Contents

[CHƯƠNG 1: QUI ƯỚC ĐẶT TÊN TRONG C++ 2](#_Toc147789934)

[1.1. Tên biến 2](#_Toc147789935)

[1.2. Tên tệp 2](#_Toc147789936)

[1.3. Tên hàm 2](#_Toc147789937)

[1.4. Tên class 2](#_Toc147789938)

[1.5. Tên biến 2](#_Toc147789939)

[CHƯƠNG 2: VECTOR TRONG C++ 3](#_Toc147789940)

[2.1. Định nghĩa 3](#_Toc147789941)

[2.2. Code 4](#_Toc147789942)

[CHƯƠNG 3: PAIR TRONG C++ 4](#_Toc147789943)

[3.1. Định nghĩa 4](#_Toc147789944)

[3.1.1. Pair 4](#_Toc147789945)

[3.1.2. Tuple 5](#_Toc147789946)

[3.2. Code 5](#_Toc147789947)

[CHƯƠNG 4: SET TRONG C++ 6](#_Toc147789948)

[4.1. Định nghĩa 6](#_Toc147789949)

[4.2. Multiset 7](#_Toc147789950)

[CHƯƠNG 5: MAP TRONG C++ 7](#_Toc147789951)

[5.1. Định nghĩa 7](#_Toc147789952)

# QUI ƯỚC ĐẶT TÊN TRONG C++

## Tên biến

Các biến được khái báo là const thì nên được viết hoa toàn bộ hoặc đặt tên chõ chữ k đứng trước tên biến.

VD:

const kDay = 7;

const kMonth\_in\_year = 12;

## Tên tệp

Tất cả tên tệp phải là chữ thường và có thể bao gồm dấu gạch dưới (\_) hoặc dấu gạch ngang (-). Tuân theo quy ước mà dự án của bạn sử dụng và đặt tên thật cụ thế.

VD:

my\_useful\_class.cpp

my-useful-class.cpp

## Tên hàm

Tên hàm thường sẽ viết hoa chữ cái đầu của mỗi từ mới

VD:

void codeLearn

void enterInfoStudent

void oneTwo

## Tên class

Tên class sẽ là các từ và viết hoa chữ cái đầu của tất cả các từ

VD:

class Student {};

class CodeLearn {};

## Tên biến

Tên biến nên đặt là từ có nghĩa, ngắn gọn, dễ hiểu, có thể viết thường, sử dụng dấu gạch ngang, hoặc viết hoa khi có viết từ mới những thông thường sẽ dùng dấu gạch ngang, miễn sao là tuân thủ các quy tắc đã thảo luận phía trên

VD:

char fullName[10];

int luong;

int code\_learn;

# VECTOR TRONG C++

## Định nghĩa

Vector có thể coi là một mảng động mà có thể thay đổi tùy ý kích thước mà không cần khai báo kích thước trước. Một tính chất nữa của vector là nó sẽ giải phóng bộ nhớ của những phần tử bị xóa.

Cú pháp khai báo vector : **vector <kieu du lieu> ten\_vector;**

Có thể truy cập các phần tử vector thông qua chỉ số của nó

Hàm push\_back() dùng để đẩy dữ liệu vào cuối vector

Truy cập đến phần tử đầu: v[0]

Truy cập đến phần tử cuối: v.back(), v[v.size()-1]

Khai báo một vector có n phần tử đều có cùng một giá trị: **vector <kieu du lieu> ten\_vector (so\_luong, gia trị);**

VD:

vector <int> v(n,100);

Duyệt các phần tử:

Cách 1:

for (int i = 0; i <v.size(); ++i){

cout<< v[i] << endl;

}

Cách 2:

for (int x ; v){

cout<< x <<endl;

}

Cách 3: Duyệt bằng interators: con trỏ v.begin() tới con trỏ v.end(). v.end() là phần tử sau phần tử cuối cùng.

for (vector<int> :: interator it = v.begin(); it != v.end(); ++it){

cout<< \*it <<endl; // it là con trỏ cho nên muốn in ra giá trị của địa chỉ mà con trỏ đang trỏ tới phải thêm dấu hoa thị.

}

Cách 4: for (auto it = v.begin(); it != v.end(); it++){

cout<< \*it <<endl;

}

## Code

#include <bits/stdc++.h>

using namespace std;

int main(){

vector <int> v;

v.push\_back(5);

v.push\_back(10);

v.push\_back(20);

cout<< v.size();

cout << “Phan tu dau tien trong vector la: “ <<v[0]<<endl;

cout << “Phan tu cuoi cung trong vector la: “<< v.back() <<endl;

return 0;

}

# PAIR TRONG C++

## Định nghĩa

### Pair

Pair để lưu một cặp số đi cùng nhau, hoặc một chuỗi với một số, hoặc một chuỗi với một chuỗi.

