实例1 字符串长度的统计

#include<stdio.h>

#include<string.h>//字符串处理头文件

#include<stdlib.h>

int main()

{

char \*s="Golden Global View";//测试字符串

printf("%s has %d chars\n",s,strlen(s)); //函数调用，以及程序结果输出

system("pause");

return 0;

}

实例2 字符串拼接

#include<string.h>

#include<stdlib.h>

#include<stdio.h>

int main()

{

char d[20]="Golden Global";

char \*s=" View";

printf("拼接前为：\nd=%s\ns=%s\n",d,s);

strcat(d,s);

printf("拼接后为：\nd=%s\ns=%s\n",d,s);//把d跟s拼接，输出d

system("pause");

return 0;

}

实例3 字符串的比较

#include<string.h>

#include<stdlib.h>

#include<stdio.h>

int main()

{

char string[20];//定义一个数组

char str[3][20];//定义一个二维数组

int i;

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

gets(str[i]);//输入三行数据 分别代表str[0][20] str[1][20] str[2][20]

if(strcmp(str[0],str[1])>0)//用比较函数，若一个数组比第二个大

strcpy(string,str[0]);//把str[0]赋给String

else

strcpy(string,str[1]);//把str[1]赋给string

if(strcmp(str[2],string)>0)//若str[2]大于现在的string

strcpy(string,str[2]);//将str[2]赋给string,否则string不变

printf("\nThe largest string is %s\n",string);//输出现在的string

system("pause");

return 0;

}

实例4 查找子串位置

#include<string.h>

#include<stdlib.h>

#include<stdio.h>

int main()

{

char \*s="Golden Global View";//定义字符串s

char \*l="lob";//定义字符串l

printf("string s is :%s\n",s);//打印s

printf("string l is :%s\n",l);//打印l

printf("seek result:\n");

char \*p;//创立指针

p=strstr(s,l);//在母串s中查找字串s的位置

if(p)//若找到了

printf("%s\n",p);//打印p

else

printf("Not Found!\n");

system("pause");

return 0;

}

实例5 自定义strlen函数

#include<string.h>

#include<stdlib.h>

#include<stdio.h>

int my\_strlen1(const char \*str)//递归

{

return ('\0' != \*str) ? (1+my\_strlen1(++str)):0;//当str不是0，继续递归

}

int my\_strlen2(const char \*str)

{

int count = 0;//初始长度为0

while (\*str !='\0')//当str不是0

{

count++;//长度加1

str++;//指针向后移动一位

}

return count;//返回最终的长度

}

int main()

{

char \*p;

p = (char \*)malloc(100\*sizeof(char));//动态申请容量为100的字符串指针

printf("please input your string:\n");

while(scanf("%s",p)!= EOF)//输入字符串

{

int count1 = 0,count2 = 0,count3=0;//分别用三个变量来记录三种长度测试方法得出的长度

count1 = my\_strlen1(p);

count2 = my\_strlen2(p);

count3 = strlen(p);

printf("count1=%d,count2=%d,count3=%d\n",count1,count2,count3);//输出结果

}

free(p);//释放申请的空间

return 0;

}

实例6 自定义strcat()函数

#include<stdlib.h>

#include<stdio.h>

char\* Strcat(char \*str1,char \*str2)//字符串连接函数

{

int size\_str1,size\_str2;

size\_str1=sizeof(str1);//1字符串的长度

size\_str2=sizeof(str2);//2字符串的长度

int size=size\_str1+size\_str2;//字符串总长度

char\* tempt=(char\*)malloc(sizeof(size));//申请这么多空间

char\* result=tempt;//将空间赋给result malloc:向系统申请分配指定size个字节的内存空间。返回类型是 void\* 类型。void\*表示未确定类型的指针。C,C++规定，void\* 类型可以通过类型转换强制转换为任何其它类型的指针

while(\*str1!='\0')//当字符串1没结束

{

\*tempt=\*str1;

tempt++;

str1++;

}//将字符串1赋给tempt

while(\*str2!='\0')//接着上面将字符串2赋给tempt

{

\*tempt=\*str2;

tempt++;

str2++;

}

\*tempt = '\0';//加上终止符

return result;

}

int main()

