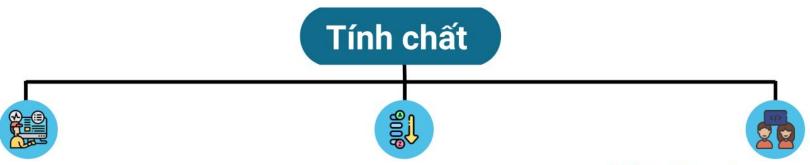
# MAP

#include <map>

### Map

#### Khái niệm

Map là container giúp lưu các phần tử theo cặp key, value (khóa - giá trị). Mỗi giá trị của key sẽ ánh xạ sang một value tương ứng. So với Set thì Map thậm chí còn mạnh mẽ và giải quyết được nhiều vấn đề hơn.



Các key trong map là những giá trị riêng biệt, không có 2 key nào có giá trị giống nhau, value thì có thể trùng nhau.

Các cặp phần tử trong map được sắp xếp theo thứ tự tăng dần của key. Mỗi phần tử trong map thực chất là một pair, với first lưu key và second lưu value.

### Map

### **CÚ PHÁP**

map <key\_data\_type, value\_data\_type> map\_name;

SỬ DỤNG MAP

/01 Các bài toán liên quan tới tần suất của các phần tử.

/02 Các bài toán cần tìm kiếm, thêm, xóa một cách nhanh chóng.

/03 Dùng map thay cho các bài toán sử dụng mảng đánh dấu khi dữ liệu không đẹp.



#### Thêm một phần tử vào trong map:

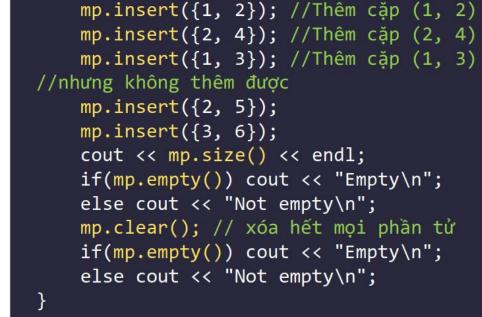
- Dùng hàm insert
- Dùng cú pháp map[key] = value nếu key chưa tồn tại trong map, hoặc sẽ thay đổi value nếu key đã tồn tại.

```
#include <bits/stdc++.h>
using namespace std;
int main(){
   map<int, int> mp;
   mp.insert({1, 2}); //Thêm cặp (1,2)
   mp.insert({2, 4}); //Thêm cặp (2,4)
   mp.insert({1, 3}); //Không thêm được cặp (1,3)
   mp[3] = 10; // thêm cặp (3,10)
   mp[2] = 5; //Thay đổi cặp (2,4) thành (2,5)
         mp = \{(1,2),(2,5),(3,10)\}
```

Hàm size: trả về số lượng phần tử trong map.

```
#include <bits/stdc++.h>
using namespace std;
int main(){
    map<int, int> mp;
    mp.insert({1, 2}); //Thêm cặp (1, 2)
    mp.insert({2, 4}); //Thêm cặp (2, 4)
    mp.insert({1, 3}); //Thêm cặp (1, 3)
//nhưng không thêm được
    mp[3] = 10; // thêm cặp (3, 10)
    mp[2] = 5; //Thay đổi cặp (2, 4)
//thành (2, 5)
    cout << mp.size() << endl;</pre>
             OUTPUT: 3
```

Hàm empty: kiểm tra map rỗng, nếu rỗng trả về true, ngược lại trả về false. Hàm clear: xóa mọi phần tử trong map.



Not empty!

Empty!

