# 题目

栈排序。 编写程序，对栈进行排序使最小元素位于栈顶。最多只能使用一个其他的临时栈存放数据，但不得将元素复制到别的数据结构（如数组）中。该栈支持如下操作：push、pop、peek 和 isEmpty。当栈为空时，peek 返回 -1。

示例1:

输入：

["SortedStack", "push", "push", "peek", "pop", "peek"]

[[], [1], [2], [], [], []]

输出：

[null,null,null,1,null,2]

示例2:

输入：

["SortedStack", "pop", "pop", "push", "pop", "isEmpty"]

[[], [], [], [1], [], []]

输出：

[null,null,null,null,null,true]

说明:

栈中的元素数目在[0, 5000]范围内。

# 分析

## 方法一：栈

思路：

可以使用一个辅助栈来实现栈的排序。

代码：

class SortedStack {

private:

stack<int> data;

stack<int> temp;

public:

SortedStack() {

}

void push(int val) {

while (!data.empty() && val > data.top()) {

temp.push(data.top());

data.pop();

}

data.push(val);

while (!temp.empty()) {

data.push(temp.top());

temp.pop();

}

}

void pop() {

if (!data.empty()) {

data.pop();

}

}

int peek() {

return data.empty() ? -1 : data.top();

}

bool isEmpty() {

return data.empty();

}

};

这个实现中，我们使用一个额外的辅助栈temp来辅助排序。在push操作中，我们将要入栈的元素val与栈顶元素比较，如果val大于栈顶元素，则将栈顶元素弹出并放入temp栈中，直到val小于或等于栈顶元素，然后将val入栈。最后，将temp中的元素依次放回data栈中，实现排序。在pop操作中，直接弹出data栈顶元素。在peek操作中，返回data栈顶元素。在isEmpty操作中，判断data是否为空。