string的使用

1. string库：`#include<string.h>`
   如果已经用了通用头文件 `<bits/stdc++.h>` 就不用再引用了

---

2. 截取一个字符串的子串：`str.substr(start,len)`
   例如：

```
    string s = str.substr(2,5);
```

   表示截取 `str` 的子串并赋值给 `s` ，开始位置为2 (从0开始)，长度为5
   - 如果第二个数 `len` 省略,则会从开始位置截取到末尾
   - 原本的字符串并没有改变，要赋值给另一个字符串

---

3. string 的初始化
   有五种方法：
   i. 使用字符串字面量 (即直接将一段字符串赋值给string)

```
string s = "Hello World";
```

   ii. 赋值
   将另一个已经存在的 string 赋值给现在的

```
string s = s0;
```

   iii. 使用部分字符串 (相当于赋值)

```
string s = s0.substr(0,5);
```

   iv. 使用字符串数组 (感觉没什么用，本来string都可以当字符串数组用)

```
const char *s0 = "Hello World";
string s(s0);
```

   v. 重复字符

```
string s(5,'A'); //即 string s = "AAAAA"
```

- 注意： 只能是字符，对字符串不适用

---

4. 将string类型转换成字符串数组类型 `str.c_str()`；

```
    char * s = str.c_str();
```

- 也是要赋值使用，对原 string 没有影响

---

5. 获取字符串长度 `s.length()`

---

6. 拼接字符串 `+` 或者 `s.append()` (用 `+` 就够了我觉得)
   `s = s1 + 'a' + s2;`
   `s = s1.append("a").append(s2);`

- 注意：append 里面要是字符串，不能是单个字符
  不能写成 `s = s1.append('a').append(s2);`

---

7. 查找字符串的位置 `s.find()`
   `int pos = s.find("world");`

- 返回类型其实是 `size_t` (无符号整数类型)
- 返回的是出现的第一个位置的第一个字符的下标 (从 0 开始)

---

8. 字符串替换 `s.replace(start,len,str)`
   例如：

```
string s = "abcdefghijklmnopq";
s.replace(5,5,"replacething");// s = "abcde replacething klmnopq"
```

   第一个参数为替换开始的位置(从0开始)，第二个参数为替换的个数，第三个参数为替换的内容

---

9. string 字符串可以直接进行比较
   `if(s1 > s2)`

- 比较规则按照字典顺序

---

10. string 的遍历

- 直接遍历

```
string s = "dsfasdfsadfsadf";
for(int i = 0;i < s.length();++i)
{
    cout << s[i];
}
```

- auto

```
string s = "dfasdfsadfdsafa";
for(auto i : s)
{
    cout << i;
}
```

---

11. 将数字转换为 string : `string s = to_string(num);`