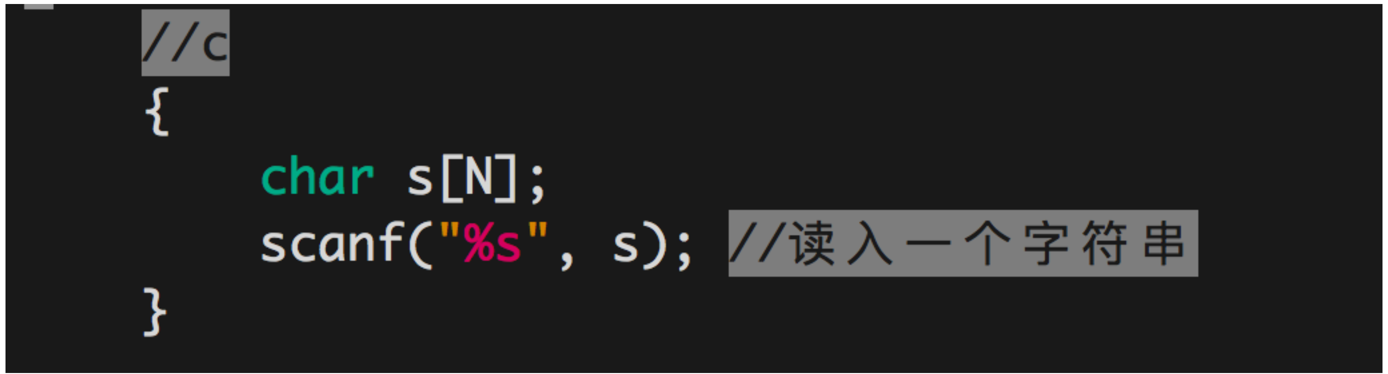
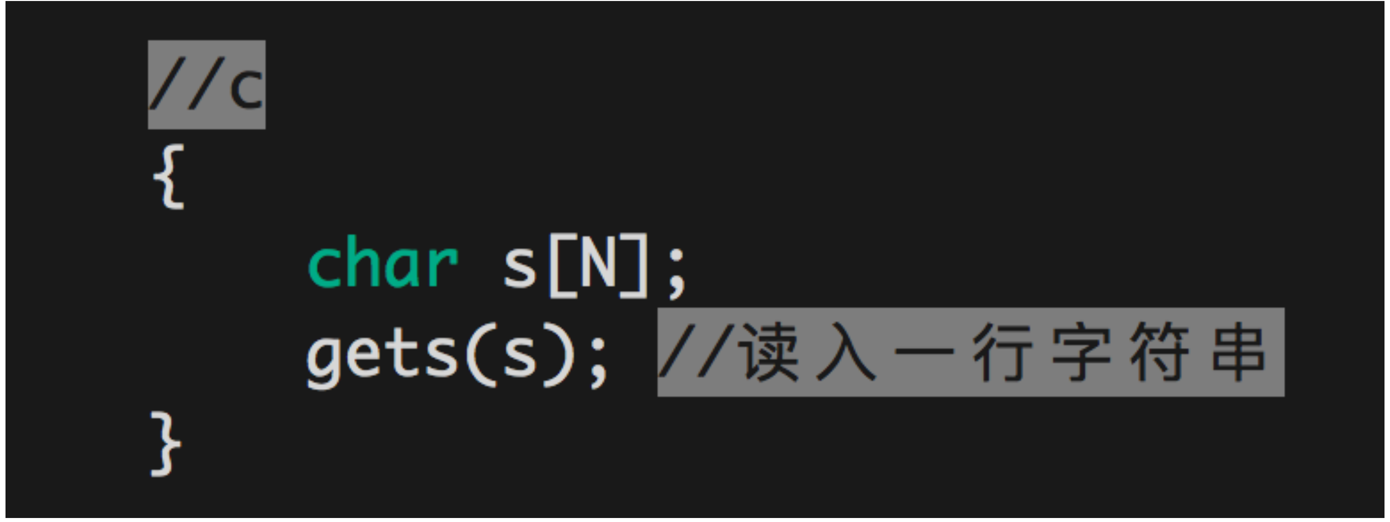
字符串处理技巧