Cú pháp khai báo pair: pair <datatype1, datatype2> v = make\_pair (first\_value, sencond\_value);

hoặc

pair <datatype1, datatype2> v {first\_value, sencond\_value};

VD:

pair <long long, long long> v = make\_pair (100,200);

pair <string, int> v = make\_pair (“Hello world” ,200);

pair <string, string> v = make\_pair (“Vietnam”, “Thailand”);

Truy cập đến phần tử thứ nhất và thứ hai: v.first, v.second

### Tuple

Tuple để lưu một bộ dữ liệu đi cùng nhau

Cú pháp khai báo tuple: tuple <datatype1, datatype2, …, datatypeN> t {value1, value2, …, valueN};

hoặc tuple <datatype1, datatype2, …, datatypeN> t = make\_tuple(value1, value2, …, valueN);

VD:

tuple <int, int, int> t(10,20,30);

Truy cập tới từng phần tử trong tuple: get<index>(t)

## Code

#include <bits/stdc++.h>

using namespace std;

int main(){

pair <string, int> sinh vien = make\_pair(“Nguyen Tuong Minh”, 1910343);

tuple <float, float, float> diem (9.6, 9.8, 10.0)

// In thong tin sinh vien

cout<< “Ho va ten sinh vien la: “<< sinhvien.first<<endl;

// In ma so sinh vien

cout <<” MSSV: “<<sinhvien.second <<endl;

// In diem trung binh ba mon

for (int i = 0; i<tuple.size(); i++){

cout<<” Diem trung binh mon thu “<< i<< “ la: “<< get<i>(diem) <<endl;

}

return 0;

}

# SET TRONG C++

## Định nghĩa

Set là một danh sách chứa các phần tử không trùng lặp theo một thứ tự từ bé đến lớn (số), tăng dần theo từ điển (chuỗi).

Set được sử dụng để tìm kiếm nhanh. đếm số lượng phần tử có giá trị tương ứng trong mảng

Include thư viện: #include <set>

Độ phức tạp thuật toán của các hàm trong thư viện set rất nhanh: O(logn)

Khai báo một set: set <datatype> s;

Thêm một phần tử trong set: s.insert(value);

Trả về số lượng phần tử trong set: s.size();

Hàm để tìm kiếm một giá trị có nằm trong set hay không:

s.count(value);

Nếu có giá trị tron set, hàm s.count() sẽ trả về 1, ngược lại sẽ trả về 0

s.find(value);

Nếu có giá trị trong set, sẽ trả về interator trỏ đến giá trị đó trong set, còn nếu không sẽ trỏ đến s.end()

Hàm để xóa phần tử trong set: s.erase(value); VD: s.erase(5);

hoặc có thể xóa bằng iterator: s.erase(iterator); VD: s.erase(s.find(5))

Cách duyệt qua các phần tử trong set:

Cách 1:

for (int x: s){ cout<<x<<endl;}

Cách 2:

for (auto x: s) {cout<<x<<endl;}

Cách 3:

for (set<int>:: iterator it = s.begin(); it != s.end(); it ++){cout << \*it << endl;}

Cách 4:

for (auto it = s.begin(); it != s.end(); it ++){ cout << \*it << endl;}

## Multiset

Là set cho phép giữ những giá trị trùng lặp nhưng vẫn có thứ tự.

Cú pháp khai báo của multiset: multiset <datatype> ten\_multiset

Các hàm giống với set, tuy nhiên sẽ có các lưu ý sau:

s.erase(value) sẽ xóa hết các phần tử có value

Muốn xóa một phần tử phải thông qua iterator.

VD:

multiset <int> ms;

ms.insert(100);

ms.insert(100);

ms.insert(200);

ms.insert(300);

ms.insert(200);

for (auto x: ms){ cout <<x <<endl;}

ms.erase(ms.begin()+2);

# MAP TRONG C++

## Định nghĩa

Map là một tập các cặp dữ liệu chứa một key và một value. Trong đó, key sẽ không trùng lặp, tuy nhiên value thì có thể trùng lặp.

Các hàm thao tác của map đều có độ phức tạp là O(logn)

Thư viện: #include <map>

Cú pháp khai báo map: map<datatype1, datatype2> mp;

Cách để chèn một phần tử vào map:

Cách 1:

mp.insert({3,400});

mp.insert({1,200});

Cách 2:

mp[3] = 400;

mp[1] = 200;

Cách duyệt qua các phần tử của map:

Cách 1:

for (pair<int. int> x:mp){cout<<x.first <<” “<< x.second<<endl;}

Cách 2:

for (auto x: mp){ cout<<x.first <<” “<< x.second<<endl;}

Cách 3:

for (map<int, int>:: iterator it = mp.begin(); it <mp.end(); it++){cout<< \*it.first << “ ”<< \*it.second <<endl;}

Cách 4:

for (auto it = mp.begin(); it <mp.end(); it++){cout<< \*it.first << “ ”<< \*it.second <<endl;}

Các hàm count, find, erase giống với set.