{

char \*a = "hello";

char \*b = ",everyOne!";

printf("string a = %s\n",a);

printf("string b = %s\n",b);

printf("字符串连接后变为：\n");

printf("%s\n",Strcat(a,b));//a即为字符串1，b即为字符串2

system("pause");

return 0;

}

实例7 自定义mystrcmp函数

#include<stdlib.h>

#include<stdio.h>

int mystrcmp(char \*one,char \*two);//自定义mystrcmp函数

int main()

{

char \*a="hello,world!";//a字符串

char \*b="helmet is a word.";//b字符串

printf("string a = %s\n",a);

printf("string b = %s\n",b);

if(mystrcmp(a,b))

printf("The largest is %s\n",a);

else

printf("The largest is %s\n",b);

system("pause");

return 0;

}

int mystrcmp(char \*one,char\*two)

{

while(one!=NULL && two!=NULL)//当a b字符串都未结束

{

if(\*one>\*two)//字符串内容的比较

return 1;

else if(\*one<\*two)

return 0;

else

{

one++;//地址加1

two++;

}

}

if(one!=NULL)//如果字符串a指针先为空

return 1;

else

return 0;

}

实例8 自定义mystrstr()函数

#include<stdio.h>

#include<stdlib.h>

#include<string.h>

char \*mystrstr(char \*s1,char\*s2)//定义strstr函数

{

char \*pChar;//定义指针

pChar = s2;//将s2赋给指针

int len = strlen(s2);//len为s2的长度

for( ;strchr(s1,\*pChar)!='\0';s1++)//strchr：查找字符串s1中首次出现字符的位置

{

if(strncmp(s1,pChar,len) == 0)//比较s1与pChar，len是指定要比较的字符数，s1=pChar时返回0

{

return s1;//返回s1

}

}

return NULL;//若比较到终止符号，停止循环,返回空值

}

int main()

{

char str1[20]="abcdhello!";//字符串1

char str2[20]="cd";//字符串2

char \*p = mystrstr(str1,str2);//调用mystrstr函数

printf("被查找字符串为：%s\n",str1);

printf("查找字符串为：%s\n",str2);

printf("查找结果为：%s\n",p);

system("pause");

return 0;

}

实例9 字符串中字母数字个数的统计

#include<stdio.h>

#include<stdlib.h>

int main(int argc,char \*argv[])

{

char c;//保存输入字符

int n=0,x=0;//n为字母计数，x为数字计数

printf("请输入字符串（以“\*”结束）：\n");

c=getchar();//输入一个字符

while(c!='\*')//如果输入的字符不等于\*

{

if((c>'a'&&c<'z')||(c>'A'&&c<'Z'))//该范围内的便是字母

{

n++;//字母个数+1

}

if(c>'0'&&c<'9')//该范围内的便是数字

{

x++;//数字个数加一

}

c=getchar();//继续输入字符，继续循环

}

printf("字母的个数为%d\n",n);

printf("数字的个数为%d\n",x);

system("pause");

return 0;

}

实例10 字符串中各类符号的统计

#include<stdio.h>

#include<stdlib.h>

#include<ctype.h>//该头文件用于检查传递的字符

int main(int argc,char \*argv[])

{

char c;

int f=0,z=0,s=0,p=0;//用于统计各种符号的个数

printf("请输入字符串：\n");

c=getchar();//输入字符串，启动循环

while(c!='\*')//当输入\*的时候结束

{

if(ispunct(c))//该函数用于检查输入字符是不是标点符号

{

f++;

}

if(isalpha(c)) )//该函数用于检查输入字符是不是字母

{

z++;

}

if(isdigit(c)) )//该函数用于检查输入字符是不是数字

{

s++;

}

if(isspace(c)) )//该函数用于检查输入字符是不是空格

{

p++;

}

c=getchar();//用于接受回车键

}

printf("标点的个数为%d\n",f);

printf("数字的个数为%d\n",s);

printf("字母的个数为%d\n",z);

printf("空格的个数为%d\n",p);

system("pause");

return 0;

}

实例11 字符串中单词个数的统计

#include<stdio.h>

#include<stdlib.h>

#include<string.h>

int main()