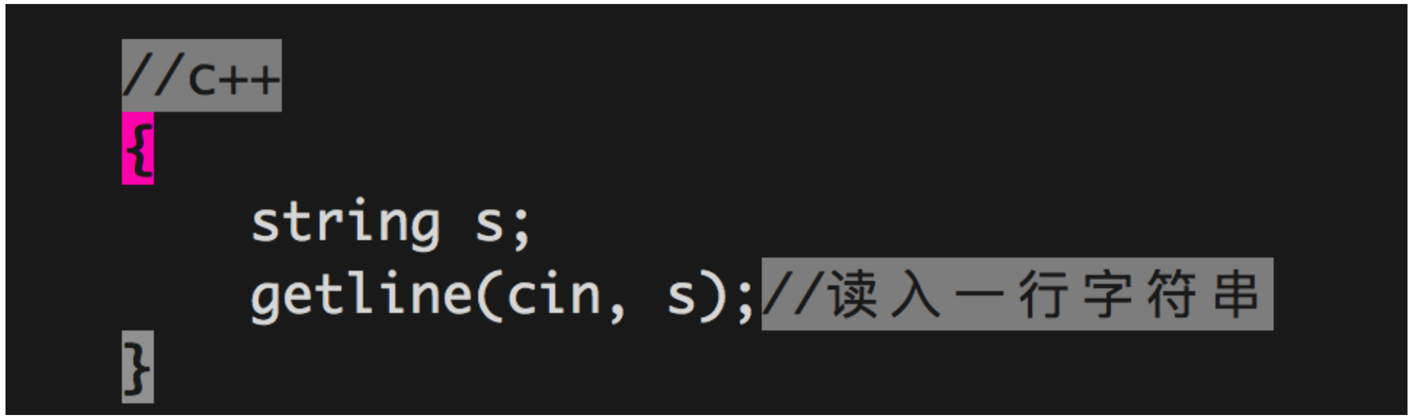
字符串的读入



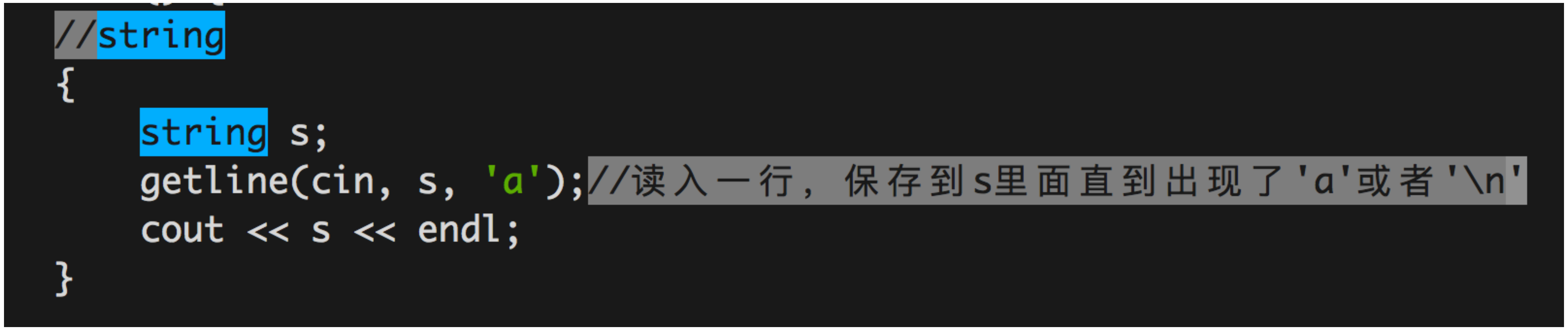
```
//c
{
    char s[N];
    scanf("%s", s); //读入一个字符串
}
```



```
//c
{
    char s[N];
    gets(s); //读入一行字符串
}
```

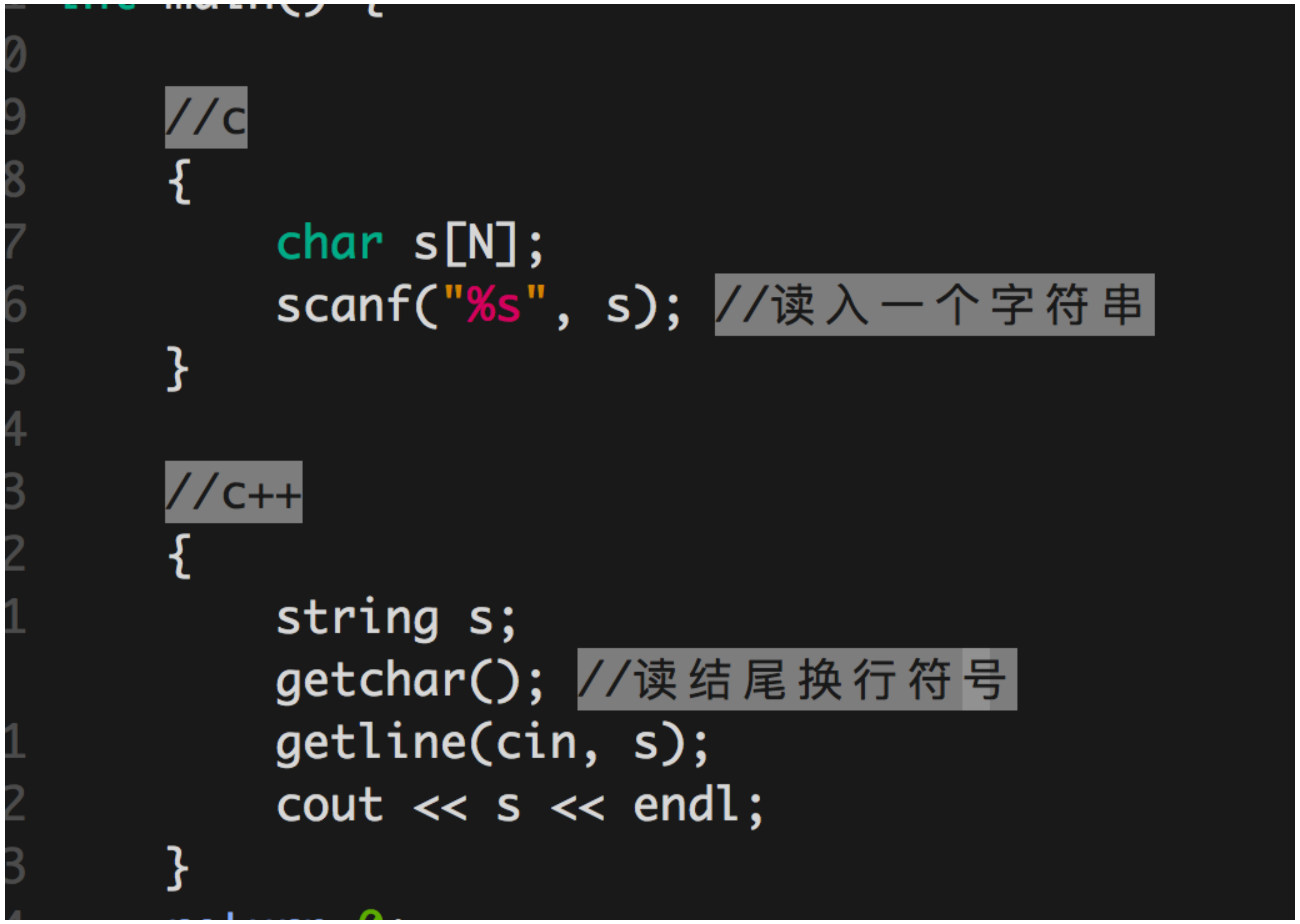


```
//c++
{
    string s;
    getline(cin, s);//读入一行字符串
}
```



```
//string
{
    string s;
    getline(cin, s, 'a');//读入一行，保存到s里面直到出现了'a'或者'\n'
    cout << s << endl;
}
```

