

65 Sorting in C++

数据结构决定了存储数据的方式，而数据在C++编程中非常重要。

假设我有一个整数向量或者一个整数数组，我想让它们按照值大小或者某种谓词排序，怎么让C++帮我来做呢？

显然你可以自己写算法，比如冒泡排序、快速排序等等，或者任何一种遍历列表并对元素排序的算法，让其按照你希望的排序方式。但有些情况下，比如在你处理C++的内置集合类型，如 `std::vector`，你没有必要自己写一个算法，你可以让C++库帮你排序，所以这里我们关心的是 `std::sort`。

1. `std::sort`

这是C++内置的排序函数，它可以为你提供给它的任何类型的迭代器执行排序。

```
C++  
  
#include <iostream>  
#include <vector>  
#include <algorithm>  
  
int main()  
{  
    std::vector<int> values = { 3,5,1,4,2 };  
    std::sort(values.begin(), values.end()); // 如果我们不提供任何类型的谓词，即不给它提供一个  
    用来排序的函数，对于整数它就会按升序排序  
  
    for (int value : values)  
        std::cout << value << std::endl;  
    std::cin.get();  
}
```

```
for (int value : values)  
    std::cout << value << std::endl;  
std::cin.get();
```

1
2
3
4
5

2. 使用lambda

如果你想要让它按照特定的方式排序，你可以传入一个函数，它既可以是一个你创建的结构体内的函数，也可以是一个 [Lambda](#)，也可以是内置函数。

```
#include <functional>

int main()
{
    std::vector<int> values = { 3,5,1,4,2 };
    std::sort(values.begin(), values.end(), std::greater<int>()); // 从大到小排序

    for (int value : values)
        std::cout << value << std::endl;
    std::cin.get();
}
```

```
... Last: values.end(), _Pred: std::greater<int>());
...
5
4
3
2
1
```

`std::sort` 函数的比较函数（**Compare**）需要返回一个 `bool` 类型的值，用于指示两个元素之间的大小关系，第一个元素在前的话为true。

```
std::sort(values.begin(), values.end(), [](int a, int b)
{
    return a < b; // 小的排前面，从小到大
});
```

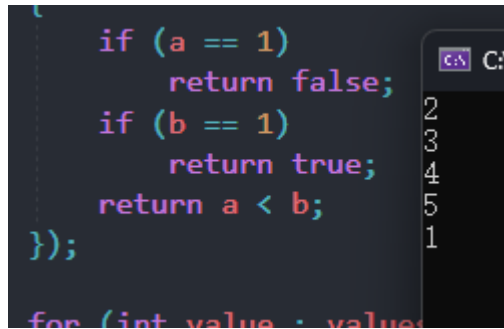
```
std::sort(_First: values.begin(), _Last: values.end(), _Pred: [](int a, int b) -> bool
{
    return a < b;
});

for (int value : values)
    std::cout << value << std::endl;
std::cin.get();
1
2
3
4
5
```

修改为 `return a > b` 则反过来，从大到小排序。

如果想将1排在最后，则可以：

```
std::sort(values.begin(), values.end(), [](int a, int b)
{
    if (a == 1)
        return false;
    if (b == 1)
        return true;
    return a < b;
});
```



```
    if (a == 1)
        return false;
    if (b == 1)
        return true;
    return a < b;
});
for (int value : values)
```

排序是非常有用的，你可以对所有类型进行排序，不一定必须是整数，你可以用string，可以用自定义的类。因此这里的 *Predicate*（谓词）也就是lambda，是非常有用的，因为这意味着我们可以设置规则，不依赖于只在内置类型或类似的东西上工作，