<https://leetcode.com/problems/generate-parentheses/>

**Generate Parentheses**

**Given n pairs of parentheses, write a function to generate all combinations of well-formed parentheses.**

Example 1:

Input: n = 3

Output: ["((()))","(()())","(())()","()(())","()()()"]

Example 2:

Input: n = 1

Output: ["()"]

Constraints:

1 <= n <= 8

**Method 1: (Brute Force)**

Use two vectors or two stacks one for storing elements, other for storing minimum obtained at each position.

Time Complexity: O(1) *[for each function]*

Space Complexity: O(n) *[]*

class MinStack {

public:

    vector<int> s, minS;

    MinStack() {

    }

    void push(int val) {

        s.push\_back(val);

        if(s.size()==1 || val < minS.back())

            minS.push\_back(val);

        else minS.push\_back(minS.back());

    }

    void pop() {

        if(!s.empty()){

            s.pop\_back();

            minS.pop\_back();

        }

    }

    int top() {

        return s.back();

    }

    int getMin() {

        return minS.back();

    }

};