混用要注意读掉结尾换行



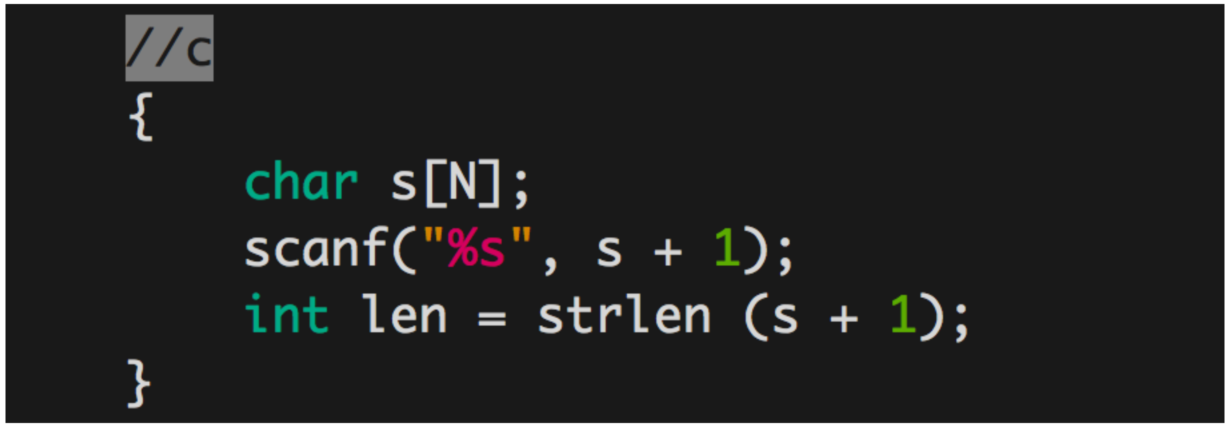
```
int main() {

    //c
    {
        char s[N];
        scanf("%s", s); //读入一个字符串
    }

    //c++
    {
        string s;
        getchar(); //读结尾换行符号
        getline(cin, s);
        cout << s << endl;
    }
    return 0;
```

C语言函数

- strcmp（s1, s2），比较两个字符串的字典序，返回值为正表示s1 > s2,负表示s1 < s2, 0表示s1 = s2

- strlen(s),获取s的长度

- 从s[1]开始读入字符串

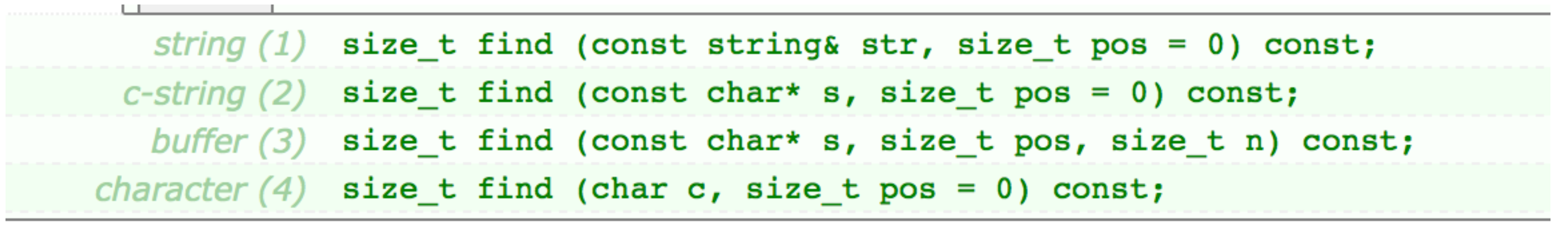
- 

```
//c
{
    char s[N];
    scanf("%s", s + 1);
    int len = strlen (s + 1);
}
```

