C++编程

目录

字符串

```
Hello C++!
--------------------------------

Process exited after 0.1336 seconds w
ith return value 0
请按任意键继续. . .
```

- 何为字符串？即一串连续的字符
- 比如：

```cpp
#include<bits/stdc++.h>
using namespace std;
int main()
{
    string s = "Hello C++!";
    cout << s;
    return 0;
}
```

- 那这不就是字符数组嘛！

字符串

- 字符串的用途的确和字符数组几乎没区别
- 实际上，字符串也是用字符数组来实现的，不过封装好了类似于函数可以直接调用

- 但在用法上还是有区别的

字符串

- 字符串是一种数据类型，而不是数组，所以它没有下标
- 那能支持多少个字符？弹性长度

特别注意：

1. 字符串不需要定义长度
2. 字符串只能用cin/cout，不支持scanf/printf

字符串的输入输出

```
3
abcdefg
abcdefg
Hello
Hello
NBA
NBA

--------------------------------
```

```cpp
#include<bits/stdc++.h>
using namespace std;
int main()
{
    int k; scanf("%d", &k);
    string s;
    while (k --)
    {
        cin >> s;
        cout << s << endl;
    }
    return 0;
}
```

字符串的长度

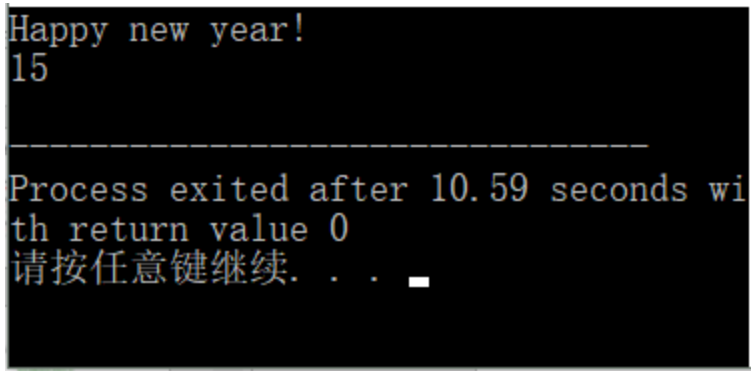
```
15

--------------------------------
Process exited after 0.2822 seconds w
ith return value 0
请按任意键继续. . .
```

- 字符数组获取长度是 strlen(x)
- 字符串获取长度则是 **x.size( )** 或者 x.length( )

```
#include<bits/stdc++.h>
using namespace std;
int main()
{
    string s = "Happy new year!";
    cout << s.size() << endl;
    return 0;
}
```

getline 函数



- cin 读入字符串也会遇到空格停下
- 字符数组解决这个问题是用gets(x)函数读入整行
- 字符串则是用**getline(cin，x)**函数读入整行

```
#include<bits/stdc++.h>
using namespace std;
int main()
{
    string s;
    getline(cin, s);
    cout << s.size() << endl;
    return 0;
}
```

标题统计

- 凯凯刚写了一篇美妙的作文，请问这篇作文的标题中有多少个字符？ 注意：标题中可能包含大、小写英文字母、数字字符、空格和换行符。统计标题字符数时，空格和换行符不计算在内
- 输入文件只有一行

Sample input	Sample output
Ca 45	4

秒掉

- 题目都说了输入只有一行，所以只有空格不计数

```cpp
#include<bits/stdc++.h>
using namespace std;
int ans;
string s;
int main()
{
    getline(cin, s);
    for (int i = 0; i < s.size(); i ++)
        if (s[i] != ' ') ans ++;
    cout << ans;
    return 0;
}
```

字符串的用法

```
NOIP2019
N
O
I
P
2
0
1
9
```

- 字符串虽然是一个数据类型，但它也是用字符数组实现的
- 所以也可以对字符串中的单个字符操作

```
#include<bits/stdc++.h>
using namespace std;
int main()
{
    string s;
    getline(cin, s);
    for (int i = 0; i < s.size(); i ++)
        cout << s[i] << endl;
    return 0;
}
```

- 从这点上看，字符数组能做到的，字符串基本也能做到

字符串的用法

```
2018
ha pp y new y ear !
^Z
2018happynewyear!
--------------------------------

Process exited after 13.71 seconds wi
th return value 0
请按任意键继续. . .
```

- 字符串联结运算“+”

```
#include<bits/stdc++.h>
using namespace std;
int main()
{
    string s, s2;
    cin >> s;
    while (cin >> s2)
        s += s2;  //字符串联结运算
    cout << s;
    return 0;
}
```

