栈

意义

先入先出

用法

我们创建一个变量名为 `stk` ，储存数据类型为 `int` 的栈

```
stack<int> stk;
```

常见基础函数

有一个 `int` 变量为 `x`
`.pop()` 栈顶弹出一次
`.push(x)` 栈顶加入 `x`
`.top()` 返回栈顶
`.clear()` 清空栈
`.size()` 返回栈里元素数量
`.empty()` 返回栈是否为空

示例

```
...
int main () {
    stack<int> stk;
    stk.push(1); // stk(top → bottom): 1
    stk.push(2); // stk(top → bottom): 2 1
    cout << stk.size() << " " << stk.empty() << endl; // 2 0
    cout << stk.top() << endl; // 2

    stk.pop(); // stk(top → bottom): 1
    cout << stk.top() << endl; // 1

    stk.clear(); // stk(top → bottom):
    cout << stk.size() << " " << stk.empty() << endl; // 0 1
}
```

队列

意义

先入后出

用法

建立变量名为 `que` 储存 `int` 的队列

```
queue<int> que;
```

基本和栈一致，但是运用按队列的规则（先入后出）走
即不同的为
`.push()` 是压入队尾
`.pop()` 弹出队尾
`.front()` 返回队首
`.back()` 返回队尾

堆

意义

内部自动排序

set

建立变量名为 `st` 储存 `int` 的堆（升序，自动去重

```
set<int> st;
```

不去重

```
multiset<int> mst;
```

常见基础函数

`.insert` 插入元素
`.erase` 删除元素
`.size()` `.empty()` `.clear()` 同上

示例

```cpp
int main () {
    set<int> st;
    st.insert(1000);
    st.insert(1);
    st.insert(500);
    // st(begin → end): 1 500 1000
}
```

priority_queue

建立变量名为 `pque` 储存 `int` 的优先队列（降序

```cpp
priority_queue<int> pque;
```

升序要用 `priority<int, vector<int>, greater<int> >`

常见基础函数

`.top()` `.push()` `.pop()` `.size()` `.empty()` 与栈相同
但无 `.clear()`

示例

```cpp
int main () {
    priority_queue<int> pque;
    pque.push(1000);
    pque.push(1);
    pque.push(500);
    // pque(top → bottom): 1000 500 1
}
```

01 串

意义

每个位置只为 `0` 或 `1`
支持位运算

用法

建立变量名为 `bst` 长为 `N` 的 01串

```cpp
bitset<N> bst;
```

.any() 是否存在1
.count() 1的个数
.set() 全部变成1
.set(pos) pos位置变成1
.reset() 全部变成0
.reset(pos) pos位置变成0
.flip() 全部取反
.flip(pos) pos位置取反

小技巧

集合 α 与集合 β 相同的元素个数
输入 $|\alpha|$ 与 $\{\alpha\}$ ，$|\beta|$ 与 $\{\beta\}$
示例:
输入
3 1 3 7
5 1 9 4 3
输出
2

```
...
int main () {
        bitset<10> bsta, bstb;
        int na; cin >> na;
        for ( int i = 0; i < na; i ++ ) {
                int x; cin >> x;
                bsta.set(x);
        }
        int nb; cin >> nb;
        for ( int i = 0; i < nb; i ++ ) {
                int x; cin >> x;
                bstb.set(x);
        }
        cout << (bsta & bstb).count() << endl;
}
```