string

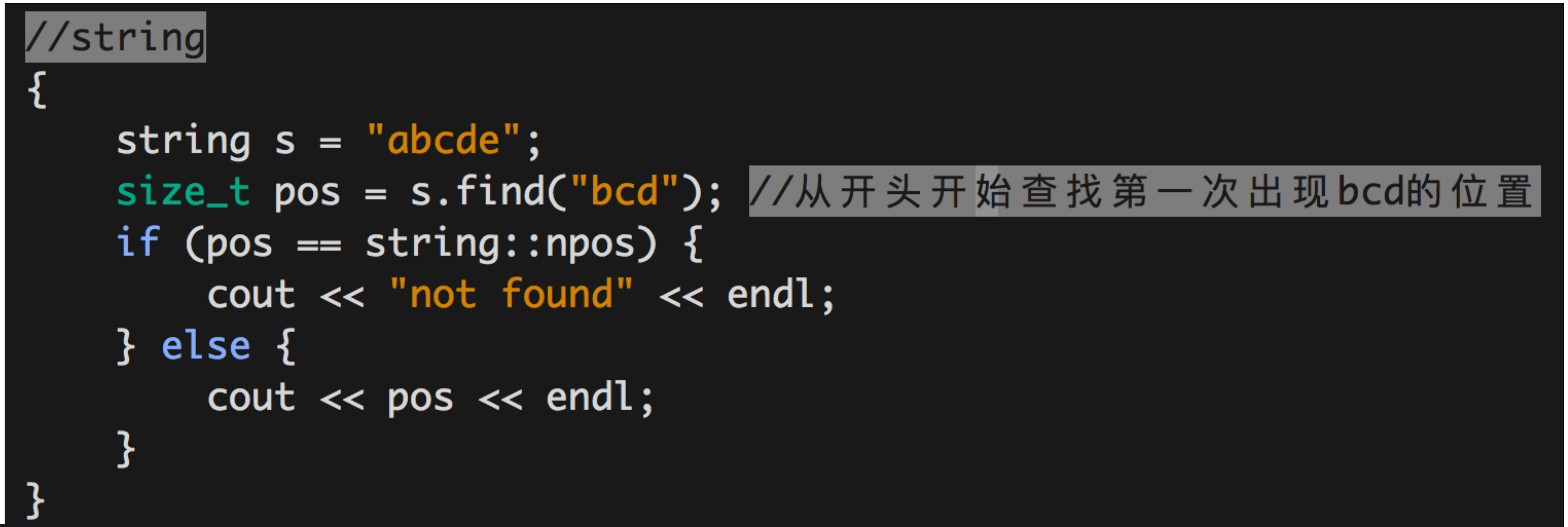
- string s；
- += 追加字符或者字符串
- s[i]用来访问某个位置
- S = “test”;//初始化

string::find

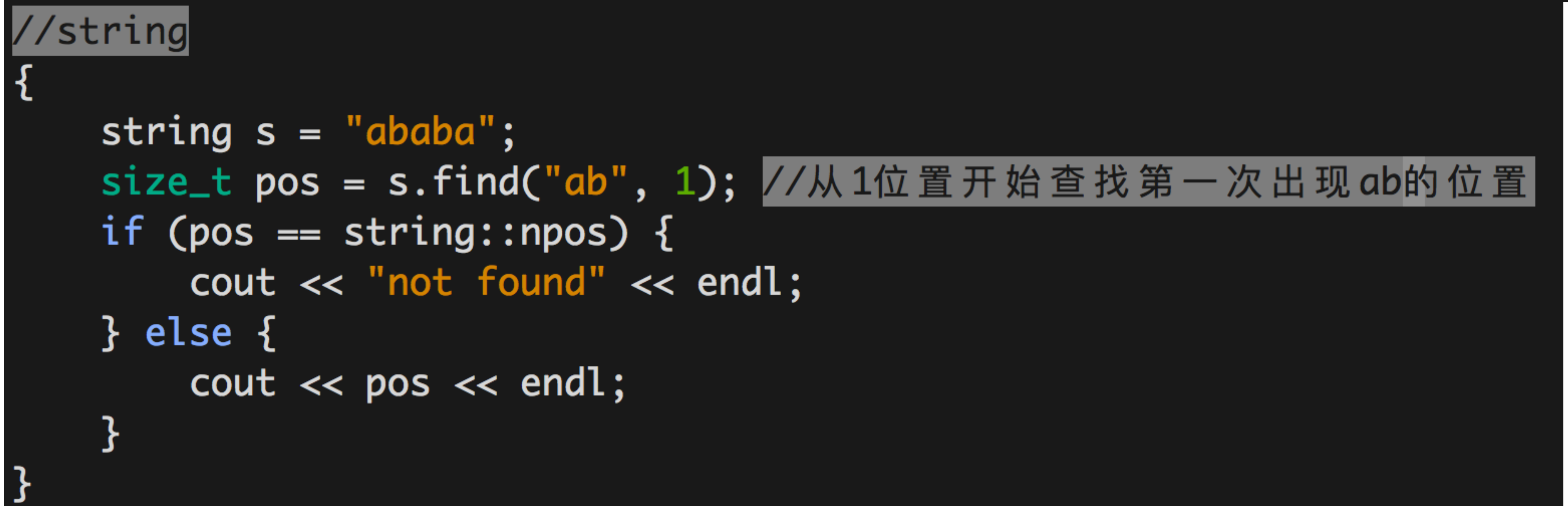


string (1)	`size_t find (const string& str, size_t pos = 0) const;`
c-string (2)	`size_t find (const char* s, size_t pos = 0) const;`
buffer (3)	`size_t find (const char* s, size_t pos, size_t n) const;`
character (4)	`size_t find (char c, size_t pos = 0) const;`

