原创 AICodeThunder 于 2024-05-18 20:52:32 发布 阅读量962 收藏 25 点赞数 8 版权

分类专栏： C++知识点总结 文章标签： c++ java 开发语言

C++知识点总结 专栏收录该内容 7 订阅 53 篇文章 订阅专栏

map 映射

一、意义

1. 映射的概念

> **映射**（mapping）
> 键（`key`）与值（`value`）的对应关系。

> **`map` 容器**
> 一个 `STL` 中常用的 容器 ，可以将任何有序的类型（基本类型、 `string`）映射到任何类型（基本类型、`string`、结构体 、`STL` 容器）。

2. map 的特点

- 动态存储空间
- 插入、删除、查找效率高
- 键与值类型定义灵活
- 适用于判断范围较大的整数（例如在排序中）或者其他类型的数据（例如 string）是否存在，以及需要建立的字符（串）与其他数据类型

1. 定义

定义格式： `map <键类型, 值类型> 容器名;`
例如：

```
#include <map>
map <string, string> m;
```

2. 初始化

初始化格式： `map <键类型, 值类型> 容器名{{键1, 值1}, {键2, 值2}, ...};`

```
#include <map>
map <string, string> m{{"a", "A"}, {"b", "B"}}; // 一定注意！没有等号！！
```

3. 赋值 & 访问

```
m["a"] = "A";
m["a"] = "B"; // 值会被覆盖
cout << m["a"]; // 输出：B
cout << m["b"]; // 输出空字符串
```

4. 拷贝

```
#include <map>
map <string, string> m2(m);
```

5. 所有操作

方法	功能
`mp[key] = value;`	修改 `key` 对应的 `value`，如果不存在则创建
`mp.insert({key1, value1}, {key2, value2});`	插入纯新的键值对（不能更新） `{key1, value1}` 和 `{key2, value2}`
`mp.emplace(key, value)`	插入纯新的键值对（不能更新） `{key, value}`
`mp.erase(key);`	根据键删除键值对
`mp.clear();`	删除所有元素
`mp.begin()`	返回指向 `map` 头部的迭代器
`mp.end()`	返回指向 `map` 尾部的下一个位置的迭代器
`mp.size()`	返回 `map` 中键值对的个数
`mp.empty()`	判断是否为空
`mp.find(key)`	获得指向键 `key` 的迭代器，如果未找到返回 `mp.end()`
`mp.count(key)`	查看指定键 `key` 是否出现，出现为 1，否则为 0

三、迭代器

1. 定义

```
#include <map>
map <string, string> m{{"a", "A"}};
auto it = m.begin();
```

2. 遍历

```
for (auto it = m.begin(); it != m.end(); it++)
{
    cout << it->first <<
}
```


AICodeThunder 关注 8 25 0 专栏目录

方法	功能
it->first	返回当前迭代器指向的键
it->second	返回当前迭代器指向的值
mp.lower_bound(key)	返回第一个键值大于等于 key 的迭代器
mp.upper_bound(key)	返回第一个键值大于 key 的迭代器

四、例题

1. 登录注册

1.1 审题

做一个登录的功能，可以实现输入一个用户名，如果用户名在数据库里面已经用过了，那么将会跟上一个后缀。

1.2 参考答案

```cpp
#include <iostream>
#include <string>
#include <map>
using namespace std;

int n;
map <string, int> m; // 键: 名字 值: 出现的次数
string s;

int main()
{
    cin >> n;
    for (int i = 1; i <= n; i++)
    {
        cin >> s;
        if (m[s] == 0) // 未出现
        {
            cout << "OK\n";
            m[s] = 1;
        }
        else
        {
            cout << s << m[s] << endl;
            m[s]++;
        }
    }
    return 0;
}
```

2. XY 数对

2.1 审题

给定一个正整数数列 a[] 以及一个正整数 z，要求计算出所有满足 $x - y = z$ 的数对的个数。

2.2 参考答案

```cpp
#include <iostream>
#include <map>
using namespace std;

int n, z, x;
long long a[200005];
map <long long, long long> m; // 键: 数 值: 出现的次数

int main()
{
    cin >> n >> z;
```


AICodeThunder 关注 8 25 0 专栏目录