// <https://github.com/flash7even> //

#include <bits/stdc++.h>  
#include <ext/pb\_ds/tree\_policy.hpp>  
#include <ext/pb\_ds/assoc\_container.hpp>  
  
using namespace std;  
using namespace \_\_gnu\_pbds;  
  
/\*\*  
 Change the typename T for pair , long long etc as necessary.  
  
 1. S.order\_of\_key( num ) : if present returns index(0 based)  
 else returns the position of first greater element.  
  
 2. \*S.find\_by\_order( pos ) : returns the element at position pos.  
 if no such position , returns 0  
  
 3. \*S.upper\_bound( num ) : returns the value just greater than num.  
 if no such value , then returns 0  
  
 4. \*S.lower\_bound( num ) : if num is present then returns num or returns  
 just the larger element than num.  
 if no such value returns 0  
  
 5. S.find( num ) != S.end() : boolean true or false , just like Set  
  
 6. S.erase( num ) : Erases number num , na thakle nai  
  
 7. orderset<int> :: iterator it;  
 we can use iterator such : it = S.begin() or it = S.end() , just like Set  
  
 Add more ..  
  
 \*/  
  
template < typename T >  
using orderset = tree<T,null\_type,less<T>,rb\_tree\_tag,tree\_order\_statistics\_node\_update>;

// act’s like multi set   
template < typename T >  
using orderset = tree<T,null\_type,less\_equal<T>,rb\_tree\_tag,tree\_order\_statistics\_node\_update>;  
  
orderset <int> S;  
  
int main( int argc, char const \*argv[] ) {  
  
 return 0;  
}

[Download as text](https://paste.ubuntu.com/p/qHwgSZ8NT7/plain/)