

华为2015笔试算法题汇总

1.通过键盘输入一串小写字母(a~z)组成的字符串。请编写一个字符串过滤程序，若字符串中出现多个相同的字符，将非首次出现的字符过滤掉。

比如字符串“abacacde”过滤结果为“abcde”。

要求实现函数：void stringFilter(const char *pInputStr, long lInputLen, char *pOutputStr);

【输入】 pInputStr: 输入字符串

lInputLen: 输入字符串长度

【输出】 pOutputStr: 输出字符串，空间已经开辟好，与输入字符串等长；

【注意】只需要完成该函数功能算法，中间不需要有任何IO的输入输出

示例

输入：“deefd” 输出：“def”

输入：“afafafaf” 输出：“af”

输入：“pppppppp” 输出：“p”

main函数已经隐藏，这里保留给用户的测试入口，在这里测试你的实现函数，可以调用printf打印输出

当前你可以使用其他方法测试，只要保证最终程序能正确执行即可，该函数实现可以任意修改，但是不要改变函数原型。

一定要保证编译运行不受影响

```
[cpp] view plaincopy
```

```
////////////////////////////////////
```

```
#include <iostream>
```

```
#include <cassert>
```

```
using namespace std;

bool g_flag[26];

void stringFilter(const char *pInputStr, long lInputLen, char *pOutputStr)
{
    assert(pInputStr != NULL);

    int i = 0;

    if (pInputStr == NULL || lInputLen <= 1)
    {
        return;
    }

    const char *p = pInputStr;
    while(*p != '\0')
    {
        if (g_flag[*p - 'a'])
        {
            p++;
        }else{
            pOutputStr[i++] = *p;
            g_flag[*p - 'a'] = 1;
            p++;
        }
    }
}
```

```
pOutputStr[i] = '\0';  
}  
int main()  
{  
    memset(g_flag,0,sizeof(g_flag));  
    char input[] = "abacacde";  
    char *output = new char[strlen(input) + 1];  
    stringFilter(input,strlen(input),output);  
    cout<<output<<endl;  
    delete output;  
    return 0;  
}
```

2.

通过键盘输入一串小写字母(a~z)组成的字符串。请编写一个字符串压缩程序，将字符串中连续出现的重复字母进行压缩，并输出压缩后的字符串。

压缩规则：

1、仅压缩连续重复出现的字符。比如字符串"abcabc"由于无连续重复字符，压缩后的字符串还是"abcabc"。

2、压缩字段的格式为"字符重复的次数+字符"。例如：字符串"xxxxyyyyyyz"压缩后就成为"3x6yz"。

要求实现函数：

```
void stringZip(const char *pInputStr, long lInputLen, char *pOutputStr);
```

【输入】 pInputStr: 输入字符串

lInputLen: 输入字符串长度

【输出】 pOutputStr: 输出字符串, 空间已经开辟好, 与输入字符串等长;

【注意】只需要完成该函数功能算法, 中间不需要有任何IO的输入输出

示例

输入: "cccddecc" 输出: "3c2de2c"

输入: "edef" 输出: "edef"

输入: "pppppppp" 输出: "8p"

[cpp] view plaincopy

////////////////////////////////////

```
#include <iostream>
```

```
#include <cassert>
```

```
using namespace std;
```

```
void stringZip(const char *pInputStr, long lInputLen, char *pOutputStr)
```

```
{  
    const char *p = pInputStr;  
    int num = 1;  
    int i = 0;  
    p++;  
    while(*p != NULL)  
    {  
        while(*p == *(p-1)&& *p != NULL)  
        {
```

```
num++;  
p++;  
}  
if (num > 1)  
{  
    int size = 0;  
    int temp = num;  
    while(num)        //计算位数  
    {  
        size++;  
        num /= 10;  
    }  
    num = 1;  
  
    for (int j = size; j > 0; j--)  
    {  
        pOutputStr[i+j-1] = '0'+ temp%10;  
        temp /= 10;  
    }  
    i +=size;  
    pOutputStr[i++] = *(p-1);  
    p++;  
}else{  
    pOutputStr[i++] = *(p-1);
```

```
        p++;
    }
}

pOutputStr[i] = '\0';
}

int main()
{
    char input[] = "cccddecc";
    char *output = new char[strlen(input) + 1];
    stringZip(input,strlen(input),output);
    cout<<output<<endl;
    return 0;
}
```

