最小栈

设计一个支持 push ， pop ， top 操作，并能在常数时间内检索到最小元素的栈。

- push(x) —— 将元素 x 推入栈中。
- pop() —— 删除栈顶的元素。
- top() —— 获取栈顶元素。
- getMin() —— 检索栈中的最小元素。

示例:

输入：

```
["MinStack","push","push","push","getMin","pop","top","getMin"]

[[],[-2],[0],[-3],[],[],[],[]]
```

输出：

```
[null,null,null,null,-3,null,0,-2]
```

解释：

```
MinStack minStack = new MinStack();

minStack.push(-2);

minStack.push(0);

minStack.push(-3);

minStack.getMin();   --> 返回 -3.

minStack.pop();

minStack.top();      --> 返回 0.

minStack.getMin();   --> 返回 -2.
```

```
class MinStack {
public:
    stack<int> A;
    stack<int> B;
    /** initialize your data structure here. */
```

```
    MinStack() {

    }

    void push(int val) {
        A.push(val);
        if(B.empty()||val<=B.top())//小栈为空或小于等于栈顶元素
        {
            B.push(val);
        }
    }

    void pop() {
        if(A.top()==B.top())
        {
            B.pop();//相等情况 更新小栈栈顶
        }
        A.pop();
    }

    int top() {
        return A.top();
    }

    int getMin() {
        return B.top();
    }
};

/**
 * Your MinStack object will be instantiated and called as such:
 * MinStack* obj = new MinStack();
 * obj->push(val);
 * obj->pop();
 * int param_3 = obj->top();
 * int param_4 = obj->getMin();
 */
```