

字符数组与字符串

Wang Houfeng

EECS, PKU

wanghf@pku.edu.cn

内容

➤ 字符数组与字符串

- 字符与字符串的输入输出
- 几个典型的字符串函数
- 例子

字符数组和字符串

- 当数组的元素类型为 `char` 时，称为字符数组

`char s[100];`

`s` 是一个一维字符数组变量

- 两种初始化方法：

- 同一般数组一样初始化，但需用单引号表示每个元素，不能超过给定长度，不够时，后面自动补 `'\0'` 或 `0` (`'\0'` 就是 ASCII 值 `0`)；

- 用双引号初始化，引号内的元素小于定义的长度，即最后面至少留一个空位存放 `'\0'` 或 `0`；

`char str[10]="abcdefghi";` // 串中的字符数小于数组长度

在 C/C++ 中 `'\0'` 或 `0` 表示串结尾符

例 char ch[5]="Boy";

用字符串常量

B o y \0 \0

ch[0] ch[1] ch[2] ch[3] ch[4]

一般数组赋值方式：逐个赋值，不够补0

例 char ch[5]={ 'B', 'o', 'y' };

B o y \0 \0

ch[0] ch[1] ch[2] ch[3] ch[4]

可以为空

二维字符数组初始化

例 `char fruit[][7]={"Apple", "Orange",
"Grape", "Pear", "Peach"};`

fruit[0]	A	p	p	l	e	\0	\0
fruit[1]	O	r	a	n	g	e	\0
fruit[2]	G	r	a	p	e	\0	\0
fruit[3]	P	e	a	r	\0	\0	\0
fruit[4]	P	e	a	c	h	\0	\0

二维字符数组初始化

例 char diamond[5] = { {'.', '!', '*'}, {'.', '*', '!', '*'},
{ '*', '!', '!', '!', '*'}, {'!', '*', '!', '*'}, {'!', '!', '*'} };

diamond[0]	.	.	*	\0	\0
diamond[1]	.	*	.	*	\0
diamond[2]	*	.	.	.	*
diamond[3]	.	*	.	*	\0
diamond[4]	.	.	*	\0	\0

字符串

- 无字符串变量，用字符数组表示字符串
- 字符串结束标志：'\0'（切记）

例 char s[]="hello"; //以串的形式赋初始值

共5个字符，因此，字符串长度为5，但在内存占6个字节

h	e	l	l	o	\0
104	101	108	108	111	0

内存存放字符ASCII码

二个例

```
main()
{
    char a[]={'h','e','l','\0','l','o','\0'};
    cout<<a<<endl;
}
```

h	e	l	\0	l	o	\0
---	---	---	----	---	---	----

输出： ? ? ?

字符数组可整体输出，但必须表示字符串

```
int a, b ;
cin>>a>>b; // 从键盘输入 21 abc ✓ (xx, 不能转换类型)
```


例 输出一个字符串

```
#include <iostream>
main()
{  char c[10]={'I',' ','a','m',' ','a',' ','b','o','y'};
   int i;
   for(i=0;i<10;i++)
       cout<<c[i];
   cout<<endl;
   cout<< c; // (XX) 此时的c不是字符串!
}
```

0	I
1	
2	a
3	m
4	
5	a
6	
7	b
8	o
9	y

结果: **I am a boy**
 I am a boy(???)

内容

➤ 字符数组与字符串

➤ 字符与字符串的输入输出

- 几个典型的字符串函数
- 例子

常用的字符(串)输出

- 字符 (c) 输出

- C语言 (包含在 `stdio.h`, 可以省略)

- `printf("%c",c);`
 - `putchar(c);`

- C++语言

- `cout<<c;`

- 字符串 (s) 输出

- C语言 (包含在 `stdio.h`, 可以省略)

- `printf("%s",s);`
 - `puts(s);`

- C++语言

- `cout<<s;`

常用的字符输入

- 字符 (c) 输入

- C语言 (包含在 `stdio.h`, 可以省略)

