```java
// Class for a special stack to get minimum element in O(1) Time, O(n) Space
// The idea is to do push() and pop() operations in such a way that the top of
// the auxiliary stack is always the minimum
class SpecialStack {
    Stack<Integer> stack = new Stack<>();
    Stack<Integer> aux = new Stack<>();
    /* returns min element from stack */
    int getMin() {
        if (stack.size() == 0) {
            return -1;
        }
        return aux.peek();
    }
    /* returns poped element from stack */
    int pop() {
        if (stack.size() == 0) {
            return -1;
        }
        aux.pop();
        return stack.pop();
    }
    /* push element x into the stack */
    void push(int x) {
        stack.push(x);
        if (aux.size() == 0) {
            aux.push(x);
        } else {
            if (aux.peek() >= x) {
                aux.push(x);
            } else {
                aux.push(aux.peek());
            }
        }
    }
}
```