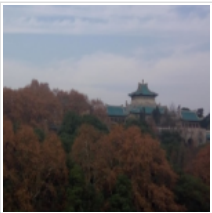


YYC的专栏

目录视图 摘要视图 RSS 订阅

个人资料



自由不死



访问： 177207次  
积分： 3126  
等级： **BLOG > 5**  
排名： 第10111名

原创： 112篇  
转载： 66篇  
译文： 0篇  
评论： 10条

文章搜索

文章分类

- C++ (102)
- C# (13)
- Android (8)
- JAVA (17)
- 数据库 (14)
- 离散数学实现 (3)
- 并发编程 (16)
- 服务Service (1)
- 数据结构 (21)
- 网络编程 (1)
- 网络工程 (23)
- 算法 (23)
- 成长心得 (2)
- OS (2)
- c (1)
- 模式分类 (1)

文章存档

- 2016年12月 (1)
- 2016年11月 (1)
- 2016年10月 (1)

C++标准库中bitset类型

标签: **c++** **bitset**

2014-02-23 13:13 1137人阅读 评论(0) 收藏 举报

分类:

C++ (101)

版权声明：本文为博主原创文章，未经博主允许不得转载。

虽然在C++的基本类型中，似乎没有二进制这个重要的类型，但是在C++标准库中却提供了能够处理二进制位的有序集合类型，这就是  
bitset类型，使用该类型时需要先包含该头文件并进行声明：

[cpp]

```
01. #include <bitset>
02. using namespace std;
```

该类型存储的是一个有序的二进制数据的集合，该集合的长度值需要在构造该类型时在尖括号中给出（因为该类型是模板类型）。  
一、bitset类型的初始化方式  
一共有四种初始化方式：

For Example:

[cpp]

```
01. //bitset对象的初始化
02. //方法一
03. bitset<32> b; //直接构造一个空对象，其中的每一位都默认为0
04. //方法二
05. unsigned long a =32;
06. bitset<32> b1(a); //构成一个unsigned long类型的二进制副本
07. //方法三
08. string a1("10001100001111001");
09. bitset<17> b2(a1); //构成一个含有二进制位串的副本
10. //方法四
11. bitset<10> b3(a1,1,10); //构成从a1的1位置开始的10个位的副本
```

二、对于该二进制集合的访问方式  
只能以位置的方式来访问其中每一个二进制元素，即以下标的方式来访问，但需注意的是像数组集合一样，该集合的下标也是从0开始的。

[cpp]

```
01. //按照下标的位置来访问该二进制集合
02. for(int i = 0;i<15;i++)
03. {
04.     cout<<b2[i]<<flush; //输出: 100111100001100
05. }
```

三、输出二进制位

2016年09月 (3)  
2015年01月 (12)

展开

阅读排行

- android-Service和Threa (3837)
- C#中FileStream文件流 (3699)
- SQL Server 登录连接失败 (3684)
- C#中流的读写器BinaryR (3383)
- 基于子网的Vlan划分配置 (3126)
- C++实现离散数学求主合 (2616)
- C++表达式求值（利用数 (2276)
- 以Android环境为例的多 (2246)
- C++数值类型极值的获 (2239)
- 路由器/三层交换机DHCF (2175)

评论排行

- 链队列的C++实现 (3)
- 线性表链式存储C++实现 (2)
- (二)：数据定义语言之 (1)
- C#中File静态类及其常用 (1)
- 循环赛日程表（非递归） (1)
- C++中程序化操作虚函数 (1)
- 算法与数据结构--图的实 (1)
- C#byte字节流读写乱码问 (0)
- C#中FileStream文件流 (0)
- C++实现离散数学“五个 (0)

推荐文章

- \* 5月书讯：流畅的Python，终于等到你！
- \* CSDN日报20170526 ——《程序员的时代焦虑与焦虑的缓解》
- \* Android中带你开发一款自动爆破签名校验工具kstools
- \* Android图片加载框架最全解析——深入探究Glide的缓存机制
- \* Android 热修复 Tinker Gradle Plugin解析
- \* Unity Shader-死亡溶解效果

最新评论

- 链队列的C++实现  
稚梟天卓: 个人见解，勿喷啦。其实，这不完全算C++实现吧，虽然是用了类模板，很不错！但是，可以自定义类啊，类函...
- 循环赛日程表（非递归）  
三名狂客: 不错！！！
- 算法与数据结构--图的实现、基本  
iwent2000: 好长
- 链队列的C++实现  
自由不死: @cjunkai327: 嗯，建议很有道理，谢谢提醒！
- 链队列的C++实现  
cjunkai327: 很全面。小小提个建议，出队后判断下是否是空指针，这样还要修改尾指针queue->rear=queue...
- 线性表链式存储C++实现  
自由不死: 对于LsList类型已有clear方法用于清理内存，而LNode节点类对象都（内部类）是在LsL...
- 线性表链式存储C++实现

