<https://www.cnblogs.com/sddai/p/5774121.html>

在C/C++语言中没有专门的字符串变量，通常用字符数组来存放字符串。字符串是以“\0”作为结束符。C/C++提供了丰富的字符串处理函数，下面列出了几个最常用的函数。

  ● 字符串输出函数puts。

  ● 字符串输出函数gets。

  ● 字符串连接函数strcat。

  ● 字符串复制函数strcpy。

  ● 测字符串长度函数strlen。

字符串是面试的重点考查部分的相关知识，通过考查字符串的相关知识可以考察程序员的编程规范以及编程习惯。并且其中包括了许多知识点，例如内存越界、指针与数组操作等。许多公司在面试时会要求应聘者写一段复制字符串或字符串子串操作的程序。本章列举了一些与字符串相关的面试题，有些题目要求较高的编程技巧。

6.1  数字与字符串的转化

应聘时经常出现数字与字符串之间转化的问题，面试官通过这类题目来考察应聘者能力，例如是否熟悉常用的库函数，是否了解ASCII码以及是否了解字符串的存储格式等。

6.1.1  数字转化为字符串

面试例题1：使用库函数将数字转换为字符串。

考点：C库函数中数字转换为字符串的使用。

出现频率：★★★

解析

C语言提供了几个标准库函数，可以将任意类型(整型、长整型、浮点型等)的数字转换为字符串，下面列举了各函数的方法及其说明。

  ● itoa()：将整型值转换为字符串。

  ● ltoa()：将长整型值转换为字符串。

  ● ultoa()：将无符号长整型值转换为字符串。

  ● gcvt()：将浮点型数转换为字符串，取四舍五入。

  ● ecvt()：将双精度浮点型值转换为字符串，转换结果中不包含十进制小数点。

  ● fcvt()：指定位数为转换精度，其余同ecvt()。

还可以使用sprintf系列函数把数字转换成字符串，其比itoa()系列函数运行速度慢。下列程序演示了如何使用itoa()函数和gcvt()函数：

1    # include <stdio.h>

2    # include <stdlib.h>

3

4    int main ()

5    {

6        int num\_int = 435;

7        double num\_double = 435.10f;

8        char str\_int[30];

9        char str\_double[30];

10

11       itoa(num\_int, str\_int, 10);  //把整数num\_int转成字符串str\_int

12       gcvt(num\_double, 8, str\_double);  //把浮点数num\_double转成字符串str\_double

13

14       printf("str\_int: %s\n", str\_int);

15       printf("str\_double: %s\n", str\_double);

16

17       return 0;

18   }

程序输出结果：

1    str\_int: 435

2    str\_double: 435.10001

  ● 代码第11行中的参数10表示按十进制类型进行转换，转换后的结果是“435”，如果按二进制类型进行转换，则结果为“1101110011”。

  ● 代码第12行中的参数8表示精确位数，这里得到的结果是“435.10001”。

答案

可以使用atoi系列函数把数字转换成字符串。

面试例题2：不使用库函数将整数转换为字符串。

考点：数字转换为字符串，理解相关ASCII码。

出现频率：★★★★

解析

如果不使用atoi或sprintf等库函数，可以通过把整数的各位上的数字加“0”转换成char类型并存到字符数组中。但是要注意，需要采用字符串逆序的方法。如以下程序所示：

1    #include <iostream>

2    using namespace std;

3

4    void int2str(int n, char \*str)

5    {

6        char buf[10] = "";

7        int i = 0;

8        int len = 0;

9        int temp = n < 0 ? -n: n;  // temp为n的绝对值

10

11       if (str == NULL)

12       {

13           return;

14       }

15       while(temp)

16       {

17           buf[i++] = (temp % 10) + '0';  //把temp的每一位上的数存入buf

18           temp = temp / 10;

