一、二维数组

```
数据类型  数组名[行标][列标];
```

一维数组中元素的最大下标是len-1;

二维数组中行数的下标[0,行标-1];

二维数组中列标的下标[0,列标-1];

二维数组，数组中元素的地址也是连续的。

二维数组名也是一个常量，代表的首元素的地址。

```
&arr[0][0]==arr;
```

1.1二维数组的初始化和赋值

通过下标访问

```
#include <stdio.h>
int main(int argc, const char *argv[])
{
    int arr[2][2];//能存四个元素的二维数组
    int arr[2][2] = {2,3,4,5};//定义一个二维数组并初始化
    int arr[2][2] = {2,3,4};//不完全初始化
    int arr[2][2] = {{1},{2}};//以行为单位不完全初始化
    int arr[2][2] = {{1,2},{3,4}};
    int arr[2][2] = {{,2},{,1}};//错的
    int arr[2][] = {2,3,4,5}; //不可以省略列号
    int arr[][2] = {2,3,4,5}; //可以省略行号
    int arr[][]={{2,3,4,5}};  //错的
    printf("%p\n",arr);//二维数组的数组名表示首元素的地址
    printf("%ld\n",sizeof(arr)/sizeof(int));
    printf("%p\n",&arr[0][0]); //0x7ffe382cad10
    printf("%p\n",&arr[0][1]); //0x7ffe382cad14
    printf("%p\n",&arr[1][0]); //0x7ffe382cad18
    printf("%p\n",&arr[1][1]); //0x7ffe382cad1c
    return 0;
}
```

1.2二维数组的大小

单个元素的数据类型*元素个数

1.3二维数组的输入输出

```
#include <stdio.h>
int main(int argc, const char *argv[])
{
    int arr[2][3] = {0};
    int i,j;
    for(i=0;i<2;i++) //遍历每行
```

```
    {
        for(j=0;j<3;j++) //遍历每列
        {
            scanf("%d",&arr[i][j]);
        }
    }
    for(i=0;i<2;i++) //遍历每行
    {
        for(j=0;j<3;j++) //遍历每列
        {
            printf("%d",arr[i][j]);
        }
    }
    return 0;
}
```

练习：

杨辉三角

```
#include <stdio.h>
int main(int argc, const char *argv[])
{
    int arr[10][10] = {0};
    arr[0][0] = 1; //把二维数组中的第一个元素置1
    int i,j;
    for (i=1;i<10;i++) //外层循环,遍历二维数组的每一行
    {
        //arr[i][0]=0;
        for (j=0;j<10;j++) //内层循环，遍历二维数组的每一列
        {
            if (j==0) //给每一行的第一列置1
            {
                arr[i][j] = 1;
                //continue;
            }
            else
            {
                arr[i][j] = arr[i-1][j]+arr[i-1][j-1];
                //除第一列外的每一列元素等于他的上一行与他同列的元素+他的上一行上一列的元素和
            }
        }
    }
    for (i=0;i<10;i++)
    {
        for (j=0;j<=i;j++)
        {
            printf("%-3d ",arr[i][j]);
        }
        putchar(10);
    }
    return 0;
}
```

找二维数组中最大元素及行标列标

```c
#include <stdio.h>
int main(int argc, const char *argv[])
{
    int arr[4][2]={2,3,4,5,6,32,56,98};
    int i,j;
    int max_h,max_l;
    max_h = 0;//假定最大元素行标是0
    max_l = 0;//假定最大元素列标是0
    for (i=0;i<4;i++)  //遍历行
    {
        for (j=0;j<2;j++)//遍历列
        {
            if (arr[max_h][max_l]<arr[i][j])
                //如果数组中的元素，比我假定的大，就交换两者的下标
            {
                max_h = i;
                max_l = j;
            }
        }
    }
    printf("%d   %d   %d",arr[max_h][max_l],max_h,max_l);
    //通过下标访问数组中的最大元素。
    return 0;
}
```

二、字符串

C语言中没有字符串这个数据类型，字符串是用字符数组来存储的，字符串末尾有一个'\0',我们看不见，这是字符串结束的标志。

2.1定义

```c
#include <stdio.h>
int main(int argc, const char *argv[])
{
    char c[] = "hello";//字符串常量初始化
    char c1[] = {'h','e','l','l','o','\0'};//\0是字符串的结束标志
    char c2[3] = ""; //默认为0
    char c3[3] ={'a'};//不完全初始化，未初始化部分默认为0
    printf("%ld\n",sizeof(c3));
    printf("%ld\n",sizeof(c));//打印大小占6个字节，为结尾的'\0'分配了空间
    printf("%s\n",c1);
    printf("%c\n",c2[0]);

    return 0;
}
```