{

char str[128] = "I really love jiawei";//指定要计算的字符串

int num = 0;//用于计数

int i = 0;//用于标记字符的位置

while('\0'!= str[i])//当字符串未结束时

{

if(' ' == str[i])//实际上是数空格的个数

{++num;}//有空格便加1

++i;//字符串位置加1

}

printf("there are %d words in \"%s\"\n",num+1,str);//记住要加最后一个字符，输出str

system("pause");

return 0;

}

实例12 字符串替换函数

#include<stdio.h>

#include<stdlib.h>

#include<string.h>

char\* replace(char\* src,char\* sub,char\*dst)

{

int i=0;

int len;

char\* temp1=NULL;

char\* temp2=NULL;

char\* SRC=NULL;//用于保存源字符串

SRC = (char\*)malloc(sizeof(src));//申请一个与原串等容量的空间

strcpy(SRC,src);//将原串复制到SRC中

len=strlen(src);//计算src的长度

for(;i<=len;i++)//遍历一遍原串

{

if(!strncmp(src,sub,strlen(sub)))//如果找到子串相同的位置

{

temp1 = (char\*)malloc(i+1);//动态申请新空间

strncpy(temp1,SRC,i);//将原串指定长度赋给tempt1

temp1[i]='\0';//为字符串加上结束标志

strcat(temp1,dst);//将替换字符串连接到之前的字符串中

src=src+strlen(sub);//将剩下的字符串部分指针赋给src

strcat(temp1,src);//将剩下的字符串部分补充到新的字符串中

return temp1;//返回替换后的字符串中

}

src++;//

}

printf("NOT FOUND\n");//没有找到子字符串，则原样返回字符串即可

return src;

}

int main()

{

char \*a="abcdefyyyjklmn";//原串

char \*b="yyy";//被替换的字符串

char \*instead="ghi";//替换的字符串

char \*result=replace(a,b,instead);//进行函数调用，实现替换

printf("原串为：%s\n",a);//输出原串

printf("子串为：%s\n",b);//输出被替换的子串

printf("替换为：%s\n",instead);//输出要替换的字符串

printf("最终结果为：\n");

printf("%s\n",result);//输出替换后的新的字符串

system("pause");

system("pause");

return 0;

}

实例13 字符串的逆序输出

#include<stdio.h>

#include<stdlib.h>

void reverse(char \*s)

{

if(s[0] == '\0')//字符串结束

return;//返回

else

{

reverse(&s[1]);//递归

putchar(s[0]);

}

}

int main()

{

char s[100];

gets(s);//输入字符串

reverse(s);//调用逆序函数

printf("\n");//输入回车

system("pause");

return 0;

}

程序过程：若输入的字符串为HE\0,首先进入reverse1,通过if判断，s[0]的地址内容是否为\0,显然不为，执行else,进入reverse2,此时调用s[0]到s[1]的地址，即s[0]=E,S[1]=\0再次判断，显然也不为，进入reverse3,此时再调用s[0]到s[1]的地址，s[0]=\0,符合if条件，返回，结束reverse3,输出当前s[0]=\0,结束reverse2,输出当前s[0]=E,结束reverse1，输出s[0]=H.

实例14 字符串的删除

#include<stdio.h>

#include<stdlib.h>

#include<string.h>

char \*del(char s[],int pos,int len)//删除函数，参数为position和length

{

int i;

for(i=pos+len-1;s[i]!='\0';i++,pos++)//

{

s[pos-1]=s[i];//即把s的位置左移length位

}

s[pos-1]='\0';//最后加上结束符

return s;

}

int main()

{

char str[50];

int position;

int length;

printf("请输入一个字符串：\n");

gets(str);

printf("请输入一个位置：\n");

scanf("%d",&position);

printf("请输入删除字符串的长度：\n");

scanf("%d",&length);

printf("结果字符串为：\n%s\n",del(str,position,length));

system("pause");

return 0;

}

实例15 字符串中字符的删除

#include<stdio.h>

#include<stdlib.h>

#include<string.h>

char \*fun(char \*s,char c)//字符删除函数

{

char \*p=s;//用指针p指向字符串s的首地址

char \*ppPtr = p;//ppPtr指向字符串p的首地址

for(;\*s!='\0';s++)//如果指向的当前字符不是“\0”

{

if(\*s!=c)//如果当前字符不等于指定字符

\*p++=\*s;//将当前字符写入指针p

}

\*p = '\0';//字符串末尾添加字符串标志

return ppPtr;//返回所得字符串

}

int main()