19       }

20

21       len = n < 0 ? ++i: i;  //如果n是负数，则多需要一位来存储负号

22       str[i] = 0;            //末尾是结束符0

23       while(1)

24       {

25           i--;

26           if (buf[len-i-1] ==0)

27           {

28               break;

29           }

30           str[i] = buf[len-i-1];  //把buf数组里的字符拷到字符串

31       }

32       if (i == 0 )

33       {

34           str[i] = '-';          //如果是负数，添加一个负号

35       }

36   }

37

38   int main()

39   {

40       int nNum;

41       char p[10];

42

43       cout << "Please input an integer:";

44       cin >> nNum;

45       cout << "output: " ;

46       int2str(nNum, p);        //整型转换成字符串

47       cout<< p << endl;

48

49       return 0;

50   }

程序中的int2str函数完成了int类型到字符串类型的转换。在代码第46行对int2str函数做了测试。程序的执行结果如下所示：

Please input an integer: 1234

Output: 1234

如果输入的是个负数，程序执行结果如下所示：

Please input an integer: -1234

Output: -1234

接下来对int2str函数的实现进行分析。

  ● 代码第9行，把参数n的绝对值赋给temp，以后在计算各个位的整数时用temp，这样保证在负数情况下取余不会出现问题。

  ● 代码第11～第14行判断str的有效性，str不为NULL。

  ● 代码第15～第19行的while循环中，将n的各个位存放到局部数组buf中，存放的顺序与整数顺序相反。例如n为整数123 456，while循环结束后buf应为“654 321”。

  ● 代码第21行计算转换后字符串的长度len，如果是负数，长度应该再加1。

  ● 代码第22～第31行把数组buf中的非0元素逆向复制到参数str指向的内存中，如果n是负数，则str指向的第一个内存存放负号。

6.1.2  字符串转化为数字

面试例题3：使用库函数将字符串转换为数字。

考点：C库函数中字符串转换为数字的使用。

出现频率：★★★★

解析

与上节数字转换为字符串类似，C/C++语言提供了几个标准库函数，可以将字符串转换为任意类型(整型、长整型、浮点型等)。以下列举了各函数的方法及其说明。

  ● atof()：将字符串转换为双精度浮点型值。

  ● atoi()：将字符串转换为整型值。

  ● atol()：将字符串转换为长整型值。

  ● strtod()：将字符串转换为双精度浮点型值，并报告不能被转换的所有剩余数字。

  ● strtol()：将字符串转换为长整值，并报告不能被转换的所有剩余数字。

  ● strtoul()：将字符串转换为无符号长整型值，并报告不能被转换的所有剩余数字。

以下程序演示如何使用atoi ()函数和atof ()函数。

1    # include <stdio.h>

2    # include <stdlib.h>

3

4    int main ()

5    {

6        int num\_int;

7        double num\_double;

8        char str\_int[30] = "435";         //将要被转换为整型的字符串

9        char str\_double[30] = "436.55";  //将要被转换为浮点型的字符串

10

11       num\_int = atoi(str\_int);          //转换为整型值

12       num\_double = atof(str\_double);  //转换为浮点型值

13

14       printf("num\_int: %d\n", num\_int);

15       printf("num\_double: %lf\n", num\_double);

16

17       return 0;

18   }

输出结果：

num\_int: 435

num\_double: 436.550000

面试例题4：不使用库函数将字符串转换为数字。

考点：字符串转换为数字时，对相关ASCII码的理解。

出现频率：★★★★

解析

程序代码如下：

1    #include <iostream>

2    using namespace std;

3

4    int str2int(const char \*str)

5    {

6        int temp = 0;

7        const char \*ptr = str;  //ptr保存str字符串开头

8

9        if (\*str == '-' || \*str == '+')  //如果第一个字符是正负号，

10       {                      //则移到下一个字符

11           str++;

12       }

13       while(\*str != 0)

