#include<cstdio>

#include<cstdlib>

#include<cstring>

#include<cmath>

#include<iostream>

#include<algorithm>

#include<string>

#include<queue>

#include<stack>

#include<functional>

#include<map>

#include<set>

#define rep(i,a,b) for(int i=a;i<=b;i++)

#define rep2(i,a,b) for(int i=a;i>=b;i--)

using namespace std;

typedef long long LL;

typedef unsigned long long ull;

const LL INF = 0x3f3f3f3f;

const LL mod = 1e9 + 7;

const int N = 2e6 + 10;

const double PI = acos(-1.0);

const LL INF = 1LL << 62 - 1LL;

const int inf = 1 << 31 - 1;

ll powmod(ll a, ll b) { ll res = 1;a %= mod; assert(b >= 0); for (;b;b >>= 1) { if (b & 1)res = res \* a % mod;a = a \* a % mod; }return res; }

ll gcd(ll a, ll b) { return b ? gcd(b, a % b) : a; }

void read(int &x){

x=0; char c=getchar();

while(c>'9'||c<'0') c=getchar();

while(c>='0'&&c<='9') x=x\*10+c-'0',c=getchar();

}

int main()

{

//freopen("./std.in", "r", stdin);

//std::ios::sync\_with\_stdio(false);

}

**二分查找**

pos = lower\_bound( number, number + 8, 3) - number，pos = 0.即number数组的下标为0的位置。

pos = lower\_bound( number, number + 8, 9) - number， pos = 1，即number数组的下标为1的位置（即10所在的位置）。

pos = lower\_bound( number, number + 8, 111) - number， pos = 8，即number数组的下标为8的位置（但下标上限为7，所以返回最后一个元素的下一个元素）。

upper\_bound():

头文件：#include<algorithm>

函数模板： 如binary\_search()

函数功能：函数upper\_bound()返回的在前闭后开区间查找的关键字的上界，返回大于val的第一个元素位置

 lower\_bound(val): 返回容器中第一个值【大于或等于】val的元素的iterator位置。

 upper\_bound(val): 返回容器中第一个值【大于】val的元素的iterator位置。

multiset<**int**> st;//可有重复元素的set

multiset<**int**>::iterator it= st.lower\_bound(-1 \* g[i].d);

**priority\_queue优先队列**

**#include<functional>**

priority\_queue<int, vector<int>, greater<int> > q;//小顶堆

priority\_queue<int> q;//大顶堆

struct Node{

int x, y;

Node( int a= 0, int b= 0 ):

x(a), y(b) {}

};

bool operator>( Node a, Node b ){//返回true，a的优先级大于b

//x大的排在队前部；x相同时，y大的排在队前部

if( a.x== b.x ) return a.y> b.y;

return a.x> b.x;

}

priority\_queue<Node,vector<Node>,greater<Node> > q;//重载

**Map**

map<string,int>date;int main()

{

date["7月30日"]=730;

cout<<date["7月30日"];

return 0;

}