2.2gets/puts

gets();

```c
char c[21] = "";
gets(c); //把gets函数获取到的数据赋给c这个字符数组。
```

格式：gets(输入给谁);

字符串输入函数，不会检查越界，所以会报警告

```
test6.c:5:2: warning: implicit declaration of function ‘gets’; did you mean ‘fgets
’? [-Wimplicit-function-declaration]
  gets(c);
  ^~~~
  fgets
/tmp/cc1T9lSo.o：在函数‘main’中：
test6.c:(.text+0x32)：警告：  the `gets' function is dangerous and should not be us
ed.
```

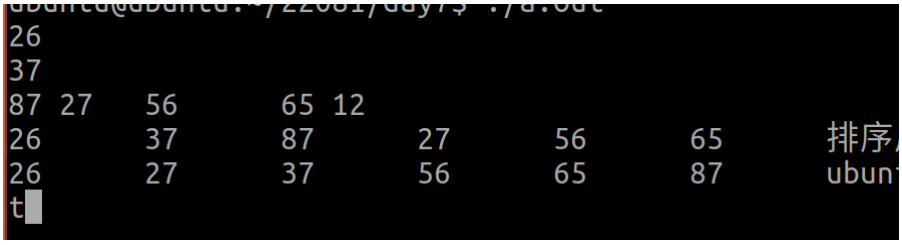
puts();

字符串输出函数，

格式：puts(要输出的内容);

补充

scanf以空格 tab键和回车作为标志，来分隔两个数据，如果获取到了与占位符数量对应的数据，按回车输入就结束。



这道题我需要6个数据，所以当我输入完第一个数据按下回车，输入不会终止，第三行数据，由于没有按回车都存入了scanf的缓冲区中，scanf在缓冲区内，以空格，tab键和回车作为分隔标志，找数据。

2.3str函数族

需要导入头文件<string.h>,从字符串的首地址开始，找到'\0'停止。

2.3.1strlen()

求字符串**实际**长度的，不包括'\0'

自己实现的strlen功能

```c
#include <stdio.h>
#include <string.h>
int main(int argc, const char *argv[])
{
    char s1[6] = "hello";
    char s2[4] = "hel";
    int i=0,count =0;
    while (s2[i]!='\0') //控制访问字符数组中的元素，到'\0'截至
    {
        count++;
        i++;
    }
    printf("%d\n",count);
    return 0;
}
```

2.3.2strcpy()

```
函数原型
    char *strcpy(char *dest, const char *src);
要求dest数组足够大。
```

把src复制到dest数组当中。

自己实现的strcpy功能

```
#include <stdio.h>
int main(int argc, const char *argv[])
{
    char s1[10] = "hello";
    char s2[20];
    int i =0;
    while(s1[i]!='\0')//控制访问字符数组中的元素，到'\0'截至
    {
        s2[i] = s1[i];
        i++;
    }
    s2[i] = '\0';//要给目标字符数组一个'\0'
    puts(s2);
    return 0;
}
```

2.3.3strcmp()

```
函数原型：int strcmp(const char *s1, const char *s2);

char c[21] = {'a','e','\0','l','l'};
char c2[5] = "hi";
printf("%d\n",strcmp(c1,c2));
输出：-7
```

比较字符串的,返回的是两个字符串同一位置不同字符相差的ASCII码值。

strcmp(c1,c2);

比较的逻辑：

从字符串的第一个位置开始比较，如果两个的第一个位置相等那就向下比较，只要相等就一直向下比较，直到不等/找到其中一个字符串的'\0'为止比较的是字符的ASCII码。

c1>c2，返回正数，不同位的两个字符的ASCII码的差值

c1<c2，返回负数

c1==c2，返回0

2.3.4strcat()

函数原型：char *strcat(char *dest, const char *src);
把src拼接到dest中。
拼接思想：找到dest结尾的'\0'，从'\0'这一位开始插入，把整个src字符串都拼接到dest后面。

```
char c[21] = {'a','e','\0','l','l'};
char c2[5] = "hi";
printf("%s\n",strcat(c1,c2));
```

输出：aehi

```
#include <stdio.h>
#include <string.h>
int main(int argc, const char *argv[])
{
    char c[21] = {'a','e','\0','l','l'};
    printf("%ld\n",strlen(c));
    char c1[23];
    strcpy(c1,c);
    char c2[5] = "hi";
    printf("%s\n",strcat(c1,c2));
    //strcat(c,c2);
    printf("%s\n",c);
    return 0;
}
```

```
ubuntu@ubuntu:~/22081/day7$ gcc test
ubuntu@ubuntu:~/22081/day7$ ./a.out
2
aehi
ae
ubuntu@ubuntu:~/22081/day7$
```