



举个栗子、

勿在浮沙筑高台、

长岛冰茶、

昵称: 长岛冰茶、
园龄: 1年5个月
粉丝: 13
关注: 6
+加关注

< 2019年8月 >						
日	一	二	三	四	五	六
28	29	30	31	1	2	3
4	5	6	7	8	9	10
11	12	13	14	15	16	17
18	19	20	21	22	23	24
25	26	27	28	29	30	31
1	2	3	4	5	6	7

搜索

找找看

谷歌搜索

常用链接

我的随笔
我的评论
我的参与
最新评论
我的标签

我的标签

HDU(3)
C++(3)
数据结构(3)
线段树(2)
中缀表达式(1)
G++(1)
ACM(1)
adb工具(1)
apktool(1)
Java(1)
更多

随笔档案

2018年7月 (1)
2018年6月 (1)
2018年4月 (3)
2018年3月 (9)

友情链接

长岛冰茶、
我的栈

最新评论

1. Re:C++ bitset 用法
能讲一下各个操作的复杂度吗?
--平凡的求知者
2. Re:C++ bitset 用法
写得很好, orz
--平凡的求知者
3. Re:C++ bitset 用法
顺便问下博主bitset互相异或的时间复杂度

C++ bitset 用法

C++的 bitset 在 bitset 头文件中, 它是一种类似数组的结构, 它的每一个元素只能是 0 或 1, 每个元素仅用 1 bit 空间。

下面是具体用法

构造函数

bitset常用构造函数有四种, 如下

```
bitset<4> bitset1;    //无参构造, 长度为4, 默认每一位为0

bitset<8> bitset2(12); //长度为8, 二进制保存, 前面用0补充

string s = "100101";
bitset<10> bitset3(s); //长度为10, 前面用0补充

char s2[] = "10101";
bitset<13> bitset4(s2); //长度为13, 前面用0补充

cout << bitset1 << endl; //0000
cout << bitset2 << endl; //00001100
cout << bitset3 << endl; //0000100101
cout << bitset4 << endl; //0000000010101
```

注意:

用字符串构造时, 字符串只能包含 '0' 或 '1', 否则会抛出异常。

构造时, 需在<>中表明bitset的大小(即size)。

在进行有参构造时, 若参数的二进制表示比bitset的size小, 则在前面用0补充(如上面的栗子); 若比bitsize大, 参数为整数时取后面部分, 参数为字符串时取前面部分(如下面栗子):

```
bitset<2> bitset1(12); //12的二进制为1100 (长度为4), 但bitset1的size=2, 只取后面部分, 即00

string s = "100101";
bitset<4> bitset2(s); //s的size=6, 而bitset的size=4, 只取前面部分, 即1001

char s2[] = "11101";
bitset<4> bitset3(s2); //与bitset2同理, 只取前面部分, 即1110

cout << bitset1 << endl; //00
cout << bitset2 << endl; //1001
cout << bitset3 << endl; //1110
```

可用的操作符

bitset对于二进制有位操作符, 具体如下

```
bitset<4> foo (string("1001"));
bitset<4> bar (string("0011"));

cout << (foo^=bar) << endl; // 1010 (foo对bar按位异或后赋值给foo)
cout << (foo&=bar) << endl; // 0010 (按位与后赋值给foo)
cout << (foo|=bar) << endl; // 0011 (按位或后赋值给foo)
```

--八水L酱

4. Re:C++ bitset 用法
谢谢博主，写的很好

--八水L酱

5. Re:C++ bitset 用法
mark。就是vc6的库比较垃圾，有any，
有none，没有all

--就是讲什么话都可以

阅读排行榜

- 1. C++ bitset 用法(20846)
- 2. C++ STL stack 用法(14048)
- 3. shunting-yard 调度场算法、中缀表达式转逆波兰表达式(745)
- 4. 【转】OJ提交时G++与C++的区别(348)
- 5. 你渴望力量吗(185)

评论排行榜

- 1. C++ bitset 用法(5)

推荐排行榜

- 1. C++ bitset 用法(14)
- 2. 博客园背景图(1)

```
cout << (foo<=2) << endl; // 1100 (左移2位, 低位补0, 有自身赋值)
cout << (foo>=1) << endl; // 0110 (右移1位, 高位补0, 有自身赋值)

cout << (~bar) << endl; // 1100 (按位取反)
cout << (bar<<1) << endl; // 0110 (左移, 不赋值)
cout << (bar>>1) << endl; // 0001 (右移, 不赋值)

cout << (foo==bar) << endl; // false (0110==0011为false)
cout << (foo!=bar) << endl; // true (0110!=0011为true)

cout << (foo&bar) << endl; // 0010 (按位与, 不赋值)
cout << (foo|bar) << endl; // 0111 (按位或, 不赋值)
cout << (foo^bar) << endl; // 0101 (按位异或, 不赋值)
```