{

char szTemp[20]="\*Hello,world";//输入字符串

printf("初始字符串为：%s\n",szTemp);//打印初始字符串

char \*pStr;

pStr=fun(szTemp,'l');//字符串删除函数，并将结果保存到pStr中

printf("要删除的字符为：l\n");

printf("最终得到的结果为：\n");

printf("str = %s\n",pStr);

system("pause");

return 0;

}

实例16 字符串中字符的排序

#include<stdio.h>

#include<stdlib.h>

#define N 10

int main()

{

char str[N];//声明字符数组变量

int i,j,t;//临时变量

printf("请输入%d个字符进行排序：\n",N);

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

scanf("%c",&str[i]);//为字符数组赋值

//冒泡排序

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

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

if(str[j]<str[i])

{

t=str[i];

str[i]=str[j];

str[j]=t;

}

printf("排序结果为：\n");

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

{

printf("%3c",str[i]);//输出字符排序后的结果

}

system("pause");

return 0;

}

实例17 字符串的循环右移

#include<stdio.h>

#include<stdlib.h>

void loopMove(char \*pStr,int steps)//字符串移位操作

{

int length=0,i=0;

char \*sub=pStr,\*str;

while (\*pStr!='\0')//当当前字符不等于字符串结束标志时

{

length++;//统计源字符串长度

pStr++;//字符串指针右移一位

}

pStr--;//字符串指针指向字符串最后一个字符

if(steps>length)//循环多于一圈情况下

steps=steps-length;//避免多次无用循环

str=(char \*)malloc(steps);//为右移多出来的字符串申请空间

for(i=0;i<steps;i++)//循环步长下多出来的字符串保存到str中

str[i]=sub[length-steps+i];//将多出来的字符保存到str字符串中

for(i=0;i<length-steps;i++)//不是多出来的字符串整体右移step位

sub[length-i-1]=sub[length-steps-i-1];//其余元素全体右移

for(i=0;i<steps;i++)//合并两部分字符串为一个字符串

sub[i]=str[i];//将str中的字符串补充到sub串的起始部分

}

int main()

{

char a[]="abcdef";//初始化字符串

printf("最初的字符串为：\n%s\n",a);

loopMove(a,2);//字符串循环右移两位

printf("循环右移两位后为：\n%s\n",a);

system("pause");

return 0;

}

实例18 单词的排序

#include"stdio.h"

#include<stdlib.h>

#include"string.h"

int main()

{

char temp[20],name[8][15]={"question","answer","sweet","glory","militant","beauty","clever","aaaaaa"};

int i,j;

printf("排序前的序列为：\n");

for(i=0;i<8;i++)

printf("%d,%s\n",i+1,name[i]);

for(i=0;i<8;i++)

for(j=0;j<8-i;j++)

if(strcmp(name[j],name[j+1])>0)

{

strcpy(temp,name[j]);

strcpy(name[j],name[j+1]);

strcpy(name[j+1],temp);//交换

}

printf("排序后的结果为：\n");

for(i=0;i<8;i++)

printf("%d,%s\n",i+1,name[i]);

system("pause");

return 0;

}

实例19 字符的插入

#include<stdio.h>

#include<stdlib.h>

#include<string.h>

int main()

{

int i,j,k,m,n;

char a[100],b[100],c;//大数组数组的声明

printf("请输入原字符串：\n");

gets(a);//输入源字符串

printf("请输入要插入的字符串：\n");//输入要插入的字符串

gets(b);

printf("请输入要插入的某个字符处：\n");

c=getchar();

m=strlen(a);//源字符串的长度

for(i=0;i<m;i++)//循环m次

{

if(a[i]==c)//一直循环到插入字符处

break;

else continue;//跳出循环

}

if(i>=m)//一直循环到最后也没找到插入字符处

{

printf("出错！\n");

return 1;//返回值非0表示程序异常结束

}

n=strlen(b);//n表示插入字符串的长度

k=m;//将源字符串的长度赋给k

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

a[j]=a[k];//将源字符串插入处之后的部分后移

for(j=i,k=0;j<n+i;j++,k++)//后移之后，将插入的字符串填补进去

a[j]=b[k];

printf("最终得到字符串为：\n");

printf("%s\n",a);//a表示a[0]的地址

system("pause");

return 0;