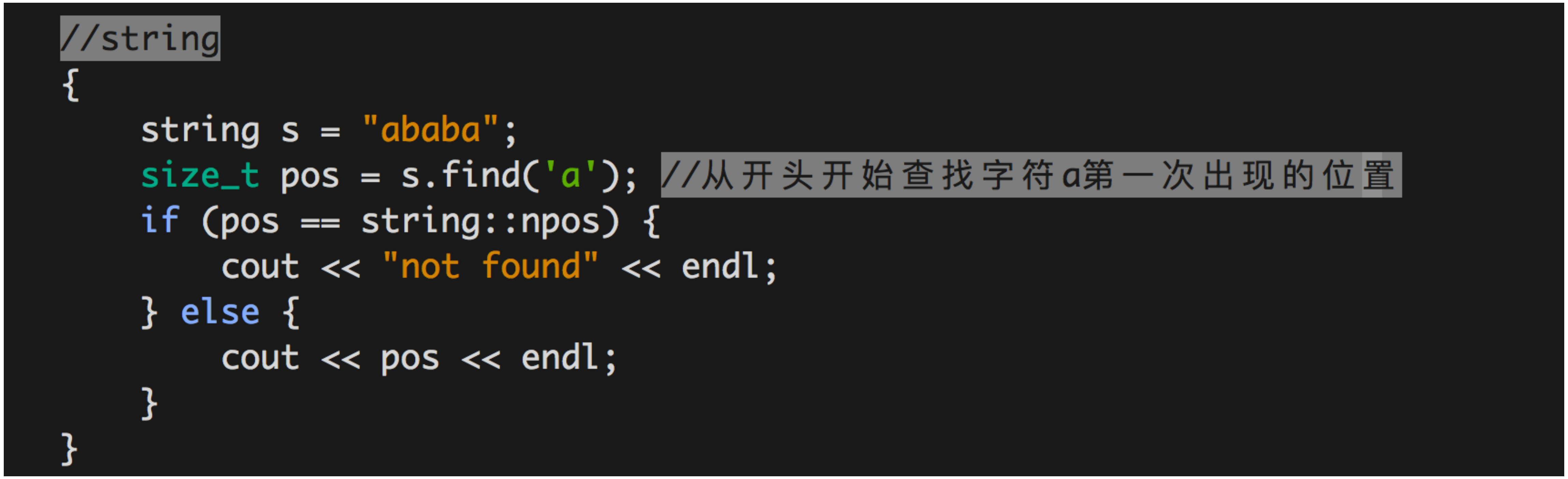
例子



```
//string
{
    string s = "abcde";
    size_t pos = s.find("bcd"); //从开头开始查找第一次出现bcd的位置
    if (pos == string::npos) {
        cout << "not found" << endl;
    } else {
        cout << pos << endl;
    }
}
```



```
//string
{
    string s = "ababa";
    size_t pos = s.find("ab", 1); //从1位置开始查找第一次出现ab的位置
    if (pos == string::npos) {
        cout << "not found" << endl;
    } else {
        cout << pos << endl;
    }
}
```