消除多余空格

- 某英文文档（由大小写英文及标点符号构成，文档长度未知）因为排版疏忽，导致部分单词间出现了多余的空格，你的程序要能消除这些多余的空格

Sample input	Sample output
This is a great program	This is a great program

参考代码

- 我们利用字符串联结运算来消除多余的空格

```
#include<bits/stdc++.h>
using namespace std;
int main()
{
    string s, s2;
    cin >> s;
    while (cin >> s2)
        s += ' ' + s2;
    cout << s;
    return 0;
}
```

更多黑科技：reverse 函数

```
CCF
FCC
--------------------------------

Process exited after 6.26 seconds wit
h return value 0
请按任意键继续. . .
```

- reverse函数用于字符串翻转

```cpp
#include<bits/stdc++.h>
using namespace std;
int main()
{
    string s;
    cin >> s;
    reverse(s.begin(), s.end());
    cout << s;
    return 0;
}
```

更多黑科技：reverse 函数

```
CCF
CFC
--------------------------------

Process exited after 2.61 seconds wit
h return value 0
请按任意键继续. . .
```

- 还可以只翻转字符串的一部分
- 表示从1号位开始翻转

```cpp
#include<bits/stdc++.h>
using namespace std;
int main()
{
    string s;
    cin >> s;
    reverse(s.begin() + 1, s.end());
    cout << s;
    return 0;
}
```

判断回文串

- 给定 k 条字符串，分别判断其是否为回文串

Sample input	Sample output
3 //k bbc noipion Aa	no yes no

秒了它

```
3
bbc
no
noipion
yes
Aa
no

--------------------------------
```

```cpp
#include<bits/stdc++.h>
using namespace std;
int main()
{
    int k; scanf("%d", &k);
    string s, s2;
    while (k --)
    {
        cin >> s;
        s2 = s;
        reverse(s.begin(), s.end());
        if (s == s2) printf("yes\n");
            else printf("no\n");
    }
    return 0;
}
```

字符串的用法

- 从这个例子看出：

1. 字符串可以直接赋值
2. 字符串可以直接判相等

- 体现出比字符数组更好的整体性

```
#include<bits/stdc++.h>
using namespace std;
int main()
{
    int k; scanf("%d", &k);
    string s, s2;
    while (k --)
    {
        cin >> s;
        s2 = s;
        reverse(s.begin(), s.end());
        if (s == s2) printf("yes\n");
            else printf("no\n");
    }
    return 0;
}
```

更多黑科技：erase 函数

```
2019CCF
2019
--------------------------------

Process exited after 3.454 seconds wi
th return value 0
请按任意键继续. . .
```

- erase 函数用于删除字符串指定部分
- **x.erase(n, m)**从字符串x的n号位置开始，连续删除m个字符

```
#include<bits/stdc++.h>
using namespace std;
int main()
{
    string s;
    cin >> s;
    s.erase(4, 3);
    cout << s;
    return 0;
}
```

清理高位零

- 用字符串读入某给定k个大整数n（$n \le 10^{10000}$）,但是发现该数前面有多余的高位零，要求清除这些高位零

Sample input	Sample output
3 //k 00002019 10086 04399	2019 10086 4399

秒了它

```
3
00002019
2019
10086
10086
04399
4399

--------------------------------
```

```cpp
#include<bits/stdc++.h>
using namespace std;
int main()
{
    int k; scanf("%d", &k);
    string s;
    while (k --)
    {
        cin >> s;
        while (s[0] == '0')
            s.erase(0, 1);
        cout << s << endl;
    }
    return 0;
}
```

延伸一下

- 如果这些数不是这么大，而是在int或者long long范围内呢？

- 那我们需要（从字符数组读入后转为）开int数组存入每个数的各位数字，然后对每个数组：


```
int i = 0;
while (a[i] == 0) i ++;  // i即是第一个非零数字的位置
for ( ; i <= 数的位数; i ++)
    printf("%d", a[i]);
```

正确做法

```
3
00002019
2019
10086
10086
03499
3499

--------------------------------
```

- int 型读入本来就会自动过滤高位零的

```cpp
#include<bits/stdc++.h>
using namespace std;
int main()
{
    int k, n; scanf("%d", &k);
    while (k --)
    {
        scanf("%d", &n);
        printf("%d\n", n);
    }
    return 0;
}
```

不忘初心，方得始终

类似的初心

- 给定一个只包含“+”、“-”，和数字的算术表达式，求其值

Sample input	Sample output
137+502+51-24	666

不忘初心，方得始终

类似的初心

- 如果我们忘了初心，意思就是：学了后面忘前面

- 我们会要用字符数组接收这条读入的算术表达式，然后在字符数组中扫一遍，根据读到的“+”、“-”运算符分别处理

不忘初心，方得始终

类似的初心

```
137+502+51-24
^Z
666
--------------------------------

Process exited after 13.91 seconds wi
th return value 0
请按任意键继续. . .
```