3.通过键盘输入100以内正整数的加、减运算式，请编写一个程序输出运算结果字符串。

输入字符串的格式为：“操作数1 运算符 操作数2”，“操作数”与“运算符”之间以一个空格隔开。

补充说明：

- 1、操作数为正整数，不需要考虑计算结果溢出的情况。
- 2、若输入算式格式错误，输出结果为“0”。

要求实现函数：

```
void arithmetic(const char *pInputStr, long lInputLen, char *pOutputStr);
```

【输入】 pInputStr: 输入字符串

lInputLen: 输入字符串长度

【输出】 pOutputStr: 输出字符串, 空间已经开辟好, 与输入字符串等长;

【注意】 只需要完成该函数功能算法, 中间不需要有任何IO的输入输出

示例

输入: "4 + 7" 输出: "11"

输入: "4 - 7" 输出: "-3"

输入: "9 ++ 7" 输出: "0" 注: 格式错误

[cpp] view plaincopy

////////////////////////////////////

```
#include <iostream>
```

```
using namespace std;
```

```
void arithmetic(const char *pInputStr, long lInputLen, char *pOutputStr)
```

```
{
```

```
    const char *input = pInputStr;
```

```
        char *output = pOutputStr;
```

```
    int sum = 0;
```

```
    int operator1 = 0;
```

```
    int operator2 = 0;
```

```
    char *temp = new char[5];
```

```
    char *ope = temp;
```

```
while(*input != ' ') //获得操作数1
{
    sum = sum*10 + (*input++ - '0');
}
input++;
operator1 = sum;
sum = 0;
```

```
while(*input != ' ')
{
    *temp++ = *input++;
}
```

```
input++;
*temp = '\0';
```

```
if (strlen(ope) > 1 )
{
    *output++ = '0';
    *output = '\0';
    return;
}
```

```
while(*input != '\0') //获得操作数2
```

```
{
    sum = sum*10 + (*input++ - '0');
}

operator2 = sum;

sum = 0;

switch (*ope)
{
case '+':itoa(operator1+operator2,pOutputStr,10);
    break;
case '-':itoa(operator1-operator2,pOutputStr,10);
    break;
default:
    *output++ = '0';
    *output = '\0';
    return;
}
}

int main()
{
    char input[] = "4 - 7";
    char output[] = " ";
    arithmetic(input,strlen(input),output);
```

更多企业校园招聘笔试面试题合集下载: <http://bimian.xuanjianghui.com.cn/>

```
    cout<<output<<endl;
    return 0;
}
```

4.输入1--50个数字，求出最小数和最大数的和

[cpp] view plaincopy

//华为2014年机试题1：输入1--50个数字，求出最小数和最大数的和

//输入以逗号隔开

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

```
#define N 50
```

```
void sort(int a[],int n);
```

```
int main(void)
```

```
{
```

```
    char str[100];
```

```
    int a[N]={0};
```

gets(str); //要点1：动态的输入1--50个整数，不能确定个数，只能用字符串
 输入，然后分离出来

```
    int i=0;
```

```
    int j=0;
```

```
    int sign=1;
```

```
    while(str[i]!='\0')
```

```
{
    if(str[i]!='.') //输入时要在半角输入
    {

        if(str[i]=='-') //要点:2: 有负整数的输入
        {
            // i++; //易错点1

            sign=-1;
        }

        else if(str[i]!='\0') //不用else的话, 负号也会减去 '0'
        {
            a[j]=a[j]*10 + str[i]-'0'; //要点3: 输入的可以是多位数

        }
    }
}
i++;

if(str[i]==',' || str[i]=='\0') //这个判断是在i自加以后
{
    a[j]=a[j]*sign; //易错点2

    sign=1; ////易错点3

    j++; //j就是a数组的个数 范围0到j-1
}
```

```
    }  
  
    sort(a,j);  
    printf("Max number + Min number = %d",a[0]+a[j-1]);  
  
    return 0;  
}  
  
void sort(int a[],int n) //选择排序  
{  
    int i,j;  
    int k;  
    int temp;  
    for(i=0;i<n-1;i++)  
    {  
        k=i;  
        for(j=i+1;j<n;j++)  
        {  
            if(a[k]>a[j])  
                k=j;  
        }  
        if(i!=k)  
        {  
            temp = a[k];
```

更多企业校园招聘笔试面试题合集下载: <http://bimian.xuanjianghui.com.cn/>

```
        a[k] = a[i];  
        a[i] = temp;  
    }  
}  
for(i=0;i<n;i++)  
    printf("%-5d",a[i]);  
puts("");  
}
```