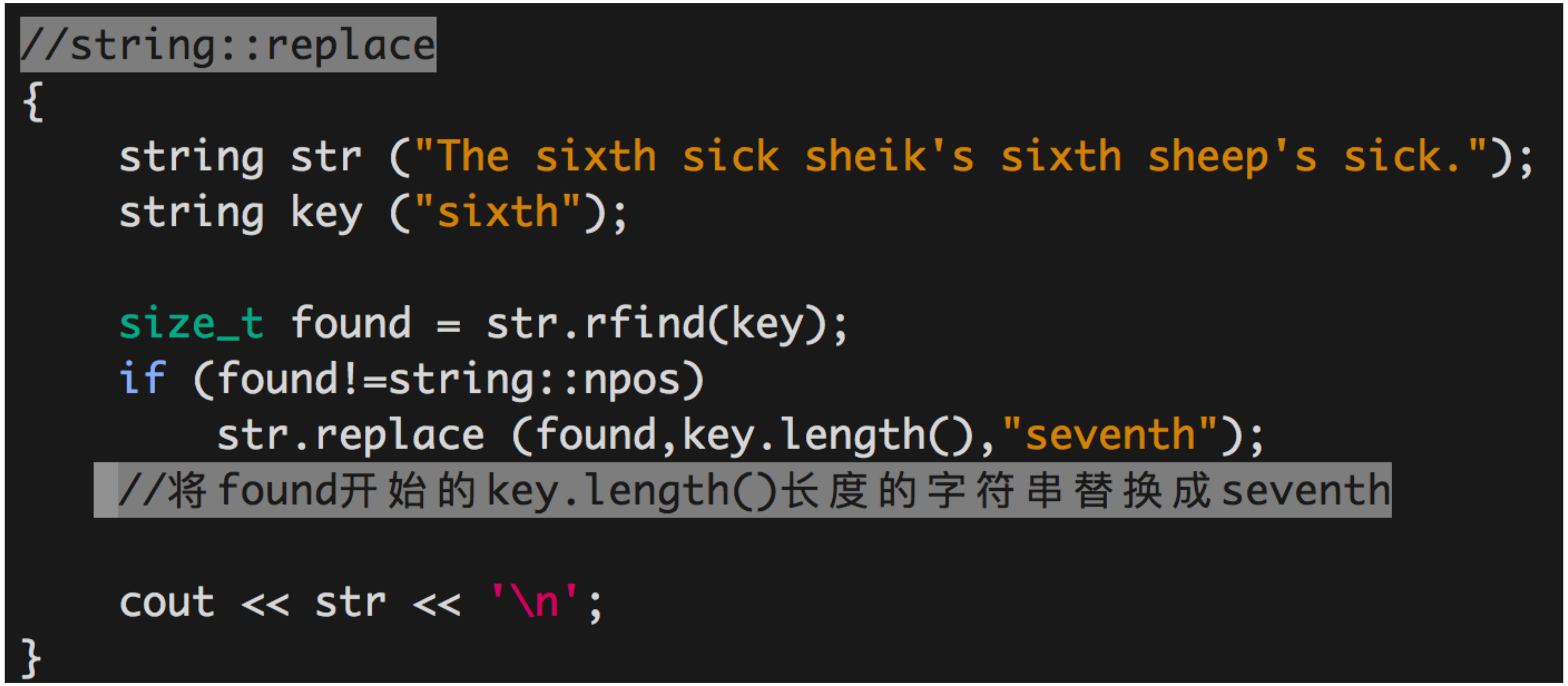
```
//string
{
    string s = "ababa";
    size_t pos = s.find('a'); //从开头开始查找字符a第一次出现的位置
    if (pos == string::npos) {
        cout << "not found" << endl;
    } else {
        cout << pos << endl;
    }
}
```

- 第一个参数可以是一个string类型，也可以是一个char*类型，也可以是一个char

string::rfind

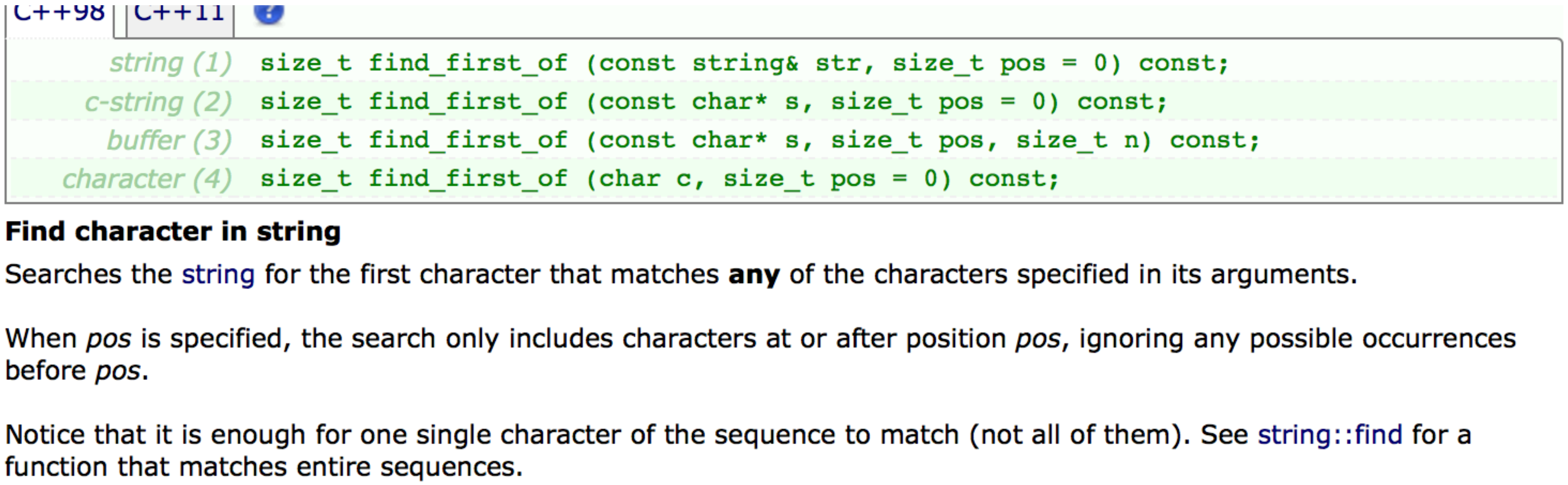
- 同string::find,从后往前查找到第一个匹配的位置

string::replace



```
//string::replace
{
    string str ("The sixth sick sheik's sixth sheep's sick.");
    string key ("sixth");

    size_t found = str.rfind(key);
    if (found!=string::npos)
        str.replace (found,key.length(),"seventh");
    //将found开始的key.length()长度的字符串替换成seventh

    cout << str << '\n';
}
```

string::find_first_of



```
   string (1)  size_t find_first_of (const string& str, size_t pos = 0) const;
 c-string (2)  size_t find_first_of (const char* s, size_t pos = 0) const;
   buffer (3)  size_t find_first_of (const char* s, size_t pos, size_t n) const;
character (4)  size_t find_first_of (char c, size_t pos = 0) const;
```