此外，可以通过 [] 访问元素(类似数组)，注意最低位下标为 0，如下：

```
bitset<4> foo ("1011");

cout << foo[0] << endl; //1
cout << foo[1] << endl; //1
cout << foo[2] << endl; //0
```

当然，通过这种方式对某一位元素赋值也是可以的，栗子就不放了。

可用函数

bitset还支持一些有意思的函数，比如：



```
bitset<8> foo ("10011011");

cout << foo.count() << endl; //5 (count函数用来求bitset中1的位数, foo中共有5个1)
cout << foo.size() << endl; //8 (size函数用来求bitset的大小, 一共有8位)

cout << foo.test(0) << endl; //true (test函数用来查下标处的元素是0还是1, 并返回false或true, 此处foo[0]为1, 返回true)
cout << foo.test(2) << endl; //false (同理, foo[2]为0, 返回false)

cout << foo.any() << endl; //true (any函数检查bitset中是否有1)
cout << foo.none() << endl; //false (none函数检查bitset中是否没有1)
cout << foo.all() << endl; //false (all函数检查bitset中是全部为1)
```



补充说明一下：test函数会对下标越界作出检查，而通过 [] 访问元素却不会经过下标检查，所以，在两种方式通用的情况下，选择test函数更安全一些

另外，含有一些函数：



```
bitset<8> foo ("10011011");

cout << foo.flip(2) << endl; //10011111 (flip函数传参数时, 用于将参数位取反, 本行代码将foo下标2处"反转", 即0变1, 1变0)
cout << foo.flip() << endl; //01100000 (flip函数不指定参数时, 将bitset每一位全部取反)

cout << foo.set() << endl; //11111111 (set函数不指定参数时, 将bitset的每一位全部置为1)
cout << foo.set(3,0) << endl; //11110111 (set函数指定两位参数时, 将第一参数位的元素置为第二参数的值, 本行对foo的操作相当于foo[3]=0)
cout << foo.set(3) << endl; //11111111 (set函数只有一个参数时, 将参数下标处置为1)

cout << foo.reset(4) << endl; //11101111 (reset函数传一个参数时将参数下标处置为0)
cout << foo.reset() << endl; //00000000 (reset函数不传参数时将bitset的每一位全部置为0)
```



同样，它们也都会检查下标是否越界，如果越界就会抛出异常

最后，还有一些类型转换的函数，如下：



```
bitset<8> foo ("10011011");

string s = foo.to_string(); //将bitset转换成string类型
unsigned long a = foo.to_ulong(); //将bitset转换成unsigned long类型
unsigned long long b = foo.to_ullong(); //将bitset转换成unsigned long long类型
```

```
cout << s << endl;    //10011011
cout << a << endl;    //155
cout << b << endl;    //155
```



希望有帮助

标签: [C++](#)

好文置顶

关注我

收藏该文

长岛冰茶

关注 - 6

粉丝 - 13

+加关注

14 0

« 上一篇: [HDU 3486 Interviewe](#)
» 下一篇: [C++ STL stack 用法](#)

posted @ 2018-04-12 19:31 长岛冰茶、 阅读(20848) 评论(5) 编辑 收

评论列表

- #1楼 2019-03-10 11:01 就是讲什么话都可以

mark. 就是vc6的库比较垃圾, 有any, 有none, 没有all

支持(0) 反对(0)
- #2楼 2019-04-11 09:27 八水L酱

谢谢博主, 写的很好

支持(0) 反对(0)
- #3楼 2019-04-11 09:31 八水L酱

顺便问下博主bitset互相异或的时间复杂度

支持(0) 反对(0)
- #4楼 2019-07-25 10:32 平凡的求知者

写得很好, orz

支持(0) 反对(0)
- #5楼 2019-07-25 10:32 平凡的求知者

能讲一下各个操作的复杂度吗?

支持(0) 反对(0)

[刷新评论](#) [刷新页面](#) [返回顶部](#)

注册用户登录后才能发表评论, 请 [登录](#) 或 [注册](#), [访问网站首页](#)。
【推荐】超50万C++/C#源码: 大型实时仿真组态图形源码
【推荐】程序员问答平台, 解决您开发中遇到的技术难题

- 相关博文:
- [C++bitset用法](#)
 - [bitset用法](#)
 - [bitset](#)

- C++ bitset用法
- C++ bitset的用法

Copyright ©2019 长岛冰茶、