- `scanf("%c", &c);`
 - `c=getchar();`

- C++语言

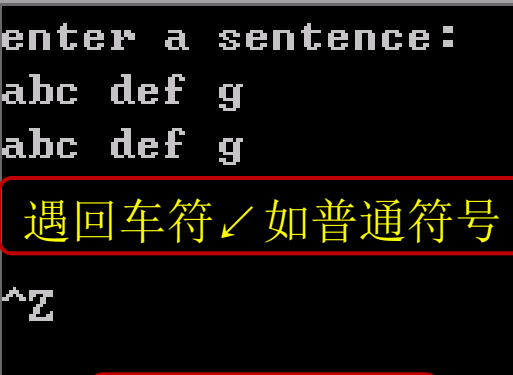
- `cin>>c;` //会自动略去 空格 和 回车符

- `cin.get(c);`
 - `c=cin.get();`

这两个函数都是遇到输入`^Z`时停止;
而 `getchar` 遇到任何符号均不自动停止, 需要条件判断

字符输入例1

```
#include <iostream>
using namespace std;
int main()
{
    char c;
    cout << "enter a sentence:" << endl;
    while ((c = cin.get()) != EOF)
        cout<<c;
    return 0;
}
```



```
enter a sentence:
abc def g
abc def g
```

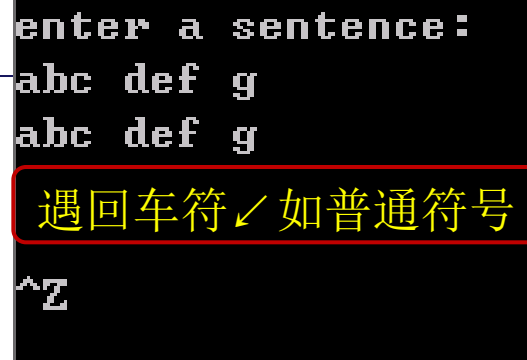
遇回车符↵如普通符号

^Z

遇到^Z停止

字符输入例2

```
#include <iostream>
using namespace std;
int main()
{
    char c;
    cout << "enter a sentence:" << endl;
    while (cin.get(c))
        cout << c;
    return 0;
}
```



```
enter a sentence:
abc def g
abc def g
^Z
```

遇回车符↵如普通符号

遇到^Z停止

//读取一个字符赋给字符变量c

字符输入例3: getchar()

```
#include <iostream>
using namespace std;
int main()
{
    char c;
    cout << "enter a sentence:" << endl;
    while (c = getchar()) //不跳过任何字符
        cout<<c;
    return 0;
}
```

```
enter a sentence:
abc def g
abc def g
```

遇回车符✓如普通符号

```
^Z
^Z
^Z
```

遇到^Z也如普通符号

常用的字符串输入 (1)

- C语言 (包含在 `stdio.h`, 可以省略)
 - `scanf("%s", s);`
- C++语言
 - `cin>>s;`
- 上述两类语句均为遇空格和回车符便终止
- 无法实现带空格的整体输入:

How are you!

例 scanf输入字符串举例

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

```
main()
```

```
{ char a[15],b[5],c[5];
```

```
scanf("%s%s%s",a,b,c);
```

替换后结果相同

```
cin>>a>>b>>c;
```

```
printf("a=%s\nb=%s\nc=%s\n",a,b,c);
```

```
scanf("%s",a);
```

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

```
}
```

scanf中%s输入时,遇空格或回车结束

H	o	w	\0																
a	r	e	\0																
y	o	u	?	\0															

- 字符串整体作为一个单元,变量即为地址。
- scanf 输入字符串时,不必用 & 表示地址

运行情况:

输入: How are you?

输出: a=How

b=are

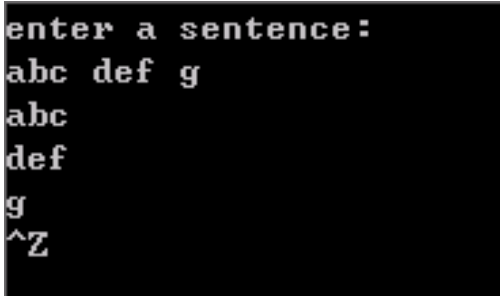
c=you?

输入: How are you?

输出: a=How

例 再看cin输入字符串

```
#include <iostream>
using namespace std;
int main()
{
    char str[10];
    cout << "enter a sentence:" << endl;
    while (cin >> str) //使用循环输入各子串
        cout << str<<endl;
    return 0;
}
```



```
enter a sentence:
abc def g
abc
def
g
^Z
```

遇到空格或回车符结束，遇^Z终止循环

例 若准备将字符串 “This is a string.”记录下来,

错误的输入语句为:

- (A) scanf("%20s",s);
- (B) for(k=0;k<17;k++)
 s[k]=getchar();
- (C) for(k=0;k<17;k++)
 scanf("%c",&s[k]);
- (D) while((c=getchar())!='\n')
 s[k++]=c;

答案: A (为什么?)

getchar实现所有符号的输入

常用的字符串输入 (2)

- C语言 (包含在 `stdio.h`, 可以省略)

- `gets(s)`;

- C++语言

- `cin.get`;

