参考：<https://zh.cppreference.com/w/%E9%A6%96%E9%A1%B5>

主要基于C++ 98

# 位集合

template<std::size\_t N>

class bitset;

类模板 bitset 表示一个 N 位的固定大小序列。可以用标准逻辑运算符操作位集，并将它与字符串和整数相互转换。

bitset 满足[可复制构造 (CopyConstructible)](https://zh.cppreference.com/w/cpp/named_req/CopyConstructible" \o "cpp/named req/CopyConstructible)及[可复制赋值 (CopyAssignable)](https://zh.cppreference.com/w/cpp/named_req/CopyAssignable" \o "cpp/named req/CopyAssignable)的要求。

## 成员函数

bitset();

或

bitset( unsigned long val );

或

template< class CharT, class Traits, class Alloc >

explicit bitset( const [std::basic\_string](http://zh.cppreference.com/w/cpp/string/basic_string)<CharT,Traits,Alloc>& str,  
                 typename [std::basic\_string](http://zh.cppreference.com/w/cpp/string/basic_string)<CharT,Traits,Alloc>::size\_type  
                     pos = 0,  
                 typename [std::basic\_string](http://zh.cppreference.com/w/cpp/string/basic_string)<CharT,Traits,Alloc>::size\_type

                     n = [std::basic\_string](http://zh.cppreference.com/w/cpp/string/basic_string)<CharT,Traits,Alloc>::npos);

1. 默认构造函数。构造所有位设为零的 bitset 。
2. 构造 bitset ，初始化其首（最右、最低） M 位位置为对应 val 的位值，其中 M 是 unsigned long long 的位数和正在构造的 bitset 中的位数 N 的较小者。若 M 小于 N （ bitset 长于 32 (C++11 前)64 (C++11 起) 位，对于典型的 unsigned long(C++11 起)long 实现），则剩余位位置被初始化为零。
3. 用 [std::basic\_string](https://zh.cppreference.com/w/cpp/string/basic_string" \o "cpp/string/basic string) str 中的字符构造 bitset 。能提供可选的起始位置 pos 和长度 n ，以及指代设置（ one ）和不设置（ zero ）位的替代值的字符。用 Traits::eq() 比较字符值。初始化字符串的有效长度是 min(n, str.size() - pos) 。 若 pos > str.size() ，则构造函数抛出 [std::out\_of\_range](https://zh.cppreference.com/w/cpp/error/out_of_range" \o "cpp/error/out of range) 。若 str 中检验到的任何字符非 zero 或 one ，则抛出 [std::invalid\_argument](https://zh.cppreference.com/w/cpp/error/invalid_argument" \o "cpp/error/invalid argument) 。

**说明：**

bitset(unsigned long val)对于C++11前使用的是32位，所以如果要传入一个64位的数据，则需要先转成二进制字符串再通过bitset(string)进行赋值，例如：

int64 lVal;

string strLow32 = bitset(lVal & 0xffffffff).toString();

string strHigh32 = biset(lVal >> 32).toString();

bitset(64)