**Easy: Given the list of array return array in which each element is the product of other element except ith element (try to do it without division operation) input: [1,2,3,4] output:[24,12,8,6]**

**CODE :**

public class ProductExceptSelf {

public static void main(String[] args) {

int[] nums = {1, 2, 3, 4};

int n = nums.length;

int[] output = new int[n];

for (int i = 0; i < n; i++) {

output[i] = 1;

}

int leftProduct = 1;

int rightProduct = 1;

for (int i = 0; i < n; i++) {

output[i] \*= leftProduct;

leftProduct \*= nums[i];

}

for (int i = n - 1; i >= 0; i--) {

output[i] \*= rightProduct;

rightProduct \*= nums[i];

}

for (int i = 0; i < n; i++) {

System.out.print(output[i] + " ");

}

}

}

**Medium: Given an array list return all possible permutations Input: nums = [1,4,3]**

**Output: [[1,4,3],[1,3,4],[4,1,3],[4,3,1],[3,1,4],[3,4,1]]**

**CODE :**

import java.util.ArrayList;

import java.util.List;

public class PermutationsWithoutFunctions {

public static void main(String[] args) {

int[] nums = {1, 4, 3};

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

boolean[] used = new boolean[nums.length];

List<Integer> current = new ArrayList<>();

if(nums.length > 0) {

generatePermutations(nums, used, current, result);

for (List<Integer> perm : result) {

System.out.println(perm);

}

} else {

System.out.println("Empty array");

}

}

private static void generatePermutations(int[] nums, boolean[] used, List<Integer> current, List<List<Integer>> result) {

if (current.size() == nums.length) {

result.add(new ArrayList<>(current));

return;

}

for (int i = 0; i < nums.length; i++) {

if (!used[i]) {

used[i] = true;

current.add(nums[i]);

generatePermutations(nums, used, current, result);

current.remove(current.size() - 1);

used[i] = false;

}

}

}

}

**Hard:**

**Return all the clubbed words**

**Input: words =["mat","mate","matbellmates","bell","bellmatesbell","butterribbon","butter","ribbon"] Output: ["matbellmates","bellmatesbell","butterribbon"]**

**CODE :**

import java.util.ArrayList;

import java.util.Arrays;

import java.util.List;

public class ClubbedWords {

public static void main(String[] args) {

String[] words = {"mat", "mate", "matbellmates", "bell", "bellmatesbell", "butterribbon", "butter", "ribbon"};

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

for (String word : words) {

if (isClubbedWord(word, Arrays.asList(words))) {

clubbedWords.add(word);

}

}

System.out.println("Clubbed Words: " + clubbedWords);

}

private static boolean isClubbedWord(String word, List<String> words) {

if (word.length() == 0) {

return false;

}

if (words.contains(word)) {

return true;

}

for (int i = 1; i < word.length(); i++) {

String prefix = word.substring(0, i);

String suffix = word.substring(i);

if (words.contains(prefix) && isClubbedWord(suffix, words)) {

return true;

}

}

return false;

}

}