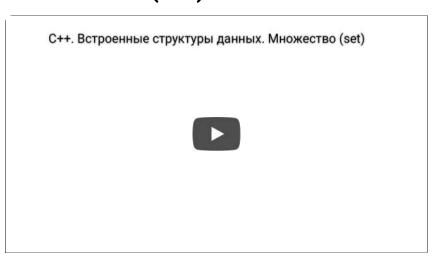
Множество (Set)



Множество в языке C++ (set) — структура данных, в которой содержится несколько элементов одного типа в отсортированном порядке без повторений.

Рассмотрим простой пример использования сета:

```
#include <iostream>
#include <set>
using namespace std;
int main()
{
    set<int> p{3, 2, 5, 7, 11}; // создаем сет р из элементов: 3, 2, 5, 7, 11
    p.insert(13); // добавляем в сет число 13
    p.erase(5); // удаляем из сета число 5
    p.erase(p.begin()); // удаляем из сета наименьший элемент (2)
    cout << p.size() << endl; // выводим размер сета на экран (4)
    return 0;
}</pre>
```

Итак, set в C++ поддерживает следующие операции:

| Метод | Описание метода |
|---------------|---|
| insert(value) | Добавляет элемент со значением value в сет. Если элемент с таким значением уже присутствует в сете, то ничего не произойдёт, элемент всё ещё будет встречаться в сете один раз. |
| erase(value) | Удаляет элемент со значением value из сета. Если такого элемента нет, то ничего не произойдёт. |
| erase(it) | Удаляет элемент, соответствующий lt. Если данный итератор указывает на несуществующий элемент, то произойдёт ошибка выполнения программы |
| size() | Возвращает количество элементов в сете |
| count(value) | Количество вхождений элементов со значением value в сет. Может быть 0 или 1. |

При реализации сета в C++ используются бинарные деревья поиска, поэтому все операции поиска, добавления, удаления элемента имеют вычислительную сложность $O(\log n)$.

Рассмотрим подробнее работу с итераторами внутри сета. Создать итератор можно так:

```
set<int>::iterator it = S.begin(); // создаём итератор it, указывающий на начало сета S
```

Можно создать итератор иначе:

```
auto it = S.begin();
```

Обратиться к итератору по значению можно таким образом:

```
\mathtt{cout} << \star(\mathtt{--S.end}()) << \mathtt{endl}; // выводим на экран значение последнего (максимального) элемента
```

Удалим минимальный и максимальный элемент из сета:

```
S.erase(S.begin());
S.erase(--S.end());
```

Итераторы также помогают сделать цикл, который будет проходить по всем элементам сета:

```
for (auto it = S.begin(); it != S.end(); ++it)
    cout << *it << " ";</pre>
```

В данном примере значения элементов сета будут выведены в порядке возрастания. Есть ещё один способ организовать цикл по сету:

```
for (auto elem: S)
   cout << elem << " ";</pre>
```

Задача "Похожие массивы"

Назовём два массива похожими, если они состоят из одних и тех же элементов (без учёта кратности). По двум данным массивам выясните, похожие они или нет.

Решение

```
#include <iostream>
#include <set>
using namespace std;
int main()
{
    set<int> a, b;
    int n, elem;
    cin >> n;
    for (int i = 0; i < n; ++i)
    {
        cin >> elem;
        a.insert(elem);
    }
    int m;
    cin >> m;
    for (int i = 0; i < m; ++i)</pre>
```

```
{
    cin >> elem;
    b.insert(elem);
}
if (a == b)
    cout << "YES" << endl;
else
    cout << "NO" << endl;
return 0;
}</pre>
```