14       {

15           if ((\*str < '0') || (\*str > '9'))  //如果当前字符不是数字

16           {                       //则退出循环

17               break;

18           }

19           temp = temp \* 10 + (\*str - '0'); //如果当前字符是数字则计算数值

20           str++;      //移到下一个字符

21       }

22       if (\*ptr == '-')     //如果字符串是以“-”开头，则转换成其相反数

23       {

24           temp = -temp;

25       }

26

27       return temp;

28   }

29

30   int main()

31   {

32       int n = 0;

33       char p[10] = "";

34

35       cin.getline(p, 20);   //从终端获取一个字符串

36       n = str2int(p);      //把字符串转换成整型数

37

38       cout << n << endl;

39

40       return 0;

41   }

程序执行结果：

输入：1234

输出：1234

输入：-1234

输出：-1234

输入：+1234

输出：1234

程序中的str2int函数作用是将字符串转换成整数。这个函数的转换过程与例题2中的int2str函数相比更加简单，它只需要做一次while循环（代码第13行）就能把数值大小计算出来，如果结果是负数，就加一个负号。

# [C++字符串,数字相互转换](http://www.cnblogs.com/fily/articles/1238004.html)

**一.将CString转为CTime的几种方法**

CString   timestr   =   "2000年04月05日";    
  int   a,b,c   ;    
  sscanf(timestr.GetBuffer(timestr.GetLength()),"%d年%d月%d日",&a,&b,&c);    
  CTime   time(a,b,c,0,0,0);      
  
  
--------or - ---------------------  
  
 CString   s("2001-8-29   19:06:23");    
  int   nYear,   nMonth,   nDate,   nHour,   nMin,   nSec;    
  sscanf(s,   "%d-%d-%d   %d:%d:%d",   &nYear,   &nMonth,   &nDate,   &nHour,   &nMin,   &nSec);    
  CTime   t(nYear,   nMonth,   nDate,   nHour,   nMin,   nSec);  
  
---- or ------------------------  
CString   timestr   =   "2000年04月05日";    
  int   year,month,day;    
  BYTE   tt[5];    
  //get   year    
  memset(tt,   0,   sizeof(tt));    
  tt[0]   =   timestr[0];    
  tt[1]   =   timestr[1];    
  tt[2]   =   timestr[2];    
  tt[3]   =   timestr[3];    
  year=   atoi((char   \*)tt);    
     
  //get   month    
  memset(tt,   0,   sizeof(tt));    
  tt[0]   =   timestr[6];    
  tt[1]   =   timestr[7];    
  month   =   atoi((char   \*)tt);    
     
  //get   day    
  memset(tt,   0,   sizeof(tt));    
  tt[0]   =   timestr[10];    
  tt[1]   =   timestr[11];    
     
  CTime   time(year,month,day,0,0,0);

从上面来看,很明显使用sscanf()函数的优势.

**二.将CTIme转换为CString的方法:**

CTime  tmSCan = CTime::GetCurrentTime();

CString szTime = tmScan.Format("'%Y-%m-%d %H:%M:%S'");

这样得到的日期时间字符串就是以"2006-11-27 23:30:59"的格式.这是不是很方便呢?

 //取得CTime中的日期  
 CString cstrDate = tmScan.Format("%Y-%m-%d");

 //取得CTime中的时间  
 CString cstrTime = tmScan.Format("%H:%M-%S");

          sprintf还有个不错的表妹：strftime，专门用于格式化时间字符串的，用法跟她表哥很像，也是一大堆格式控制符，只是毕竟小姑娘家心细，她还要调用者指定缓冲区的最大长度，可能是为了在出现问题时可以推卸责任吧。这里举个例子：

 更多更好的sprintf()函数说明参考:《spirntf，你知道多少？》

<http://blog.csdn.net/steedhorse/archive/2005/03/25/330206.aspx>

time\_t t = time(0);