- `cin.getline`;

自动加串尾符 `'\0'`

- 上述两类语句均保留所有符号 (含空格)
- 可以实现带空格的整体输入:

How are you!

cin.get 格式

- 有3个参数的get函数

`cin.get(ch, 10, '\n');`

- 读取(10-1)个字符（包含空格），赋给字符数组 ch;
- 如果在读取10-1个字符**之前**，遇到指定的终止字符'\n'，则**提前结束**读取；**如果第3个参数没有指定，则默认为'\n'；**
- 读取成功返回非0值(真)，如失败(遇文件结束符) 则返回0值(假)。

cin.get 举例

```
#include <iostream>
using namespace std;
int main()
```

```
{
```

```
    char ch[20];
```

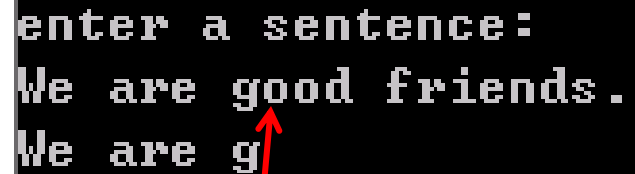
```
    cout << "enter a sentence:" << endl;
```

```
    cin.get(ch, 10, 'o'); //指定终止符为 'o'
```

```
    cout << ch << endl;
```

```
    return 0;
```

```
}
```



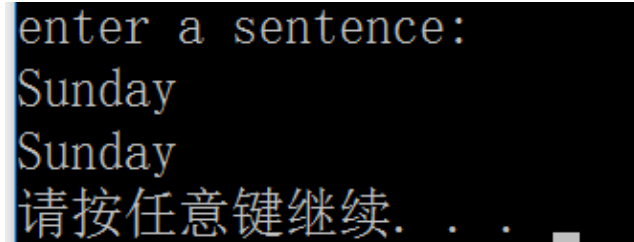
```
enter a sentence:
We are good friends.
We are g
```

改成 `cin.getline(ch, 10, 'o');` 运行结果一样

cin.get 举例（缺省终止符）

```
#include <iostream>
using namespace std;
int main()
{
```

```
    char ch[20];
    cout << "enter a sentence:" << endl;
    cin.get(ch, 10); //缺省终止符时，默认为 '\n'
    cout << ch << endl;
    return 0;
}
```

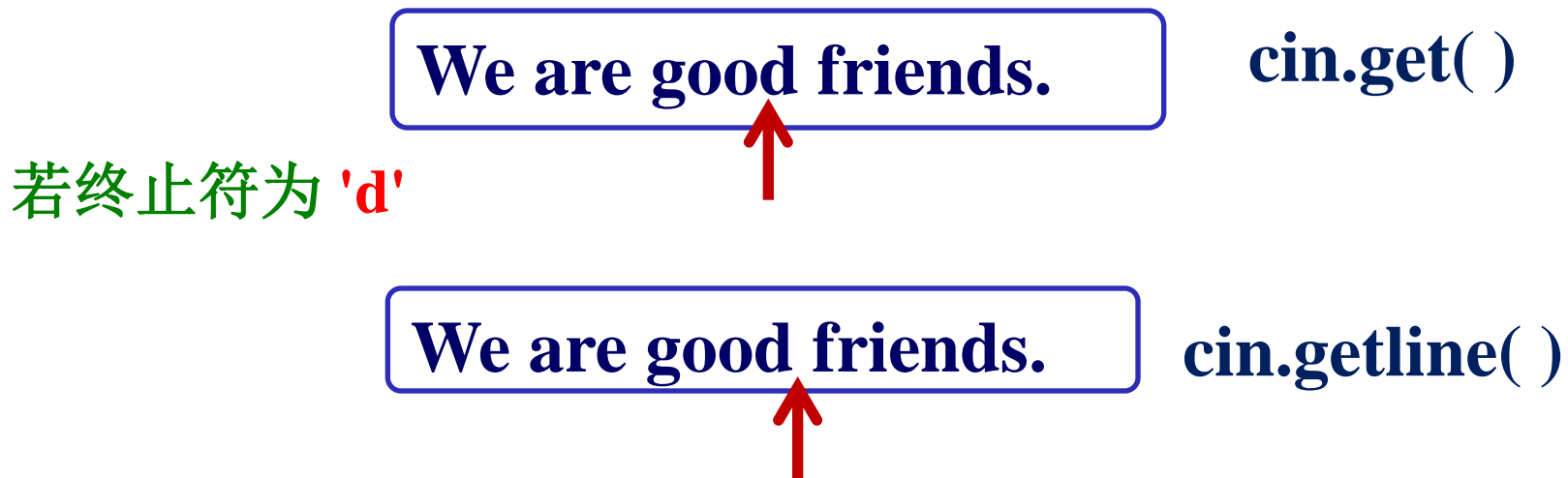


```
enter a sentence:
Sunday
Sunday
请按任意键继续. . .
```