}

实例20 字符串的加密，解密

#include<stdio.h>

#include<stdlib.h>

#include<string.h>

int main()

{

int result=1; //result=1加密 result=2解密 return=3 退出

int i;

int count=0;

char Text[128]={'\0'}; //统计字符串长度

char Cryptograph[128]={'\0'};//存储需要加密的字符串

while(1)//加密过程

{

if(result==1)

{

printf("请输入要加密的明文：\n");

scanf("%s",&Text);

count=strlen(Text);//输入明文的长度

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

Cryptograph[i]=Text[i]+i+5;//加密后的密文转换公式即为向后推i+5个asc||码

Cryptograph[i]='\0';//加上结束符号

printf("加密后的密文是：%s \n",Cryptograph);

}

else if(result==2)

{

count=strlen(Text);

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

Text[i]=Cryptograph[i]-i-5;//与之相反的解密过程

Text[i]='\0';//加上结束符

printf("解密后的明文是：%s\n",Text);

}

else if(result==3)//退出程序

break;

else

printf("请输入正确命令符：\n");//如果输入不为1 2 3

printf("请输入： 1、加密新的明文");

printf(" 2、对刚加密的密文进行解密\n");

printf(" 3、退出系统\n");

printf("请输入命令符：\n");

scanf("%d",&result);

}

system("pause");

return 0;

}

实例21 字符串是否是回文

#include<stdio.h>

#include<stdlib.h>

#include<string.h>

bool check(char \*str);//回文函数检测函数

int main()

{

printf("请输入一个字符串：\n");

char str[80];

gets(str);//输入字符串, 和scanf()的区别在于输入的字符串是否中间有空格：对于前者，只有遇到"\n"时才停止输入，而对于后者，出现"\n"或空格都停止输入。

scanf()以Space、Enter、Tab结束一次输入，不会舍弃最后的回车符（即回车符会残留在缓冲区中）；   
getchar()以Enter结束输入，也不会舍弃最后的回车符；   
读取字符串时：  
 scanf()以Space、Enter、Tab结束一次输入   
gets()以Enter结束输入（空格不结束），接受空格，会舍弃最后的回车符！

printf("\n");

puts(str);

if(check(str))

printf("该字符串回文\n");

else

printf("该字符串不回文\n");

system("pause");

return 0;

}

bool check(char \*str)//布尔类型

{

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

char \*tempt=str;//将被检测字符串首地址赋给tempt

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

{

if(\*(tempt+i)!=\*(tempt+len-1-i))//判断条件：如果第i个字符与倒数第i个字符不相等

return false;//返回false，即为0

}

return true;//返回ture即为1

}

实例22 日期类数据的录入

#include<stdio.h>

#include<stdlib.h>

#include<string.h>

struct Date

{

int year,month,day;

};

bool check\_valid(Date date)

{

int month[12]={31,28,31,30,31,30,31,31,30,31,30,31};

if(date.year<1900||date.year>5000)//年份不在范围内

return false;

if(date.month<=0||date.month>12)//月份不在范围内

return false;

if(date.day>month[date.month-1]||date.day<=0)//日期必须在范围内

return false;

return true;

}

int main()

{

Date date;

printf("请输入日期（eg,1949-10-01）:\n");//输入一个日期

while(true)//无条件进行循环

{

scanf("%d-%d-%d",&date.year,&date.month,&date.day);//输入日期

if(check\_valid(date))//若日期输入正确

{

printf("输入正确！");

break;//跳出循环

}

else

{

printf("您输入的格式有误！请重新输入：\n");//若日期输入不正确

continue;//继续循环，跳至scanf

}

}

system("pause");

return 0;

}

实例23检查身份证号码的正确性

#include<stdio.h>

#include<stdlib.h>

#include<string.h>

#include<ctype.h>

int Chk18PaperId(const char \*sPaperId)//

{

long lSumQT =0;

int R[]={7,9,10,5,8,4,2,1,6,3,7,9,10,5,8,4,2};//加权因子

char sChecker[11]={'1','0','X','9','8','7','6','5','4','3'};//校验码

if(18!=strlen(sPaperId)) //如果检验的长度不等于18，则不正确

return -1;

int i=0;

for(i=0;i<18;i++)