#include <bits/stdc++.h>

map<int, int> mp;

**OUTPUT: 3** 

using namespace std;

int main(){

**Hàm find:** Tìm kiếm sự xuất hiện của một key nào đó trong map. Độ phức tạp là O(logN).

Hàm này trả về iterator tới cặp phần tử nếu nó tìm thấy, ngược lại nó trả về iterator end() của map khi giá trị key tìm kiếm không tồn tại trong map.

```
#include <bits/stdc++.h>
using namespace std;
int main(){
   map<int, int> mp;
    mp.insert({1, 2}); //Thêm cặp (1, 2)
    mp.insert({2, 4}); //Thêm cặp (2, 4)
    mp.insert({3, 6}); //Thêm cặp (3, 6)
    auto it = mp.find(1);
    if(it == mp.end()){
        cout << "NOT FOUND KEY\n";</pre>
    else{
        cout << (*it).first << ' ' <<</pre>
(*it).second << endl;
          OUTPUT: 12
```

**Hàm count:** Hàm này dùng để đếm số lần xuất hiện của 1 key nào đó trong map. Đối với map hàm count trả về 0 hoặc 1, có thể sử dụng hàm này để thay cho hàm find. Độ phức tạp O(logN).

```
#include <bits/stdc++.h>
using namespace std;
int main(){
    map<int, int> mp;
    mp.insert({1, 2}); //Thêm cặp (1, 2)
    mp.insert({2, 4}); //Thêm cặp (2, 4)
    mp.insert({3, 6}); //Thêm cặp (3, 6)
    cout << mp.count(1) << endl;</pre>
    cout << mp.count(5) << endl;</pre>
        OUTPUT: 1
```

Hàm erase: Xóa một phần tử khỏi map với độ phức tạp là O(logN), trước khi sử dụng hàm erase hãy đảm bảo phần tử bạn cần xóa tồn tại trong map nếu không sẽ xảy ra lỗi runtime error.

#### Xóa thông qua giá trị của key

```
#include <bits/stdc++.h>
using namespace std;
int main(){
   map<int, int> mp;
   mp.insert({1, 2}); //Thêm cặp (1, 2)
    mp.insert({2, 4}); //Thêm cặp (2, 4)
   mp.insert({3, 6}); //Thêm cặp (3, 6)
   mp.erase(1);
    for(auto it : mp){
        cout << it.first << ' ' <<
it.second << endl;</pre>
        OUTPUT: 24
                   36
```

Hàm erase: Xóa một phần tử khỏi map với độ phức tạp là O(logN), trước khi sử dụng hàm erase hãy đảm bảo phần tử bạn cần xóa tồn tại trong map nếu không sẽ xảy ra lỗi runtime error.

#### Xóa thông qua iterator

```
#include <bits/stdc++.h>
using namespace std;
int main(){
    map<int, int> mp;
    mp.insert({1, 2}); //Thêm cặp (1, 2)
    mp.insert({2, 4}); //Thêm cặp (2, 4)
    mp.insert({3, 6}); //Thêm cặp (3, 6)
    auto it = mp.find(3);
    if(it != mp.end()){
        mp.erase(it);
    for(auto it : mp){
        cout << it.first << ' ' <<
it.second << endl;</pre>
        OUTPUT: 12
```

### Duyệt map

#### Duyệt map



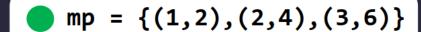
#### Duyệt bằng for each:

```
//for each
for(pair<int, int> it : mp){
    cout << it.first << ' ' << it.second << endl;
}
//for each dùng auto thay cho pair
for(auto it : mp){
    cout << it.first << ' ' << it.second << endl;
}
//Từ key suy ra value
for(auto it : mp){
    int key = it.first;
    cout << key << ' ' << mp[key] << endl;
}</pre>
```

### Duyệt map

#### Duyệt map





### **Duyệt bằng iterator:**

```
//Dùng iterator
for(map<int, int>::iterator it = mp.begin(); it !=
mp.end(); ++it){
   cout << (*it).first << ' ' << (*it).second << endl;
}
//Thay bằng auto cho tiện
for(auto it = mp.begin(); it != mp.end(); ++it){
   cout << (*it).first << ' ' << (*it).second << endl;
}</pre>
```