改成 `cin.getline(ch, 10);` 运行结果一样

getline与get的区别

- getline遇到终止标志字符时结束，缓冲区指针移到终止标志字符**之后**；
- get遇到终止字符**时**停止读取，指针不移动



用cin.getline 读入字符

```
#include <iostream>
using namespace std;
int main() {
    char ch[20];
    cout<<"enter a sentence:"<<endl;
    cin>>ch;
    cout<<"The string read with cin is:"<<ch<<endl;
    cin.getline(ch, 20, '/'); //读19个字符或遇'/'结束
    cout<<"The second part is:"<<ch<<endl;
    cin.getline(ch, 20); //读19个字符或遇'/n'结束
    cout<<"The third part is:"<<ch<<endl;
    return 0;
}
```

- **getline**指针移到终止标志字符**之后**;
- **get**遇到终止字符**时**停止指针不动
- **cin** 停在空格/回车符等位置不动

如果把cin.getline换成cin.get?

程序运行情况如下:

```
enter a sentence:
I like C++./I study C++./I am happy.
The string read with cin is:I
The second part is:like C++.
The third part is:I study C++./I am h
请按任意键继续. . .
```

C++./

```
enter a sentence:
I like C++./I study C++./I am happy.
The string read with cin is:I
The second part is:Ilike C++.
The third part is:/I study C++./I am
请按任意键继续. . .
```

再看例子

```
2
abc
efg
abc
efg
请按任意键继续. . .
```

```
#include<iostream>
using namespace std;
void main()
```

```
{
    char a[10][10];
    int i, n=0;
    cin>>n;
    for(i=0;i<n;i++)
        cin.getline(a[i], 10);
    for(i=0;i<n;i++)
        cout<<a[i]<<endl;
}
```

cin遇回车符结束，指针停留在回车符位置不动

```
2
abc
efg
```

```
2
abc
abc
请按任意键继续. . .
```

2后面的回车符

```
#include<iostream>
using namespace std;
void main()
{
```

```
    char a[10][10];
    int i, n=0;
    cin>>n;
    cin.get( ); //cin.get() 跳过回车符
    for(i=0;i<n;i++)
        cin.getline(a[i],10);
    for(i=0;i<n;i++)
        cout<<a[i]<<endl;
```

```
}
```

cout 输出字符串

```
#include <iostream>
using namespace std;
int main()
{
    char a[10] = "Computer";
    cout<<a;
    return 0;
}
```



Computer

C语言常用 puts(s)

C	o	m	p	u	t	e	r	\0	\0
a[0]	a[1]	a[2]	a[3]	a[4]	a[5]	a[6]	a[7]	a[8]	a[9]

cout 输出字符串

```
#include <iostream>
using namespace std;
int main()
{
    char a[8] = { 'C', 'o', 'm', 'p', 'u', 't', 'e', 'r' };
    cout<<a;
    return 0;
}
```



Computer烫烫J程[t?

a不是字符串，不能以字符串输出

C	o	m	p	u	t	e	r
a[0]	a[1]	a[2]	a[3]	a[4]	a[5]	a[6]	a[7]

cout 输出字符串

```
#include <iostream>
using namespace std;
int main()
{
    int a[8] = { 1,2,3,4,5,6 };
    cout<<a;
    return 0;
}
```



0040FE24

地址

内容

- 字符数组与字符串
- 字符与字符串的输入输出
- 几个典型的字符串函数
- 例子

几个常用的字符串函数

- 包含在头文件 `string.h` 或 `cstring` 头文件：

`#include < string >`

◆ 字符串拷贝函数 `strcpy`

格式：`strcpy`(字符数组1,字符串2)

功能：将字符串2，拷贝到字符数组1中去

说明：① 字符数组1必须足够大

② 拷贝时 `'\0'` 一同拷贝

③ 不能使用赋值语句为一个字符数组赋值

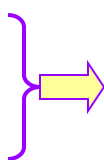
例 `char str1[20],str2[20];`

`str1={"Hello!"};`

`str2=str1;`

(×)

(×)



`strcpy(str1, "Hello!")`

`strcpy(str2,str1)`

◆ 字符串连接函数strcat

格式： **strcat**(字符数组1,字符数组2)