与一般容器集合不同的是bitset二进制集合可以直接用输出操作符进行正题输出。

[cpp]

```
01. cout<<"b:"<<b<<endl;//输出: 00000000000000000000000000000000
02. cout<<"b1:"<<b1<<endl;//输出: 00000000000000000000000001000000
03. cout<<"b2:"<<b2<<endl;//输出: 10001100001111001
04. cout<<"b3:"<<b3<<endl;//输出: 0001100001
```

四、对于bitset对象的值的获取

有两种获取方式：分别是：转换成一个unsigned long的整数类型（调用成员函数：.to\_ulong()）  
转换成一个包含该集合所有二进制位的字符串（调用成员函数：.to\_string()）

[cpp]

```
01. string str("000110001");
02. bitset<10> bitver(str);
03. //转换成一个unsigned long的整数类型
04. unsigned long ul = bitver.to_ulong();//输出113
05. cout<<ul<<endl;
06. //转换成一个包含该集合所有二进制位的字符串
07. string ss = bitver.to_string();//输出000110001
08. cout<<ss<<endl;
```

五、利用内置成员函数对真个bitset对象进行设置

[cpp]

```
01. .set() //将所有的二进制位都设置为1
02. .reset() //将所有的二进制位都设置为0
03. .size() //返回集合的大小
04. .set[pos] //将pos位置的二进制位设置为1
05. .reset[pos] //将pos位置的二进制位设置为0
06. .flip () //将所有的二进制位都取反
07. .flip(pos) //将pos位置的二进制位取反
08. //。。。。
```

六、bitset集合最重要的特性是此类也支持内置的位操作符

但须注意的是：bitset集合按位运算的对象只能是也是一个bitset集合，而不能与一个整数进行按位运算。

For Example:

[cpp]

```
01. #include<iostream>
02. #include<bitset>
03. #include<string>
04. #include<vector>
05. using namespace std;
06. int main()
07. {
08.     string str("000110001");
09.     bitset<10> bitver(str);
10.     //取反
11.     bitver = ~bitver;
12.     cout<<bitver<<endl;//输出1110001110
13.     //左移两位
14.     bitset<10> b1 = bitver<<2;
15.     cout<<"b1"<<b1<<endl;//输出1000111000
16.     //右移两位
17.     bitset<10> b2 = bitver>>2;
18.     cout<<"b2"<<b2<<endl;//输出0011100011
19.     //按位与&
20.     unsigned long a = 1;
21.     bitset<10> a1(a);
22.     cout<<"a1"<<a1<<endl;//输出0000000001
23.     bitset<10> b3 = bitver & a1;
24.     cout<<"b3"<<b3<<endl;//输出0000000000
25.     return 0;
26. }
```

nscboy: 析构函数中为啥不把new的对象清理干净啊?会造成内存泄漏的.

C++中程序化操作虚函数列表实现alga\_1: good!

C#中File静态类及其常用静态方法孟子郢: 呵呵呵呵, 原来你还在这里

顶

1

踩

0

上一篇

C++程序员必看书单

下一篇

C++中的内存分配

相关文章推荐

- C++：标准库类型（string、vector、list、bitset）
- 标准库bitset类型
- 标准库bitset类型介绍
- C++复习 03 标准库类型
- c++ primer读书笔记-第三章 标准库类型

- C++Primer学习笔记第三章(3/18) 标准库类型
- 【C++ Primer】摘记--第3章 标准库类型
- 《C++Primer》第三章 标准库类型
- C++Primer 学习笔记 第三章（标准库类型）
- 《c++ primer》第三章标准库类型学习笔记整理\_201...

猜你在找

- c++面向对象前言及意见征集（来者不拒）视频教程
- 顾荣：开源大数据存储系统Alluxio（原Tachyon）的原理分
- Linux环境C++编程基础视频教程
- C++语言基础
- C++程序设计
- 深入浅出C++程序设计（基础篇）
- C++基础第一季
- C++标准模板库从入门到精通
- Visual Studio 2015开发C++程序的基本使用
- 华为工程师，带你实战C++

查看评论

暂无评论

您还没有登录,请[\[登录\]](#)或[\[注册\]](#)

\* 以上用户言论只代表其个人观点，不代表CSDN网站的观点或立场

核心技术类目

全部主题

Hadoop

AWS

移动游戏

Java

Android

iOS

Swift

智能硬件

Docker

OpenStack

VPN

Spark

ERP

IE10

Eclipse

CRM

JavaScript

数据库

Ubuntu

NFC

WAP

jQuery

BI

HTML5

Spring

Apache

.NET

API

HTML

SDK

IIS

Fedora

XML

LBS

Unity

Splashtop

UML

components

Windows Mobile

Rails

QEMU

KDE

Cassandra

CloudStack

FTC

coremail

OPhone

CouchBase

云计算

iOS6

Rackspace

Web App

SpringSide

Maemo

Compuware

大数据

aptech

Perl

Tornado

Ruby

Hibernate

ThinkPHP

HBase

Pure

Solr

Angular

Cloud Foundry

Redis

Scala

Django

Bootstrap