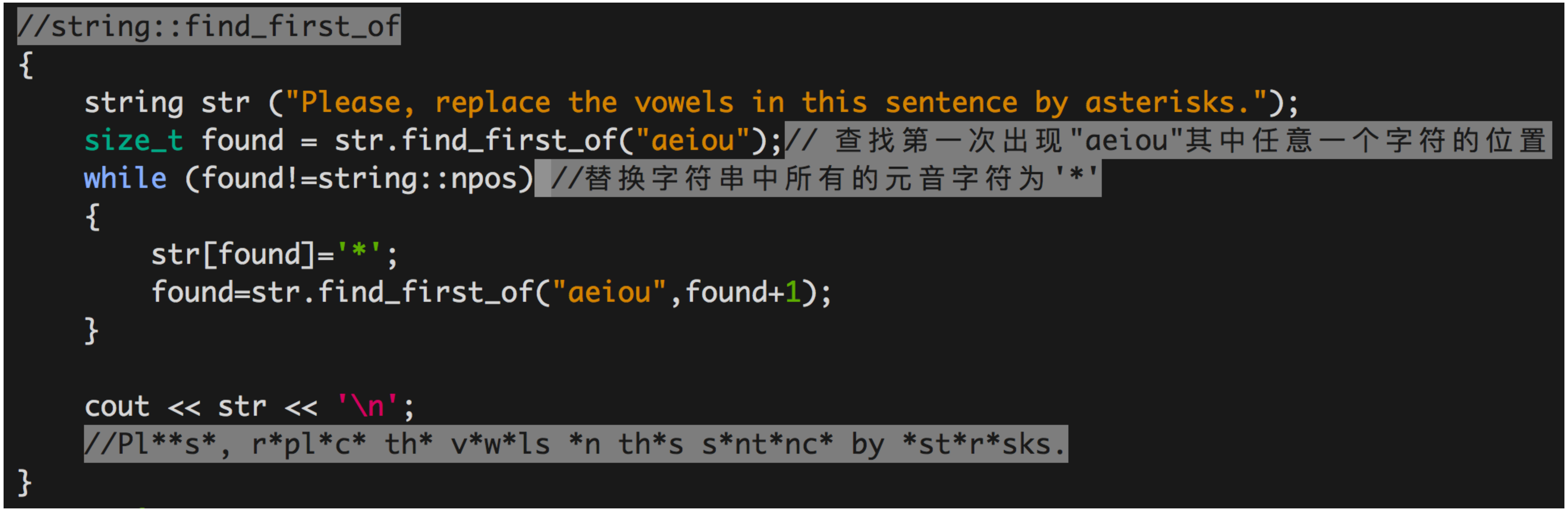
Find character in string

Searches the string for the first character that matches **any** of the characters specified in its arguments.

When *pos* is specified, the search only includes characters at or after position *pos*, ignoring any possible occurrences before *pos*.

Notice that it is enough for one single character of the sequence to match (not all of them). See string::find for a function that matches entire sequences.

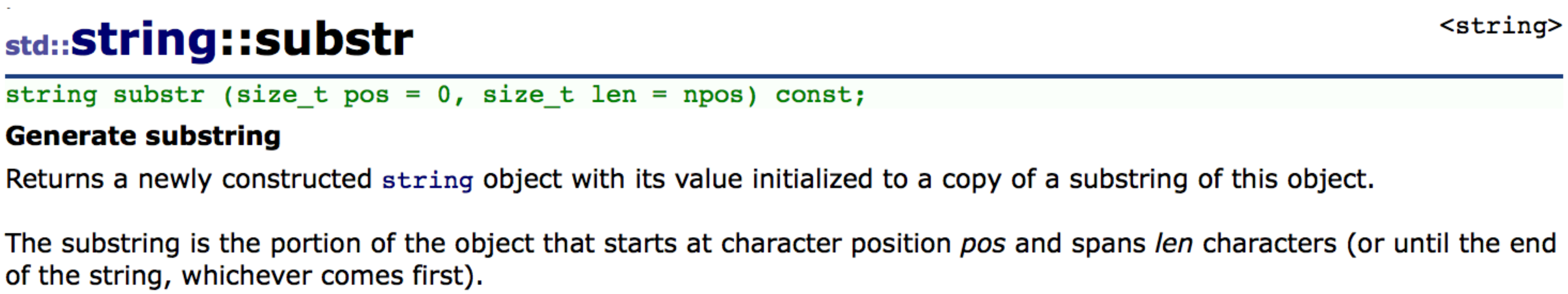
例子



```
//string::find_first_of
{
    string str ("Please, replace the vowels in this sentence by asterisks.");
    size_t found = str.find_first_of("aeiou");// 查找第一次出现"aeiou"其中任意一个字符的位置
    while (found!=string::npos) //替换字符串中所有的元音字符为'*'
    {
        str[found]='*';
        found=str.find_first_of("aeiou",found+1);
    }

    cout << str << '\n';
    //Pl**s*, r*pl*c* th* v*w*ls *n th*s s*nt*nc* by *st*r*sks.
}
```