功能：把字符数组2连到字符数组1后面

返回值：返回字符数组1的首地址

说明：①字符数组1必须足够大

②连接前,两串均以 '\0' 结束;连接后,串1的'\0'取消,
新串最后加'\0'

例 strcpy与strcat举例

```
#include <string>
#include<iostream>
using namespace std;
int main()
{ char destination[25];
  char blank[] = " ", p[] = "Programming",
    DoubleC[] = " C++ ";
  strcpy(destination, DoubleC);
  strcat(destination, blank);
  strcat(destination, p);
  cout<<destination<<endl;
  return 0;
}
```

输出结果:

C++ Programming

◆ 字符串比较函数strcmp

格式：strcmp(字符串1,字符串2)

功能：比较两个字符串

比较规则：对两串从左向右逐个字符比较（ASCII码），直到遇到不同字符或'\0'为止

返回值：返回int型整数，a. 若字符串1 < 字符串2，返回-1

b. 若字符串1 > 字符串2，返回 1

c. 若字符串1 == 字符串2，返零

说明：字符串比较不能用“==”，必须用strcmp

◆ 字符串长度函数strlen

格式：strlen(字符数组)

功能：计算字符串长度

返回值：返回字符串实际长度，不包括'\0'在内

例 对于以下字符串，strlen(s)的值为多少？

(1) char s[10]={'A', '\0', 'B', 'C', '\0', 'D'};

(2) char s[]= "\t\v\\0will\n";

(3) char s[]= "\x69\0\n";

举 例

```
#include <string>
#include<iostream>
using namespace std;
int main()
{  char str1[] = "Hello!", str2[] = "How are you? ",str[20];
   int len1,len2,len3;
   len1=strlen(str1);    len2=strlen(str2);
   if(strcmp(str1, str2)>0)
   {  strcpy(str,str1);   strcat(str,str2);  }
   else if (strcmp(str1, str2)<0)
   {  strcpy(str,str2);   strcat(str,str1);  }
   else  strcpy(str,str1);
   len3=strlen(str);
   cout<<str<<endl;
   cout<<"Len1="<<len1<<","Len2="<<len2<<","Len3="<<len3<<endl;
   return 0;
}
```

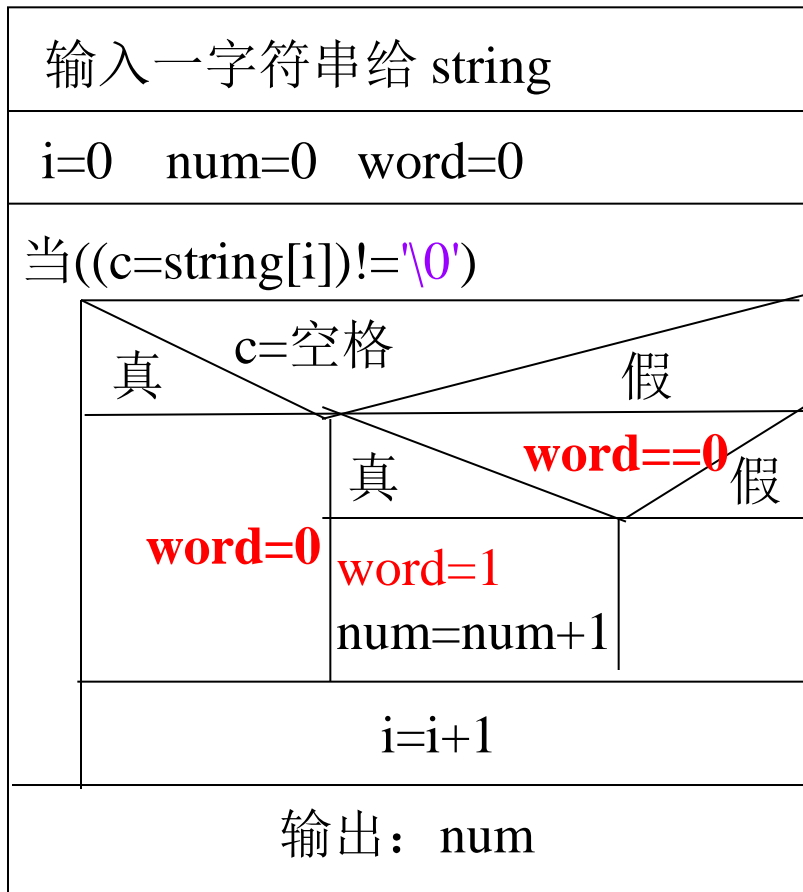
输出结果:

