**c++优先队列priority\_queue（自定义比较函数）**

可以使用现成的

less<T>来定义大顶堆

greater<T>来定义小顶堆

从文档出可以看到，传入的可以是 函数指针或者 函数对象（类对操作符()进行了重载，）

参考链接：函数指针和函数对象

参考链接：decltype

方式一：struct重载运算符()

通过struct重载()操作符，定义了一个函数对象

struct cmp{

bool operator()(vector<int>&a,vector<int>&b){

return a[0]>b[0];

}

};

priority\_queue<vector<int>,vector<vector<int>>,cmp> q;//小顶堆

**这是属于传入 函数对象 的方式**

## 方式二：class[重载运算符](https://so.csdn.net/so/search?q=%E9%87%8D%E8%BD%BD%E8%BF%90%E7%AE%97%E7%AC%A6&spm=1001.2101.3001.7020)()

通过class重载()操作符，定义了一个函数对象  
注意要加public

class cmp{

public:

bool operator()(vector<int>&a,vector<int>&b){

return a[0]>b[0];

}

};

priority\_queue<vector<int>,vector<vector<int>>,cmp> q;//小顶堆

**这是属于传入 函数对象 的方式**

## 方式三：定义函数

首先定义一个比较函数

bool cmp(vector<int>&a,vector<int>&b){

return a[0]>b[0];

}

decltype()是用于获得函数指针的 类型的。在模板中也要传入它们的类型。  
decltype()要传入的是一个对象的地址，因此需要对cmp加取值符，&cmp为对象的地址  
在这里插入图片描述  
因此可以由函数地址cmp 转为函数指针 类型 decltype(&cmp)

priority\_queue<vector<int>,vector<vector<int>>,decltype(&cmp)> q(cmp);//小顶堆

**写法一：**







#include<iostream>

#include<vector>

#include<queue>

using namespace std;

int main(){

auto cmp=[](vector<int>&a,vector<int>&b)->bool{

return a[0]>b[0];

};

priority\_queue<vector<int>,vector<vector<int>>,decltype(cmp)> q(cmp);//小顶堆

q.push({3,4});

q.push({1,2});

q.push({5,6});

vector<int> vec=q.top();

q.pop();

cout<<vec[0]<<endl;

cout<<vec[1]<<endl;

system("pause");

return 0;

}