      //产生"YYYY-MM-DD hh:mm:ss"格式的字符串。

char s[32];

strftime(s, sizeof(s), "%Y-%m-%d %H:%M:%S", localtime(&t));

sprintf在MFC中也能找到他的知音：CString::Format，strftime在MFC中自然也有她的同道：CTime::Format，这一对由于从面向对象哪里得到了赞助，用以写出的代码更觉优雅。

三,字符串转换为数值类型

将字符串"20.0E6"转换为数字

1,sscanf("20.0e5","%d",&x);

2,atof("20.0E6");

许多人用atoi(), atof() 和这个“家族”中的其它函数. 它们方便应用，但是有一个重要的缺点：  
在转换失败和转换字符串"0"时都返回0， 这样使得一致性错误检查变得几乎不可能。 为了完整性我们给出了小段代码：

代码:  
--------------------------------------------------------------------------------  
   const char\* str\_int = "777";  
   const char\* str\_float = "333.3";  
   int i = atoi(str\_int);  
   float f = atof(str\_float);  
--------------------------------------------------------------------------------

一个更好的办法：

更有一点复杂， 更遗一致的办法是利用sscanf()

代码:  
--------------------------------------------------------------------------------  
   const char\* str\_int = "777";  
   const char\* str\_float = "333.3";  
   int i;  
   float f;  
   if(EOF == sscanf(str\_int, "%d", &i)){  
      //错误  
   }  
   if(EOF == sscanf(str\_float, "%f", &f)){  
      //错误  
   }  
--------------------------------------------------------------------------------

Since sscanf() takes a const char\* parameter, you can directly use a CString with it:  
因为sscanf()用const char\* 作为参数， 所以你可以直接用CString作参数：

代码:  
--------------------------------------------------------------------------------  
   CString str\_int("777");  
   if(EOF == sscanf(str\_int, "%d", &i)){  
      //error  
   }  
--------------------------------------------------------------------------------

小心格式描述符（如本例中的"%d")。 sscanf()没有办法检查格式描述符与传递变量的类型匹配与否。如果不匹配你将得到不可预期的结果。 同样注意sscanf()可以一次从字符串中提取一个或多个数值。 详细信息请查阅MSDN。

C++ 方法

如下的例子展示了利用标准C++类的来完成这个任务的模板函数

代码:  
--------------------------------------------------------------------------------  
#include <string>  
#include <sstream>  
#include <iostream>

template <class T>  
bool from\_string(T &t,  
                 const std::string &s,  
                 std::ios\_base & (\*f)(std::ios\_base&))  
{  
   std::istringstream iss(s);  
   return !(iss>>f>>t).fail();  
}

int main()  
{  
   int i;  
   float f;  
   // from\_string()的第三个参数应为如下中的一个  
   // one of std::hex, std::dec 或 std::oct  
   if(from\_string<int>(i, std::string("ff"), std::hex)){  
      std::cout<<i<<std::endl;  
   }  
   else{  
      std::cout<<"from\_string failed"<<std::endl;  
   }  
   if(from\_string<float>(f,  
                               std::string("123.456"),  
                               std::dec))  
   {  
      std::cout<<f<<std::endl;  
   }  
   else{  
      std::cout<<"from\_string failed"<<std::endl;  
   }  
   return 0;  
}

四, int char \* float and CString Covernt

**1。 int <->CString**

1) int ->CString

int n = 1;

CString str;

str.Format("%d",n);

2) CString->int

CString str = "1";

int n = atoi(str.GetBuffer(0));

**2. char\* 与CString**

1)char\*->CString

char sz[128];

CString str;

str.Format("%s",sz);

2) CString -> char\*

CString str;

//int nLength = str.GetLength();

char\* sz = str.GetBuffer(0);

**3. float<->CString**

1）float->CString

float f = 0.0;

CString str;

str.Format("%f",f);

2) CString->float

CString str = "0.0";

float f = atof(str.GetBuffer(0));