How are you?Hello!
Len1=6,Len2=12,Len3=18

内容

- 字符数组与字符串
- 字符与字符串的输入输出
- 几个典型的字符串函数
- 例子

题1：输入一行字符，统计有多少单词



```
#include <stdio.h>
#include<iostream>
using namespace std;
int main()
{ char string[81];
  int i,num=0,word=0;
  char c;
  while((c=cin.get())!='\n')
  gets(string);
  for(i=0;(c=string[i])!='\0';i++)
    if(c== ' ') word=0;
    else if(word==0)
    { word=1; num++; }
  cout<<"The number are "<<num;
  return 0;
}
```

题2：字符串加密

- 输入一个字符串，把每个字符变成它后续字符，如果是'Z'或者'z'，则变成'A'或'a'。空格则不变。然后将变换后的字符串输出；
- 要求能够接受连续输入；

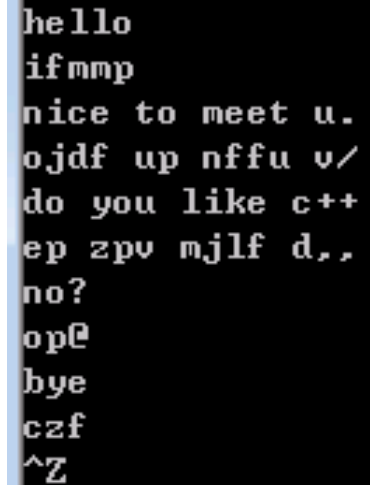
```
hello
ifmmp
nice to meet u.
ojdf up nffu v/
do you like c++
ep zpv mjlf d,,
no?
op@
bye
czf
^Z
Press any key to continue
```

字符串加密

- 思路：
 - 读入字符串（以什么方式输入？）
 - 从字符头到尾循环：
 - 是'Z'则直接赋值'A'，跳过以下步骤
 - 是'z'则直接赋值'a'，跳过以下步骤
 - 空格不做处理，跳过以下步骤
 - 其他字符++
 - 输出新字符串；

字符串加密

```
#include <iostream>
using namespace std;
int main()
{
    char str[200];
    while (cin.getline(str, 200))
    {
        for (int i = 0; str[i] != '\0'; i++)
        {
            if (str[i] == 'Z')
            {
                str[i] = 'A'; continue;}
            if (str[i] == 'z')
            {
                str[i] = 'a'; continue;}
            if (str[i] == ' ') continue;
            str[i]++; //其余情况下增1
        }
        cout << str << endl;
    }
    return 0;
}
```



```
hello
ifmmp
nice to meet u.
ojdf up nffu v/
do you like c++
ep zpv mjlf d,,
no?
op@
bye
czf
^Z
```


题3：十六进制转十进制

- 以字符串的形式输入一个十六进制数（不超过5位），将其转换成十进制数输出
- 分析：
 - 检查是不是合法的16进制数（仅考虑大写）
 -0...9.....A...F.....
 - 如果不合法可以重新输入, 合法输入则退出，否则循环。do { } while(1)
 - 将一个字符变成相应的数字
 - **decimal = hex[i] - 'A' + 10;**
 - 将字符串变成数字
 - **number = number * 16 + decimal;**

```

int main()
{
    char hex[6];
    int i, number = 0, decimal;
    cout << "please input a hex number: ";
    do // 获取一个合法的16进制字符串
    {
        cin.getline(hex,6);
        for(i = 0;hex[i] != '\0';i++) //判断合法字符
        {
            if ((hex[i] < '0' || hex[i] > '9') && (hex[i] < 'A' || hex[i] > 'F'))
                break; //跳出循环
        }
        if(hex[i] == '\0') //如果字符串正常结束
            break;
        else
            continue;
    }while(1);

    for(i = 0; hex[i] != '\0'; i++)
    {
        if (hex[i] >= '0' && hex[i] <= '9') // 单个字符变成数字
            decimal = hex[i] - '0';
        else
            decimal = hex[i] - 'A' + 10;
        number = number * 16 + decimal; // 按位求和
    }
    cout << hex << number << endl;
    return 0;
}

