1. The string "PAYPALISHIRING" is written in a zigzag pattern on a given number of rows like this: (you may want to display this pattern in a fixed font for better legibility)
Example 1:
Input: s = "PAYPALISHIRING", numRows = 3
```
P   A   H   N
A P L S I I G
Y   I   R
```
And then read line by line: "PAHNAPLSIIGYIR"

Write the code that will take a string and make this conversion given a number of rows:

```
string convert(string s, int numRows);
```

Example 2:

Input: s = "PAYPALISHIRING", numRows = 4
Output: "PINALSIGYAHRPI"
Explanation:
```
P     I    N
A   L S  I G
Y A   H R
P     I
```

Constraints:

1 <= s.length <= 100
s consists of English letters (lower-case and upper-case), ',' and '.'.
1 <= numRows <= 100

===
2. 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

===
3. Given a string s and an array of strings words of the same length. Return all starting indices of substring(s) in s that is a concatenation of each word in words exactly once, in any order, and without any intervening characters.

You can return the answer in any order.

Example 1:

Input: s = "barfoothefoobarman", words = ["foo","bar"]
Output: [0,9]
Explanation: Substrings starting at index 0 and 9 are "barfoo" and "foobar" respectively.
The output order does not matter, returning [9,0] is fine too.
Example 2:

Input: s = "wordgoodgoodgoodbestword", words = ["word","good","best","word"]

```
Output: []
Example 3:

Input: s = "barfoofoobarthefoobarman", words = ["bar","foo","the"]
Output: [6,9,12]

Constraints:

1 <= s.length <= 104
s consists of lower-case English letters.
1 <= words.length <= 5000
1 <= words[i].length <= 30
words[i] consists of lower-case English letters.
================================================================
4. Given a sorted array of distinct integers and a target value, return the
index if the target is found. If not, return the index where it would be if it
were inserted in order.

Example 1:

Input: nums = [1,3,5,6], target = 5
Output: 2
Example 2:

Input: nums = [1,3,5,6], target = 2
Output: 1
Example 3:

Input: nums = [1,3,5,6], target = 7
Output: 4

Constraints:

1 <= nums.length <= 100
0 <= nums[i] <= 100
nums contains distinct values sorted in ascending order.
0 <= target <= 100
================================================================
5. Given a positive integer n, return the nth term of the count-and-say
sequence.
For example, the saying and conversion for digit string "3322251":

3322251 - given
two 3's, three 2's, one 5 and one 1
23+32+15+11
23321511

1211
one 1, one 2,two 1's
11+12+21
111221
================================================================
```