## 题目描述：

## 定义栈的数据结构，请在该类型中实现一个能够得到栈中所含最小元素的min函数（时间复杂度应为O（1））。

## 解题思路：

 看到这个问题, 我们最开始可能会想, 添加一个成员变量用于保存最小元素, 每次压栈时如果压栈元素比当前最小元素更小, 就更新最小元素.

    但是这样会有一个问题, 如果最小元素被弹出了呢, 如何获得下一个最小元素呢? 分析到这里可以发现, 仅仅添加一个成员变量存放最小元素是不够的, 我们需要在最小元素弹出后还能得到次小元素, 次小的弹出后, 还要能得到次次小的.

    因此, 用另一个栈来保存这些元素是再合适不过的了. 我们叫它最小元素栈.

    每次压栈操作时, 如果压栈元素比当前最小元素更小, 就把这个元素压入最小元素栈, 原本的最小元素就成了次小元素. 同理, 弹栈时, 如果弹出的元素和最小元素栈的栈顶元素相等, 就把最小元素的栈顶弹出.

## 算法描述：

class Solution {

public:

stack<int> stack1, stack2;//只能定义在外面

void push(int value) {

stack1.push(value);

if (stack2.empty())

stack2.push(value);

if (value <= stack2.top())

stack2.push(value);

}

void pop() {

if (stack1.top() == stack2.top())

stack2.pop();

stack1.pop();

}

int top() {

return stack1.top();

}

int min() {

return stack2.top();

}

};