Program for Combination of Number

```
import java.util.ArrayList;
import java.util.List;

public class CombinationNumber {

    // Function to generate all combinations of k elements from a set of n elements
    public static List<List<Integer>> combination(int n, int k) {
        List<List<Integer>> result = new ArrayList<>();
        List<Integer> current = new ArrayList<>();
        combineHelper(result, current, 1, n, k);
        return result;
    }

    // Recursive helper function to generate combinations
    private static void combineHelper(List<List<Integer>> result, List<Integer>
current, int start, int n, int k) {
        if (k == 0) {
            result.add(new ArrayList<>(current));
            return;
        }
        // Generate combinations by selecting elements from 'start' to 'n'
        for (int i = start; i <= n; i++) {
            current.add(i);
            combineHelper(result, current, i + 1, n, k - 1);
            current.remove(current.size() - 1);
        }

    }

    public static void main(String[] args) {
        int n = 4;
        int k = 2;
        List<List<Integer>> result = combination(n, k);
        System.out.println("Combinations are :");
        for (List<Integer> combination : result) {
            System.out.print(combination);
        }
    }

}
```

OUTPUT

Combinations are :

[1, 2][1, 3][1, 4][2, 3][2, 4][3, 4]

Program for Generate Parenthesis

```
import java.util.ArrayList;
import java.util.List;

public class GenerateParentheses {

    public static List<String> generateParenthesis(int n) {
        List<String> result = new ArrayList<>();
        generateParenthesisHelper(result, "", 0, 0, n);
        return result;
    }

    private static void generateParenthesisHelper(List<String> result, String
current, int open, int close, int n) {
        if (current.length() == 2 * n) {
            result.add(current);
            return;
        }

        if (open < n) {
            generateParenthesisHelper(result, current + "(", open + 1, close, n);
        }

        if (close < open) {
            generateParenthesisHelper(result, current + ")", open, close + 1, n);
        }
    }

    public static void main(String[] args) {
        int n = 3;
        List<String> parenthesesCombinations = generateParenthesis(n);

        System.out.println("Combinations of parentheses for n = " + n + ":");
        for (String combination : parenthesesCombinations) {
            System.out.print("["+combination+"] ");

         }
     }
}
```

OUTPUT

Combinations of parentheses for n = 3:

[((()))] [(()())] [(())()] [()(())] [()()()]