d",&n);

struct Student array[n];

int i;

for (i=0;i<n;i++)

{

scanf("%s",array[i].num);

scanf("%s",array[i].name);

scanf("%d",&array[i].result);

}

sort(array);

sort2(array);

for (i=0;i<n;i++)

{

printf("%s ",array[i].num);

printf("%s ",array[i].name);

printf("%d\n",array[i].result);

}

return 0;

}

//正序排序

void sort(struct Student \*p)

{

int i,j;

for (i=0;i<5;i++)

{

for (j=i;j<5;j++)

{

if(p[i].result<p[j].result)

{

struct Student temp;

temp=p[i];

p[i]=p[j];

p[j]=temp;

}

}

}

}

//序号排序

void sort2(struct Student \*p)

{

int i,j;

for (i=0;i<5;i++)

{

for (j=i;j<5;j++)

{

if(p[i].result==p[j].result)

{

if (strcmp(p[i].num,p[j].num)>0)

{

struct Student temp;

temp=p[i];

p[i]=p[j];

p[j]=temp;

}

}

}

}

}

7.8

编写程序，输入一个日期和一个正整数k，计算k天以后是哪天。例如，输入的日期为2019年5月15日，k为16，则2019年4月15日的16天后 是2019年5月1日，输出2019-5-1。  
输入：四个整数，前三个分别表示年、月、日 ，后一个标识天数k  
输出：k天之后的日期，数据间用英文减号隔开。  
  
【输入输出样例】  
输入：  
2019 4 15 16  
输出：  
2019-5-1  
  
提示：（1）日期用结构体。  
（2）先考虑加1一天的函数。  
（3）函数可以返回结构体。  
（4）主函数如下：  
int main()  
{  
struct DATE d1,d2;  
int k;  
scanf("%d %d %d",&d1.year,&d1.month,&d1.day);  
scanf("%d",&k);  
d2=adddays(d1,k);  
print(d2);  
return 0;  
}

#include<stdio.h>

struct Date

{

int year;

int month;

int day;

};

void days\_of\_year(struct Date d1,int day2);

int isleap(int year);

int main()

{

struct Date d1;

scanf("%d %d %d",&d1.year,&d1.month,&d1.day);

int day2;

scanf("%d",&day2);

days\_of\_year(d1,day2);

return 0;

}

void days\_of\_year(struct Date d1,int day2)

{

int flag=0,sum=0,i=0;

int year=d1.year,month=d1.month,day=d1.day;

int array[13][2]={{0,0},{31,31},{28,29},{31,31},{30,30},{31,31},{30,30},{31,31},{31,31},{30,30},{31,31},{30,30},{31,31}};

for (i=1;i<=day2;i++)

{

flag=isleap(year);

day=day+1;

if ( day > array[month][flag] )

{

month++;

day=1;

}

if ( month > 12)

{

year++;

month=1;

}

}

printf("%d-%d-%d",year,month,day);

}

int isleap(int year)

{

return ((year%4==0)&&(year%100!=0)||(year%400==0));

}