8 杂项

8.1 std::views 语法糖

std::views 是**惰性求值**的，只会在遍历时计算；并且**不复制**数据；**创建** std::views 的复杂度为常数

```cpp
vector<int> nums{1, 2, 3, 4, 5, 6}; // 假设有这个数组
```

过滤 views::filter

```cpp
auto even = nums | views::filter([](int n){ return n % 2 == 0; });
for (int n : even) std::cout << n << " "; // 输出: 2 4 6
```

转换 views::transform

```cpp
auto squares = nums | views::transform([](int n){ return n *n; });
for (int n : squares) std::cout << n << " "; // 输出: 1 4 9 16 25 36
```

取前 N 个 / 最后 N 个 views::take / views::take_last

```cpp
auto first3 = nums | views::take(3); // 输出: 1 2 3
auto last3 = nums | views::take_last(3); // 输出: 4 5 6
```

丢弃前 N 个 / 最后 N 个 views::drop / views::drop_last

```cpp
auto without_first2 = nums | views::drop(2); // 输出: 3 4 5 6
auto without_last2 = nums | views::drop_last(2); // 输出: 1 2 3 4
```

管道操作符 |

```cpp
// 链式操作：过滤偶数 -> 平方 -> 取前2个
auto result = nums
    | views::filter([](int n){ return n % 2 == 0; })
    | views::transform([](int n){ return n *n; })
    | views::take(2);
for (int n : result) std::cout << n << " "; // 输出: 4 16
```

反转 views::reverse

```cpp
auto reversed = nums | views::reverse; // 输出: 6 5 4 3 2 1
```

键 / 值视图 views::keys / views::values

```cpp
map<int, string> m{{1, "a"}, {2, "b"}};
auto keys = m | views::keys; // 1, 2
auto values = m | views::values; // "a", "b"
```

连接 views::join

```cpp
std::vector<std::vector<int>> vecs{{1,2}, {3,4}};
auto flattened = vecs | views::join; // 输出: 1 2 3 4
```

分割字符串 views::split

```cpp
#include <string>
std::string str = "hello,world,cpp";
auto parts = str | views::split(',');
// parts: "hello" "world" "cpp"
```

生成序列 views::iota

```cpp
// 生成无限序列
auto infinite = views::iota(1); // 1, 2, 3, 4...
auto first5 = views::iota(1) | views::take(5); // 1, 2, 3, 4, 5

// 带步长
auto evens = views::iota(0, 10, 2); // 0, 2, 4, 6, 8
```

转换为容器

```cpp
auto view = nums | views::take(3);
vector<int> vec(view.begin(), view.end());
```

范围 for 循环中使用

```cpp
for (int n : nums | views::filter([](int x){ return x > 3; })) {
  std::cout << n << " "; // 4 5 6
}
```