{

if(!isdigit(sPaperId[i])&&!(('X'==sPaperId[i]||'x'==sPaperId[i])&&17==i))//isdigit作用为检验所有位是否为数字 如果不为数字或X,x，返回-1

{

return -1;

}

}

for(i=0;i<=16;i++)

{

lSumQT+=(sPaperId[i]-48)\*R[i];//检验校验码

}

if(sChecker[lSumQT%11]!=sPaperId[17])

{

return -1;

}

return 0;

}

int main()

{

char Personal\_ID[18];

printf("请输入一个身份证号码：\n");

gets(Personal\_ID);//输入身份证

if(!Chk18PaperId(Personal\_ID))//检验函数不等于0

printf("合法\n");

else

printf("不合法\n");

system("pause");

return 0;

}

实例24字符串的存在方式

#include<stdio.h>

#include<stdlib.h>

#include<string.h>

int main()

{

char str\_arr[20]="Hello,my friends.";

printf("输出字符数组：\n");

puts(str\_arr);//保存在字符数组中，用puts输出

char \*p\_str=str\_arr;

printf("指向指针的指针直接输出字符串：\n");

printf("%s\n",p\_str);//保存在字符型指针中，用%s输出

printf("按一个一个字符输出指向指针的指针所对应的字符串");

for(unsigned int i=0;i<strlen(str\_arr);i++)

printf("%c",\*p\_str++);//利用循环一位一位输出

printf("\n");

system("pause");

return 0;

}

实例26字符串大小写统一变换

#include<stdio.h>

#include<stdlib.h>

#include<string.h>

int main()

{

char s[]="aBcDeFgH12345;!#$";

int i;

printf("before toupper():%s\n",s);

for(i=0;i<sizeof(s);i++)

s[i]=toupper(s[i]);//将字符c转换为大写字母

printf("after toupper():%s\n",s);

for(i=0;i<sizeof(s);i++)

s[i]=tolower(s[i]); //将字符c转换为小写字母

printf("after tolower():%s\n",s);

system("pause");

return 0;

}

实例27字符串类型的转换

#include<stdio.h>

#include<stdlib.h>

#include<string.h>

int main()

{

char a[]="-100";//

char b[]="456";

int c;

c=atoi(a)+atoi(b);//将字符串转换为整型数

printf("c=%d\n",c);

char \*a2="-100.23";

char \*b2="200e-2";

float c2;

c2=atof(a2)+atof(b2);//将字符串转换为浮点数

printf("c2=%.2f\n",c2);

system("pause");

return 0;

}

实例28虚拟键码与ASCII码之间的转换

#include<stdio.h>

#include<stdlib.h>

#include<string.h>

#include<ctype.h>

int main()

{

int a = 217;

char b;

printf("before toascii():a value = %d(%c)\n",a,a);

b= toascii(a);//直接使用函数转换

printf("after toascii(): a value =%d(%c)\n",b,b);

system("pause");

return 0;

}

实例29字符串的分割

#include<stdio.h>

#include<stdlib.h>

#include<string.h>

int main()

{

char s[]="ab-cd : ef;gh :i-jkl;mnop;qrs-tu: vwx-y;z";//原来的字符串

char \*delim="-: ";

char \*p;

printf("分割出来的字符串序列有：\n");

printf("%s",strtok(s,delim));// 分解字符串为一组字符串。s为要分解的字符，delim为分隔符字符，如果传入字符串，则传入的字符串中每个字符均为分割符）

while((p=strtok(NULL,delim)))//p为分割完的字符串，只要字符串未终止，继续循环

printf("%s ",p);

printf("\n");

system("pause");

return 0;

}

实例30子字符串的查找

#include<stdio.h>

#include<stdlib.h>

#include<string.h>

int main()

{

char \*s="0123456789012345678901234567890";//输入一个字符串

printf("s=%s\n",s);

char \*p;

p=strrchr(s,'5');//strrchr的功能为查找一个字符C在另一个字符串str中末次出现的位置（即从字符串末端查找该字符串首次出现的位置），并返回从字符串中的这个位置起，一直到字符串结束的所有字符，如果未能找到指定字符，那么函数将返回NULL。

printf("查找结果为：%s\n",p);

system("pause");

return 0;

}