- 那么初心是什么呢？我们试试cin/scanf的功能有多强大

```
#include<bits/stdc++.h>
using namespace std;
int ans, a;
int main()
{
    cin >> ans;  //第一个数单独处理
    while (cin >> a) ans += a;
    cout << ans;
    return 0;
 }
```

不忘初心，方得始终

更多黑科技：find 函数

- find 函数用于字符串内部定位
- **x.find(y)**：在字符串x中查找字符串y第一次出现的位置
- 如果没找到，返回-1

```
NOIP
OI
1
--------------------------------
Process exited after 13.39 seconds wi
th return value 0
请按任意键继续. . .
```

```cpp
#include<bits/stdc++.h>
using namespace std;
int main()
{
    string s, s2;
    cin >> s >> s2;
    cout << s.find(s2);
    return 0;
}
```

更多黑科技：find 函数

- find 函数用于字符串内部定位
- **x.find(y)**：在字符串x中查找字符串y第一次出现的位置
- 如果没找到，返回-1
- 空格也计算位置

```
Happy new year!
ye
10
--------------------------------
Process exited after 24.6 seconds wit
h return value 0
请按任意键继续. . .
```

```cpp
#include<bits/stdc++.h>
using namespace std;
int main()
{
    string s, s2;
    getline(cin, s);
    getline(cin, s2);
    cout << s.find(s2);
    return 0;
}
```

更多黑科技：find 函数

```
Happy new year!
new
-1
--------------------------------
Process exited after 6.363 seconds wi
th return value 0
请按任意键继续. . .
```

- find 函数用于字符串内部定位
- **x.find(y)**：在字符串x中查找字符串y第一次出现的位置
- 如果没找到，返回-1
- 空格也计算位置
- 还可以从x的指定位置开始查找
- **x.find(y, m)**：从字符串x的m号位置开始查找字符串y第一次出现的位置

```
#include<bits/stdc++.h>
using namespace std;
int main()
{
    string s, s2;
    getline(cin, s);
    getline(cin, s2);
    cout << (int)s.find(s2, 10);
    return 0;
}
```

单词统计

- 一般的文本编辑器都有查找单词的功能，该功能可以快速定位特定单词在文章中的位置，有的还能统计出特定单词在文章中出现的次数
- 现在，请你编程实现这一功能，具体要求是：给定一个单词，请你输出它在给定的文章中出现的次数和第一次出现的位置，没有出现则输出-1。注意：匹配单词时，不区分大小写，但要求完全匹配

Sample input	Sample output
to To be or not to be is a question	2 0

分析

- 需要解决几个问题：

1. 大小写不一致

这个可以直接利用tolower函数全转换为小写

2. to 和 toto 是不匹配的，但find函数会认为它们匹配

空格	to	空格
空格	toto	空格

```
word = ' ' + word + ' ';
text = ' ' + text + ' ';
```

分析

- 需要解决几个问题：

3. find函数很容易找到第一个位置

```
first = text.find(word);
```

4. 而在这之后我们需要继续往后找

To be or not to be is a question

分析

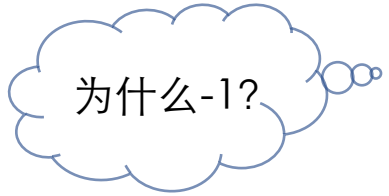
- 需要解决几个问题：

4. 为此我们记录当前找的位置pos，并作为下次查找的起点

```
pos = text.find(word, pos)
```

5. 在往后跳之前，需要把当前找到的word给跳过，不然会重复

```
pos += word.size() - 1;
```



To be or not to be is a question

分析

- 需要解决几个问题：

6. 最后一个问题，这种查找什么时候结束？

因为当find函数没有找到的时候，会返回-1，我们就利用这点

```
while ((pos = text.find(word, pos)) >= 0)
{


}
```

参考代码

```cpp
#include<bits/stdc++.h>
using namespace std;
string word, text;
int pos, first, ans;
void convert()  //转换为全小写并加空格
{
    for (int i = 0; i < word.size(); i ++)
        word[i] = tolower(word[i]);
    for (int i = 0; i < text.size(); i ++)
        text[i] = tolower(text[i]);
    word = ' ' + word + ' ';
    text = ' ' + text + ' ';
}
```

```cpp
int main()
{
    getline(cin, word);
    getline(cin, text);
    convert();
    first = text.find(word);
    while ((pos = text.find(word, pos)) >= 0)
    {
        pos += word.size() - 1;
        ans ++;
    }
    cout << ans << " " << first;
    return 0;
}
```

课外加练

- luogu 5015　　标题统计
- luogu 1308　　统计单词数