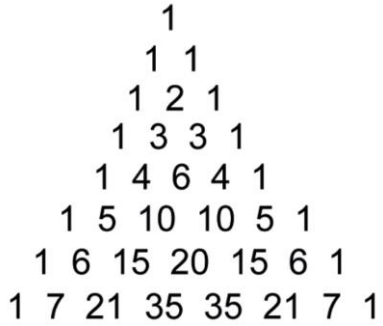
Code Logic:

1. Easy- Pascal's Triangle



-The corner elements of the triangle are 1. Hence, initialized to 1.
-Let the name of the triangle be 'A', and let 'row' indicate the row number.
At row = 3, we notice that (considering indexing from 1):
A[3][2] = A[2][1] + A[2][2].
Hence, this is the logic which is followed to write the code.

2. Medium- Given an integer array of size `n`, find all elements that appear more than `⌊ n/3 ⌋` times.

   This Python script (findElements) is designed to identify elements in an array that occur more than one-third of the total array length. It utilizes a frequency counting approach with hashing to determine the occurrences of each element in the array and then checks if the frequency exceeds the one-third limit.

   -We maintain a hash table to count the occurrence of each unique element.
   -The count is compared with the one-third of total array length in order to find the elements whose occurrences exceed the calculated limit(n/3).
   -The final result, containing elements occurring more than one-third of the array length, is returned.

3. Hard: Shortest Palindrome

   Logic: Suppose, string = "abab";
   reversed_string = "baba";
   new_string = + reversed_string + string= "b**aba**abab"
   we notice that the underlined test is what we need to get rid of.

   Approach:

   new_string = stirng + reversed_string = "ababb**aba**"
   We notice that the underlined text is what we need to get rid of. How do we find out how many elements do we need to get rid of?

To do so we use KMP(Knuth Morris Pratt) algorithm here. We maintain a LPS(Longest Prefix Suffix) table.

For the above example, LPS table would look like:
char: a b a b b a b a
prefix: 0 0 1 2 0 1 2 **3**
Hence, 3 is the number of elements that are common. 3 is the length of the longest proper prefix which is also a proper suffix.

- We Construct the shortest palindrome by taking the substring from the length obtained to the end of the original text, reversing it, and concatenating it with the original text.