You're trying to implement the generateParenthesis function, which returns **all combinations of well-formed parentheses** for a given number n.

This is a classic **backtracking** problem.

Here is the **correct implementation**:

import java.util.\*;

class Solution {

public List<String> generateParenthesis(int n) {

List<String> result = new ArrayList<>();

backtrack(result, "", 0, 0, n);

return result;

}

// Helper method for backtracking

private void backtrack(List<String> result, String current, int open, int close, int max) {

// If the current string is a valid combination

if (current.length() == max \* 2) {

result.add(current);

return;

}

// If we can still add an opening parenthesis

if (open < max) {

backtrack(result, current + "(", open + 1, close, max);

}

// If we can add a closing parenthesis

if (close < open) {

backtrack(result, current + ")", open, close + 1, max);

}

}

}