- 同样的，还有find_last_of查找最后一个出现给定字符集中某个字符的位置，
- find_first_not_of. find_last_not_of等通过字面也能理解，用法都类似

string::substr



std::string::substr

<string>

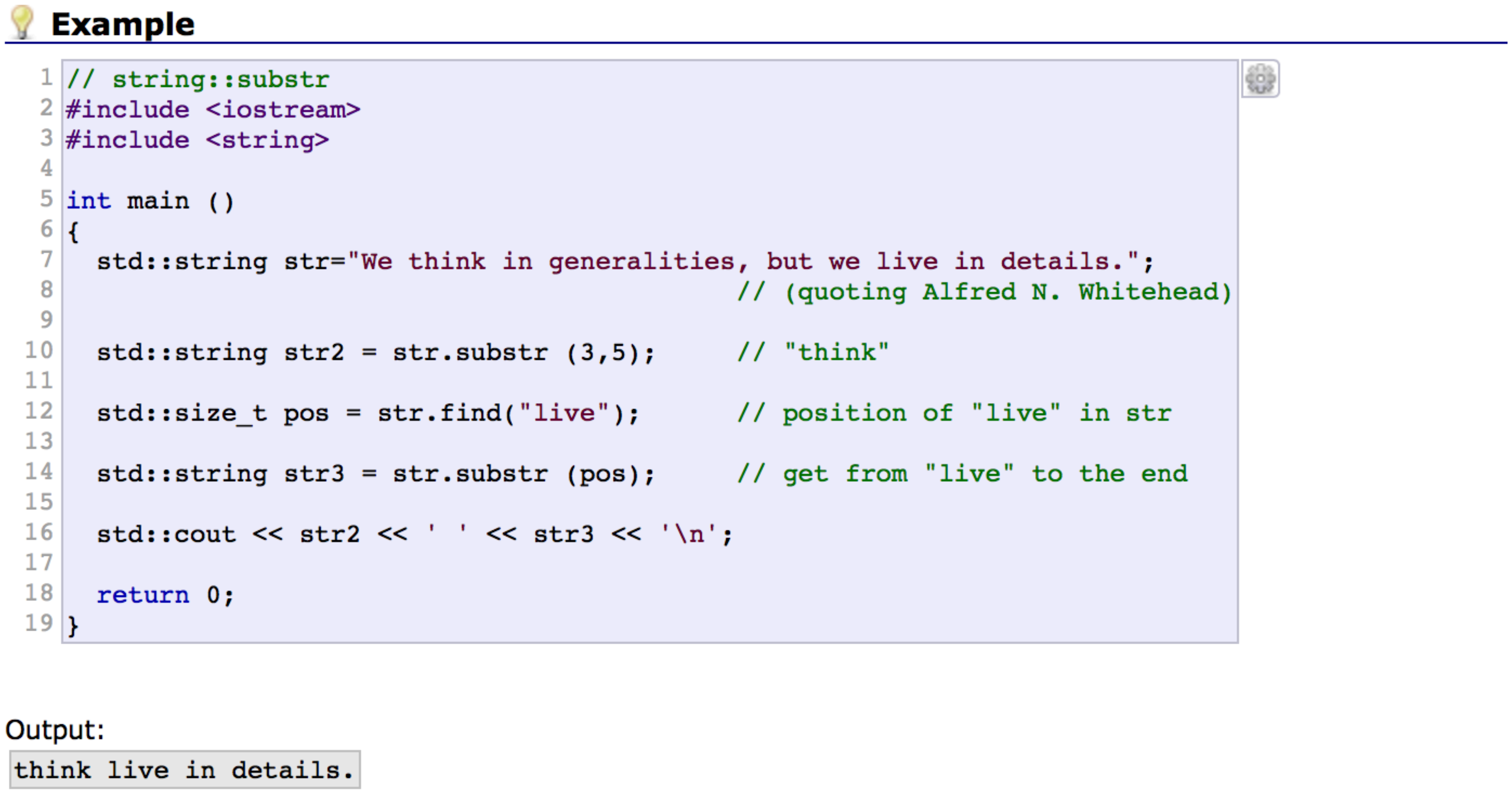
```
string substr (size_t pos = 0, size_t len = npos) const;
```

Generate substring

Returns a newly constructed `string` object with its value initialized to a copy of a substring of this object.

The substring is the portion of the object that starts at character position *pos* and spans *len* characters (or until the end of the string, whichever comes first).

例子



Example

```cpp
// string::substr
#include <iostream>
#include <string>

int main ()
{
  std::string str="We think in generalities, but we live in details.";
                                           // (quoting Alfred N. Whitehead)

  std::string str2 = str.substr (3,5);     // "think"

  std::size_t pos = str.find("live");      // position of "live" in str

  std::string str3 = str.substr (pos);     // get from "live" to the end

  std::cout << str2 << ' ' << str3 << '\n';

  return 0;
}
```

Output:

```
think live in details.
```

更多更详细

- http://www.cplusplus.com/reference/string/string/?kw=string