std::set trong C++

Giới thiệu chung

- std::set là một cấu trúc dữ liệu chứa các phần tử khác nhau cùng kiểu dữ liệu, có sắp xếp thứ tự và hỗ trợ 3 phương thức chính, tất cả trong thời gian O(log(n)):
  - Kiểm tra phần tử có trong set hay không.
  - Thêm phần tử.
  - Xoá phần tử.
- Bài viết này sử dụng namespace std.

Khai báo

- Cú pháp khai báo của set trong C++
  - `set<data_type, comparator> name;`
    - data_type: kiểu dữ liệu của set
    - comparator: hàm so sánh của người dùng, giúp cho set sắp xếp theo thứ tự của người dùng. Có thể để trống, khi này mặc định set sẽ sắp sếp tăng.
    - name: đây là tên của set, tuân thủ theo quy tắc đặt tên trong ngôn ngữ lập trình.
- Khởi tạo:
  - Cách 1: Khởi tạo trống: set<data_type> name;
  - Cách 2: Khởi tạo trên khoảng: set<data_type> name(it1, it2);
    - Ví dụ cụ thể đối với mảng và vector:

```
vector<int> a1 = {1,4,2,3,4};
int a2[] = {1,4,2,3,4};
set<int,cmp> s1(a1.begin(), a1.end());
set<int,cmp> s2(a2,a2+5);

for (int i:s1) cout << i << ' ';
cout << '\n';
for (int i:s2) cout << i << ' ';
cout << '\n
```

    - Đối với mảng và vector đã sắp xếp tăng, độ phức tạp O(k): k là khoảng cách giữa it1 và it2.
    - Đối với mảng chưa sắp sếp thì độ phức tạp là O(k*log(k)).

- Cách 3: copy từ set cũ có cùng kiểu dữ liệu:
  - Kiểu 1: set<data_type> s1 (old_set); Độ phức tạp O(n).
  - Kiểu 2: set<data_type> s1 = old_set; Độ phức tạp O(n).

Các phương thức của set

Cách sử dụng **phương thức**:

- setName.phuongThuc()

Iterators: trả về set<data_type, comparator>::iterator.

- begin(): trả về một iterator trỏ ngay tại phần tử đầu tiên.

- end(): trả về một iterator trỏ ngay sau phần tử cuối cùng (trỏ vào NULL).

- Ngoài ra còn có rbegin(), rend(), cbegin(), cend(), crbegin(), crend(). Bạn đọc vui lòng tự tìm hiểu.

Capacity:

- empty(): trả về giá trị boolean 1 nếu set rỗng, trả về 0 nếu set không rỗng. O(1).

- size(): trả về số phần tử đang có trong set. O(1).

Modifiers:

- insert(value): thêm một giá trị vào set. Nếu giá trị đã tồn tại thì sẽ không được thêm vào set một lần nữa. O(log(n)).

- erase(value): xoá một giá trị nếu nó có tồn tại trong set. Nếu giá trị không tồn tại trong set thì sẽ không xoá. O(log(n)). **Thú vị một chỗ là nó cũng sẽ trả về giá trị value.**

- erase(set::iterator): xoá phần tử mà iterator đang trỏ vào. O(1).

- erase(first_iterator, second_iterator): xoá các phần tử được trỏ từ iterator first_it đến **trước** iterator second_it (Tức là nó sẽ không xoá phần tử được iterator second_it trỏ vào). O(k): trong đó k là khoảng cách giữa 2 iterator. Lưu ý là nếu second_iterator < first_iterator thì sẽ báo lỗi.

- clear(): Xoá hết tất cả phần tử trong set. Độ phức tạp: O(n).

Tìm kiếm:

- find(value)

Cách viết comparator (hàm so sánh cho set).

Các tính chất của set. Độ phức tạp thời gian

Ưu điểm, nhược điểm và một số lưu ý

So sánh set với multiset

So sánh set với vector

So sánh set với mảng thông thường

So sánh set với linked list

So sánh set với map

So sánh set với unordered set

Ứng dụng của set trong giải thuật

Các bài tập tự luyện