

02G1 位图

#数据结构邓神

Bitmap用来描述一个整数集合

从 0 ~ U，没有重复的选择出来构成一个集合

```
// 注意集合中元素不重复
bool test(int k); // 是否存在
void set(int k); // 把K并入集合
void clear(int k); // 从集合中删除K
// 要求 O(1)
```

因为他最大限定为在 U，所以可以通过牺牲空间的方式，使得所有操作都在O(1)时间内完成

实现

学堂在线 xuetangx.com

```
❖ bool test( int k ) { return M[ k >> 3 ] & ( 0x80 >> (k & 0x07) ); }
❖ void set( int k ) { expand( k ); M[ k >> 3 ] |= ( 0x80 >> (k & 0x07) ); }
❖ void clear( int k ) { expand( k ); M[ k >> 3 ] &= ~( 0x80 >> (k & 0x07) ); }
```

Diagram illustrating the bit mask and memory layout:

0x80: [1] [0] [0] [0] [0] [0] [0] [0]

bit mask: [0] [0] [0] [0] [0] [1] [0] [0]

Memory layout: [0] [0] [0] [0] [0] [0/1] [0] [0]

Diagram showing the mapping of k to the bit mask:

k >> 3 → k / 8

k & 0x07 → k % 8

Diagram showing the bit mask and memory layout with a red circle around the bit mask and a green circle around the memory layout.

offset

Data Structures & Algorithms, Tsinghua University

```
#include "bits/stdc++.h"
class Bitmap{
private:
    int N;
    char * M;
public:
    Bitmap(int n = 8){
        M = new char[N = (n+7)/8];
        memset(M,0,N);
    }
```

```

}

~Bitmap(){
    delete [] M;
    M = nullptr;
}

bool test(int k){
    return M[k >> 3] & (0x80 >> (k & 0x07)); // k >> 3 == k/8    k & 0x07
    == k % 8
    // k & 0x07 生成 mask
}

void set(int k){
    M[k >> 3] |= (0x80 >> (k & 0x07));
}

void clear(int k){
    M[k >> 3] &= ~(0x80 >> (k & 0x07));
}
};


```

BitMap应用

小集合 + 大数据

老问题：

小集合 + 大数据

 学堂
xuetang

<p>❖ 老问题</p> <ul style="list-style-type: none"> - int A[n]的元素均取自[0, m) - 如何剔除其中的重复者？ <p>❖ 仿照Vector::deduplicate()改进版</p> <p>先排序，再扫描</p> <p>—— $O(n \log n + n)$ —— 毫无压力</p> <p>❖ 新特点：数据量虽大，但重复度极高</p> <ul style="list-style-type: none"> - 想想我们电脑里的mp3、mp4 - 还有，朋友圈 ... 	<p>❖ 比如</p> <p>$2^{24} = m \ll n = 10^{10}$</p> <p>亦即，10,000,000,000个24位无符号整数</p> <p>❖ 如果采用内部排序算法</p> <p>至少需要 $4 * n = 40GB$ 内存</p> <p>—— 否则，频繁的I/O将导致整体效率的低下</p> <p>❖ 那么</p> <p>$m \ll n$ 的条件，又应如何加以利用？</p>
--	---

素数问题

筛法：实现

```
❖ void Eratosthenes( int n, char * file ) {  
    Bitmap B( n );  
    B.set( 0 ); B.set( 1 );  
    for ( int i = 2; i < n; i++ )  
        if ( ! B.test( i ) )  
            for ( int j = 2*i; j < n; j += i )  
                B.set( j );  
    B.dump( file );  
}
```



Eratosthenes
(276 - 194 B.C.)