[**[C++]字符串处理方法(STL与C风格)**](https://www.cnblogs.com/iyjhabc/archive/2013/05/10/3070789.html)

STL string

迭代器 ： string::iterator it; it = str.begin();  
1、初始化-把c字符串转换成string  
char s[]="abc e df";  
string str=s;  
string str(5,'x');//初始化为5个x

2、查找  
从前开始搜索:str.find(目标子串,开始搜索位置)  
从后开始搜索:str.rfind  
如查找不到目标，则返回string::npos

3、替换  
str.replace(开始下标,覆盖长度,替代字符串，可加<使用代替字符串长度>)  
当覆盖长度填为0时，其实就是插入操作。

4、删除  
str.erase(开始下标，删除长度)

5、插入  
str.insert(开始下标，插入字符串)

6、比较  
str.compare(与str对比的字符串)  
-1为str小，1为str大，0为相等。如果前段相等短的小。

7、返回c风格字符串  
str.c\_str();//返回值是const char\*

8、分割  
string substr (开始下标,截取长度);

c风格字符串  
1、分割

char \*p=strtok(s,d);//第一次得到结果，s为目标串，d为tok

while(p){

cout<<p<<endl;

p=strtok(NULL,d);//得到后续结果，目标串用NULL代替

}

2、比较  
strcmp(s1,s2); //-1为s1小，1为s2小，0为相等。如果前段相等短的小。

3、连接(加到目标串尾部)  
char \*strcat(char \*dest,char \*src);  
把src所指字符串添加到dest结尾处(覆盖dest结尾处的'\0')并添加'\0'。  
说明：src和dest所指内存区域不可以重叠且dest必须有足够的空间来容纳src的字符   
串。

4、复制  
char \*strcpy(char \*dest,char \*src); strcpy(str,s.c\_str());  
把src的字符串连\0复制到dest，必须保证dest有足够空间。

5、长度  
unsigned int strlen(char \*s);  
计算字符串s的(unsigned int型）长度，不包括结束符\0。

6、字符串与数字的转换

itoa(int i,char \*p result,进制)，一般进制填10，表示十进制。

任意的小数整数转字符串，都可以用sprintf(result,格式"%d",i)

对于字符串转数字，atoi和atof使用就比较简单了，直接传入需要转换的数字即可

string特性描述

可用下列函数来获得string的一些特性：

int capacity()const; //返回当前容量（即string中不必增加内存即可存放的元素个数）

int max\_size()const; //返回string对象中可存放的最大字符串的长度

int size()const; //返回当前字符串的大小

int length()const; //返回当前字符串的长度

bool empty()const; //当前字符串是否为空

void resize(int len,char c); //把字符串当前大小置为len，多去少补，多出的字符c填充不足的部分

string的查找

由于查找是使用最为频繁的功能之一，string提供了非常丰富的查找函数：（注：string::npos）

size\_t find( const basic\_string &str) size\_t

size\_t find( const basic\_string &str, size\_t index ); //返回str在字符串中第一次出现的位置（从index开始查找），如果没找到则返回string::npos

size\_t find( const char \*str, size\_t index ); // 同上

size\_t find( const char \*str, size\_t index, size\_t length ); //返回str在字符串中第一次出现的位置（从index开始查找，长度为length），如果没找到就返回string::npos

size\_t find( char ch, size\_t e index ); // 返回字符ch在字符串中第一次出现的位置（从index开始查找），如果没找到就返回string::npos

注意：查找字符串a是否包含子串b,不是用 strA.find(strB) > 0 而是 strA.find(strB) != string::npos 这是为什么呢？

其他常用函数

string &insert(int p,const string &s); //在p位置插入字符串s

string &replace(int p, int n,const char \*s); //删除从p开始的n个字符，然后在p处插入串s

string &erase(int p, int n); //删除p开始的n个字符，返回修改后的字符串

string substr(int pos = 0,int n = npos) const; //返回pos开始的n个字符组成的字符串

void swap(string &s2); //交换当前字符串与s2的值

string &append(const char \*s); //把字符串s连接到当前字符串结尾

void push\_back(char c) //当前字符串尾部加一个字符c

const char \*data()const; //返回一个非null终止的c字符数组，data():与c\_str()类似，用于string转const char\*其中它返回的数组是不以空字符终止,

const char \*c\_str()const; //返回一个以null终止的c字符串，即c\_str()函数返回一个指向正规C字符串的指针, 内容与本string串相同,用于string转const char\*

---my test

#include<iostream>

#include<string>

#include <string.h>

using namespace std;

bool cao(string str){

cout<<str<<endl;

size\_t found=-1;

while(1){

found = str.find("!1");

if(found==string::npos){

break;

}else{

str=str.replace(str.begin()+found,str.begin()+found+2,"0");

cout<<str<<endl;

}

}

while(1){

found = str.find("!0");

if(found==string::npos){

break;

}else{

str=str.replace(str.begin()+found,str.begin()+found+2,"1");

cout<<str<<endl;

}

}

while(1){

found = str.find("&");

if(found==string::npos){

break;

}else if(str[found-1]=='0'||str[found+1]=='0'){

str=str.replace(str.begin()+found-1,str.begin()+found+2,"0");

cout<<str<<endl;

}else {

str=str.replace(str.begin()+found-1,str.begin()+found+2,"1");

cout<<str<<endl;

}

}

while(1){

found = str.find("|");

if(found==string::npos){

break;

}else if(str[found-1]=='1'||str[found+1]=='1'){

str=str.replace(str.begin()+found-1,str.begin()+found+2,"1");

cout<<str<<endl;

}else {

str=str.replace(str.begin()+found-1,str.begin()+found+2,"0");

cout<<str<<endl;

}

}

cout<<"after delet ! & | str = "<<str<<endl;//after delet &

if(str[0]=='1'){

return 1;

}else{

return 0;

}

}

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

{

string str;

cin>>str;

int len =str.length();

string::iterator it;

it = str.begin();

int l=0;

for(int i=0;i<len-l;i++){

if(str[i]==')'){

cout<<"i "<<i<<endl;

for(int j =i;j>=0;j--){

if(str[j]=='('){

cout<<"j "<<j<<endl;

char new\_str[128] ;

strncpy(new\_str,&str[j+1],i-j-1);

new\_str[i-j-1]='\0';

cout<<"new\_str "<<new\_str<<endl;

if(cao(new\_str)){

str.replace(it+j,it+i+1,"1");

l+=(i-j);

}else{

str.replace(it+j,it+i+1,"0");

l+=(i-j);

}

i=j;

cout<<"after delet () str = "<<str<<endl;//after delet &

break;

}

}

}

}

cout<<"end str = "<<str<<endl;//after delet &

if(cao(str)){

cout<<"1"<<endl;

return 1;

}else {

cout<<"0"<<endl;

return 0;

}

}