简单的STL调用

简单的STL调用

刷新

题目描述

棋盘的空间被抽象为二维点阵，现在有D枚棋子依次占领棋盘中的位置。若一个棋子占领一个格子，则将这个格子命名为自己的名字。数据保证一个格子不会被重复占领。

现有查询操作，查询某个位置的棋子的名字是什么。若查询位置无棋子占领，则输出 `Not found` 。

棋子的名字可能为string类型，也可能是int类型，程序需要支持这两种类型。

我们实现了一个简单的vector版本，使用了迭代器来进行运算，而这种运算是非常慢的。使用的容器是 `vector` ,查找点操作使用迭代器最坏情况下需要O(n)的时间，而在排序完成的情况下通过二分查找只需要O(logn)的时间。

因此，请你根据抽象类Container实现所需接口，修改 `vector` 的特化版本，并提供在 `map` 的特化版本，使得在使用这两种容器时，效率尽可能高。

文件下载地址：下载链接 (/staticdata/1998.Mqs70h0bdTHAMzNF.pub/A7afxqvhHUaP7Vtn.download.zip/download.zip)

提示：你可以使用 `<utility>` 库中的pair, make_pair函数方便打包坐标数据，方便二元组值的比较

提示：你可以使用 `<algorithm>` 库中的sort函数来完成对容器的排序，lower_bound完成二分查找，可以在cplusplus (https://cplusplus.com/)上找到有关接口的使用方法

提示：请观察数据范围、特点和时间限制来设计相应的特化，只要能通过OJ的自动评测即可

输入样例

第一行是两个字符串，分别为使用的容器，以及棋子名字的类型。

第二行是一个整数 `n` ，表示操作个数。

接下来 `n` 行，每行描述一个操作。首先是一个字符串，如果是 `insert` ，代表一个插入操作，之后是插入的位置和一个指定类型的元素。如果是 `find` ，代表一个查询操作，之后是两个整数 `x,y` ，表示查询 `(x,y)` 坐标上的棋子名字。

```
vector string
8
insert 4 3 Bob
insert 3 4 Alice
insert 6 2 Peter
insert 3 1 Liu
find 3 2
find 4 3
find 3 1
find 6 1
```

输出样例

对每个查询操作，输出一行为查询的答案。

```
Not found
Bob
Liu
Not found
```

要求

我们提供了 `BasicContainer.h` ， `Container.h` ， `main.cpp` ， `Makefile` 四个文件。你应该在 `Container.h` 中完成对vector和map的特化。不允许修改其他文件。

限制与约定

- 保证点坐标为整点，在int类型范围内，棋子名字可能重名。
- 对于使用 `vector` 的数据，n不超过 10^5 ，且所有查询操作出现在插入操作之后。
- **提示：你可以在所有插入操作结束之后，进行一次排序。**

- 对于使用 `map` 的数据，n不超过 10^5 ，不保证所有查询操作出现在插入操作之后。
- 棋子名字类型可能为 `int` 或是 `string` 。所有插入的字符串长度不超过 10。
- **提示：你可以在cplusplus (https://cplusplus.com/)网站上查询STL函数接口和容器各个操作的时间复杂度。**
- **时间限制：1s**
- **空间限制：256MB**

提交格式

- 你应该提交 `Container.h` ，我们会将 `BasicContainer.h` 、 `main.cpp` 、 `Makefile` 拷贝到目录下，与你提交的代码一同编译运行

语言和编译选项

#	名称	编译器	额外参数	代码长度限制
0	custom	`make`		65536 B

递交历史

#	状态	时间

递交答案

语言和编译选项 custom

```
1
```

提交

文件请拖入编辑器中，或 上传文件