```

题4：字符串匹配

- 问题描述
 - 输入两个字符串a,b, 判断a是否是b的子串, 如果是, a在b中出现了几次?
 - 例如: a = “aba”, b = “ababab”, 则a在b中出现了2次
 - 再如: a = “aaa”, b = “aaaaaa”, 则a在b中出现了4次, 而不是2次
- 分析: lb, la分别是两个串的长度
 - 需要两重循环, 外循环从0到lb-la结束, 内循环从0到la结束

```
for(i = 0; i <= lb - la; i++)  
    for(j = 0; j < la; j++)
```
 - 内循环谁和谁比较? a[j]和b[i+j]
 - 设一标志:
 - 出现不相同, 标志为假, 非正常退出内循环
 - 正常退出, 为真, 说明相同, 是子串, 计数

```
#include <cstring> // cstring, string, string.h
```

```
int main()
```

```
{
```

```
    char a[100], b[100];
```

```
    int i, j, flag, la, lb, count = 0;
```

```
    cin >> a; cin >> b;
```

```
    la = strlen(a); lb = strlen(b); //函数返回的字符串长度不含\0
```

```
    for(i = 0; i <= lb - la; i++) //最后要留出子串的长度la
```

```
    {
```

```
        flag = 1; //每次都要设定一个标记
```

```
        for(j = 0; j < la; j++) //到子串的最后一个字符，逐个比较
```

```
            if( a[j] != b[ i + j ] ) //每次比较都从子串的头开始j=0
```

```
                { flag = 0; break; } //有1位不相等，则不必要再作比较
```

```
        if( flag ) count++;
```

```
    }
```

```
    if (count==0) cout << "a不是b的子串"<<endl;
```

```
    else cout<<"a在b中出现了"<<count<<"次"<<endl;
```

```
    return 0;
```

```
}
```

如果题目是：

a = “aaa”, b = “aaaaaa”, 则a在b中出现2次
该如何修改？

题5： 日历问题

- 问题描述

- 在我们现在使用的日历中, 闰年被定义为能被4整除的年份, 但是能被100整除而不能被400 整除的年是例外, 它们不是闰年
 - 例如: 1700, 1800, 1900和2100不是闰年, 而1600, 2000 和2400是闰年
- 给定从公元2000 年1月1日开始逝去的天数, 请编写程序给出这一天是哪年哪月哪日星期几

日历问题

- 输入输出要求

- 输入一个正整数，表示从2000年1月1日开始已经过去的天数。
- 对输入的每个天数，输出一行，该行包含对应的日期和星期几。格式为：

"YYYY-MM-DD DayOfWeek "

其中 "DayOfWeek" 必须是：Sunday, Monday, Tuesday, Wednesday, Thursday, Friday and Saturday。

- 输入最后一行是-1，不必处理。可以假设结果的年份不会超过9999。

日历问题

- 问题分解

- 如何判定闰年？
- 如何确定给出的一天是哪一年？那一月？
- 如何判定那一天是星期几？
 - 已知2000.1.1是周六
- 如何打印输出？

- 基本思路

- ◆ 从总天数中减去每年的天数；
 - 从2000年开始，闰年天数不同
- ◆ 然后减去每月的天数；

```

char week[7][10] = {"Saturday", "Sunday", "Monday", "Tuesday",
                    "Wednesday", "Thursday", "Friday"};
int year[2] = {365, 366}, days, i, j, leap_year;
int month[2][12] = { {31, 28, 31, 30, 31, 30, 31, 31, 30, 31, 30, 31},
                     {31, 29, 31, 30, 31, 30, 31, 31, 30, 31, 30, 31}};
while ((cin >> days) && days != -1) //输入正常天数
{
    dayofweek = days % 7; i = 2000; //从2000年按年往后计算
    leap_year = (i % 4 == 0 && i % 100 != 0 || i % 400 == 0);
    while(days >= year[leap_year]) //当前剩余天数比一年长
    {
        days -= year[leap_year]; i++;
        leap_year = (i % 4 == 0 && i % 100 != 0 || i % 400 == 0);
    }
    j = 0;
    while(days >= month[leap_year][j]) //当剩余天数比一月长
    {
        days -= month[leap_year][j];
        j++;
    }
    cout << i << "-" << j + 1 << "-" << days + 1 << " " << week[dayofweek];
}

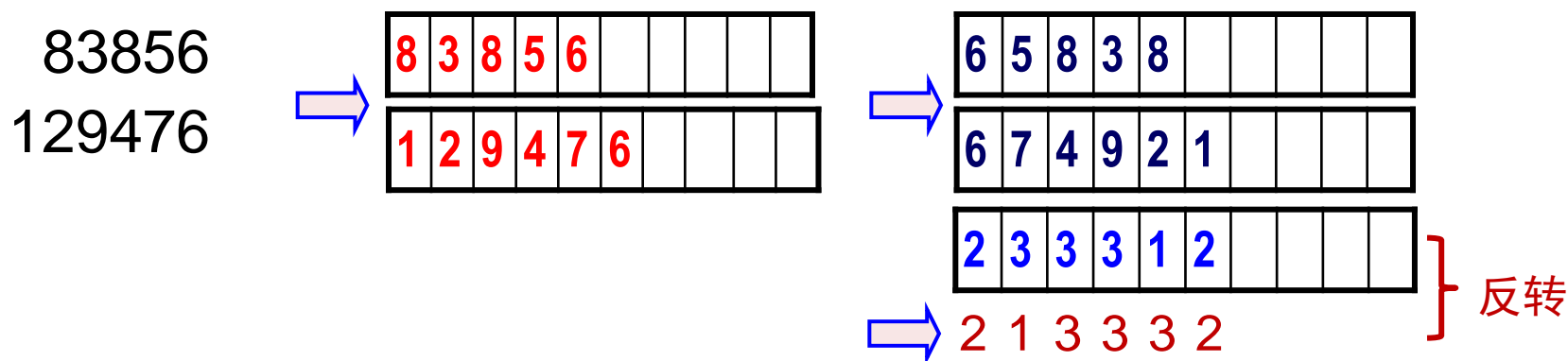
```


