此文章参考其他人的两篇文章，所以标为转载，点击这里查看参考文档查看

String类——按照类的方式进行动态管理字符串

调用说明

1、引入头文件

string要使用先引入头文件

```
#include <string>
```

注意这里的是`string`，学了C的同学请注意，不是string.h

底层：是一种顺序表的结构，元素是char类型的字符

2、命名空间

string是模板类，位于命名空间std中，通常为了使用方便还要加上这样一句

```
using namespace std;
```

String简介

C++在C的基础上增加了**类**，和**模板**（STL），string就是C++的一个类模板。但是C没有string，它有字符数组。

注意：C的字符串是指针（数组的本质是指针），而C++的字符串是类模板

string可以存储一串字符，所以叫做字符串，一个string变量C++好像并没有规定最大限度，反正可以存储非常大的量，我们可以用`s.max_size()`的方法查看最大限度，就是可以存储多少个字符，因为一个字符是一个字节。（max_size()的数值是4611686018427387897）

string存储参考：https://bbs.csdn.net/topics/30485933、https://zhidao.baidu.com/question/438024764000528404.html

string也可以变成一个数组，这样可以存储多行的数据

String语法基础

定义

定义一个String变量很简单

```
string s;
```

即可

输入

也很简单，直接用cin就足以了

```
string s;
cin >> s;
```

即可

整行输入

用getline输入法

```
string s;
getline(cin, s);
```

即可输入带有空格的字符串

输出

```
cout << s << endl;
```

就足够，可以用printf和scanf代替

测试代码1

```
#include <iostream>
#include <string>
using namespace std;
int main ( )
{
    string str;  //定义了一个空字符串str
    str = "Hello world";   // 给str赋值为"Hello world"
    char cstr[] = "abcde";  //定义了一个C字符串
    string s1(str);       //调用复制构造函数生成s1，s1为str的复制品
    cout<<s1<<endl;
    string s2(str,6);     //将str内，开始于位置6的部分当作s2的初值
    cout<<s2<<endl;
    string s3(str,6,3);  //将str内，开始于6且长度顶多为3的部分作为s3的初值
        cout<<s3<<endl;
    string s4(cstr);   //将C字符串作为s4的初值
    cout<<s4<<endl;
    string s5(cstr,3);  //将C字符串前3个字符作为字符串s5的初值。
```

```
    cout<<s5<<endl;
    string s6(5,'A');  //生成一个字符串，包含5个'A'字符
    cout<<s6<<endl;
    string s7(str.begin(),str.begin()+5); //区间str.begin()和str.begin()+5内的字符作为初值
    cout<<s7<<endl;
    return 0;
}
```

程序的运行结果如下：

```
Hello world
world
wor
abcde
abc
AAAAA
Hello
```

string的比较操作

你可以用 ==、>、<、>=、<=、和!=比较字符串，可以用+或者+=操作符连接两个字符串，并且可以用[]获取特定的字符。

```
#include <iostream>
#include <string>
using namespace std;
int main()
{
    string str;
    cout << "Please input your name:"<<endl;
    cin >> str;
    if( str == "Li" )   // 字符串相等比较
        cout << "you are Li!"<<endl;
    else if( str != "Wang" )  // 字符串不等比较
        cout << "you are not Wang!"<<endl;
    else if( str < "Li")     // 字符串小于比较，>、>=、<=类似
        cout << "your name should be ahead of Li"<<endl;
    else
        cout << "your name should be after of Li"<<endl;
    str += ", Welcome!";  // 字符串+=
    cout << str<<endl;
    for(int i = 0 ; i < str.size(); i ++)
        cout<<str[i];  // 类似数组，通过[]获取特定的字符
    return 0;
}
```

结果：

```
Please input your name:
Zhang↙
you are not Wang!
Zhang, Welcome!
Zhang, Welcome!
```

上例中，`cout<< str[i];`可换为：`cout<< str.at(i);`

string类的常用构造函数：

- string str——构造空的string类对象，即空字符串
- string str(str1)——str1 和 str 一样
- string str("ABC")——等价于 str="ABC"
- string str("ABC",strlen)——等价于 "ABC" 存入 str 中，最多存储 strlen 个字节
- string str("ABC",stridx,strlen)——等价于 "ABC" 的stridx 位置，作为字符串开头，存到str中，最多存储strlen 个字节
- string str(srelen,'A')——存储 strlen 个 'A' 到 str 中

---

string类的常用函数

下面的string变量统一叫做`str`

assign函数

1. str.assign(“ABC”) ——把str清空，重新赋值”ABC”
2. str.assign(“ABC”, 2) ——把str清空，重新赋值”ABC”，然后保留2位，str=“AB”

---

1. str.length() ——返回字符串长度、
2. str.size() ——和`length`一样
3. str.**reasize**(10) ——设置当前 str 的大小为10，若大小大与当前串的长度，\0 来填充
4. str.**reasize**(10,char c) ——设置当前 str 的大小为10，若大小大与当前串的长度，字0符c 来填充
5. str.reserve(10) ——设置str的容量 10，不会填充数据
6. str.swap(str1) ——交换 str1 和 str 的字符串
7. str.push_back('A') ——在str末尾添加一个字符 'A' ，参数必须是字符形式
8. str.append("ABC") ——在str末尾添加一个字符串 "ABC"，参数必须是字符串形式

---

insert函数方法：

1. str.insert(2,3,'A')——在str下标为2的位置添加 3个 字符'A'
2. str.insert(2,"ABC")——在str下标为2的位置添加 字符串 "ABC"
3. str.insert(2,"ABC",1)——在str下标为2的位置添加 字符串 "ABC" 中 1个 字符
4. str.insert(2,"ABC",1,**1**)——在str下标为2的位置添加 字符串 "ABC" 中从位置 1 开始的 1 个字符

注：上个函数参数中加粗的 **1** ，可以是 **string::npos**，这时候最大值，从 位置1 开始后面的全部字符

---

1. str.erase(2)——删除 下标2 的位置开始，之后的全删除
2. str.erase(2,1)——删除 下标2 的位置开始，之后的 1个 删除
3. str.clear()——删除 str 所有
4. str.replace(2,4,"abcd")——从 下标2 的位置，替换 4个字节 ，为"abcd"
5. str.empty()——判空

反转相关

引入头文件`#include <algorithm>`

```
reverse(str.begin(), str.end());
```

格式如上，可以原封不动的用这个语句，意思是将str的开始到结尾反转，当然begin和end也可以改成具体的值

拷贝相关

1. str1=str.substr(2)——提取子串，提取出 str 的 下标为2 到末尾，给 str1
2. str1=str.substr(2,3)——提取子串，提取出 str 的 下标为2 开始，提取三个字节，给 str1
3. const char* s1=str.data()——将string类转为字符串数组，返回给s1

   char* s=new char[10]
4. str.copy(s,count,pos)——将 str 里的 pos 位置开始，拷贝 count个 字符,存到 s 里

参考：

1、https://blog.csdn.net/qq_42659468/article/details/90381985

2、https://www.cnblogs.com/X-Do-Better/p/8628492.html