题6：大整数加法

- 问题描述
 - 请编写一个程序帮助统计局完成以下计算任务：
从键盘输入两个正整数 m 和 n （根据统计需要， m 和 n 最多可以是200位十进制正整数），计算并 m 和 n 的和，并打印输出。

大整数加法计算过程

- 计算 $83856 + 129476$
 - 解决输入的问题：利用字符数组接收输入；
 - 为了进行计算：把字符数组转换成整数数组，每个元素与字符数组中的每个字符相对应；
 - 转换过程中可以顺便更换一下摆放顺序，以便符合计算习惯；
 - 按照规则对位计算（个位对齐），用数组元素操作每一位；
 - 把操作结果按照“先高位再低位”的顺序输出出来



```
int main()
{
    const int MAX_LEN = 200;
    unsigned an1[MAX_LEN]; unsigned an2[MAX_LEN];
    char seLine1[MAX_LEN + 1]; char seLine2[MAX_LEN + 1];
    cout << "please input two integers" << endl;
    cin.getline(seLine1, MAX_LEN + 1);
    cin.getline(seLine2, MAX_LEN + 1);
    int nLen1 = strlen(seLine1);
    int nLen2 = strlen(seLine2);
    memset(an1, 0, sizeof(an1));
    memset(an2, 0, sizeof(an2));
```

memset(void *s, int c, unsigned n)

**作用：将已开辟内存空间 s 的首
n 个字节的值设为值 c。**

例：memset(buffer, '*', strlen(buffer));

```

int i, j=0;
for (i = nLen1-1; i>=0; i--)
    an1[j++] = seLine1[i] - '0'; //将字符数组变成整数数组，并倒置
j=0;
for (i = nLen2-1; i>=0; i--)
    an2[j++] = seLine2[i] - '0'; //高位补0
for (i = 0; i < MAX_LEN; i++)
{
    an1[i] += an2[i];
    if(an1[i] >= 10) //需要进位情况
    {
        an1[i] -= 10;
        an1[i + 1]++; // i++?
    }
}
i = MAX_LEN - 1;
while(an1[i]==0) i--; //找到第一个不为0的位
for(; i >= 0; i--)
    cout << an1[i]; //输出每一位数
cout << endl;
}

```

83856
129476



8	3	8	5	6					
---	---	---	---	---	--	--	--	--	--

1	2	9	4	7	6					
---	---	---	---	---	---	--	--	--	--	--



6	5	8	3	8					
---	---	---	---	---	--	--	--	--	--

6	7	4	9	2	1					
---	---	---	---	---	---	--	--	--	--	--



2	3	3	3	1	2					
---	---	---	---	---	---	--	--	--	--	--

6	7	4	9	2	1					
---	---	---	---	---	---	--	--	--	--	--



213332

Memset函数

- #include <memory>
- 这个函数不管数组是什么类型，都是以**字节为单位**处理。所以如果memset(buf,1, size)的话，它将buf中的所有字节都赋值为1，就算buf本来是int类型的时候也是这样。

```
int buf[10];  
memset(buf, 1, sizeof(buf));  
for (int i = 0; i < 10; i++)  
    cout << buf[i] << " ";
```

```
16843009 16843009 16843009 16843009 16843009  
16843009 16843009 16843009 16843009 16843009
```

```
00000001 00000001 00000001 00000001
```

每4个字节一个整数，因此，按整数输出的话，就是16843009。

使用这个函数要注意，常用于清0，其它赋值需慎用