TIP BITS GOA

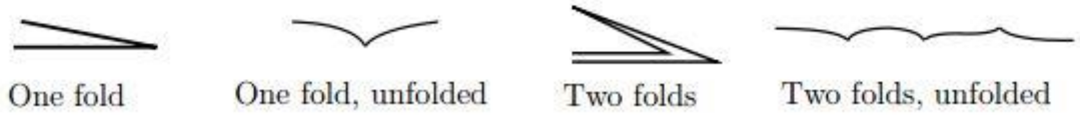
Competitive Programming Quiz.

Instructions:

- There are **3** questions: Each of them carry 20 marks. Each question has 3 parts. **If u solve all parts correctly, you get 20 marks for that question. Otherwise, you get 5 marks for each part that you solve correctly.**
- **Only final answer will fetch you marks, otherwise zero. You should get used to this type of marking in BITS :-P .**

Questions:

1. Here in *BITS-Pilani, Goa* we tend fold our answer scripts in a haphazard manner (perhaps more often, especially when we get single digit scores in “NOT TO BE MENTIONED”:-P subjects).
   And please do not fold this question paper/answer sheet after reading this question ;-).
   Suppose we take a sheet of paper and fold it in half lengthwise, putting the right side over the left. When we unfold the paper, there will be a center crease like a valley, which we write as ‘V’.
   Suppose we fold it lengthwise twice—that is, we fold it lengthwise in half, right over left, and then fold the folded sheet lengthwise in half again, right over left. When we unfold this doubly folded sheet, we see three equally spaced creases, some like hills ‘Λ’, some like valleys ‘V’. The actual pattern we see is – ‘ V V Λ ‘.

   One fold One fold, unfolded Two folds Two folds, unfolded

   (a) How many valleys are there if we fold the paper 10 times and unfold?
   (b) Write down the pattern of hills and valleys of the first ten creases if we fold the paper 15 times and unfold.
   (c) Write down the pattern of hills and valleys of the last ten creases if we fold the paper 18 times and unfold.

----

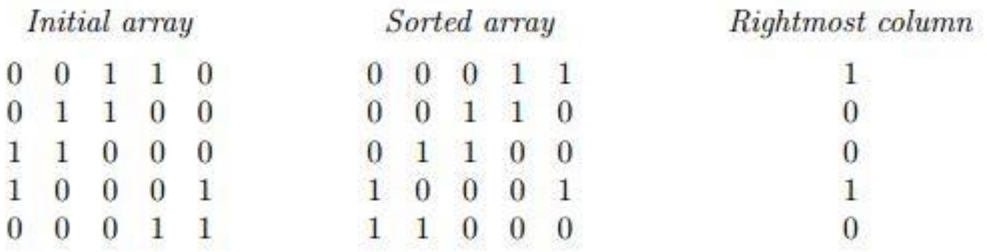
2. Consider a sequence b₁, b₂, b₃,….b$_N$ of N binary digits. Given such a sequence, we rotate it by one digit to the left N−1 times to generate a block of N sequences arranged in an N × N array of 0’s and 1’s, as follows.

$$\begin{array}{cccc} b_1 & b_2 & \dots & b_{N-1} & b_N \\ b_2 & b_3 & \dots & b_N & b_1 \\ \vdots & \vdots & & \vdots & \vdots \\ b_{N-1} & b_N & \dots & b_{N-3} & b_{N-2} \\ b_N & b_1 & \dots & b_{N-2} & b_{N-1} \end{array}$$

We then sort these sequences in lexicographic order—that is, regard each row of the array as a binary number and rearrange the rows in ascending order.

After this rearrangement, we extract the last column of the new N × N array. The goal is to work backwards from this column and compute the top row of the N × N sorted array that created it.

For example, consider the binary sequence 0 0 1 1 0. After sorting the rows of the 5 × 5 array that this sequence generates, the last column reads 1 0 0 1 0 from top to



Initial array	*Sorted array*	*Rightmost column*
0 0 1 1 0	0 0 0 1 1	1
0 1 1 0 0	0 0 1 1 0	0
1 1 0 0 0	0 1 1 0 0	0
1 0 0 0 1	1 0 0 0 1	1
0 0 0 1 1	1 1 0 0 0	0

Each of the inputs below describes the rightmost column of a sorted array, from top bottom. Your task is to compute the first row of the corresponding sorted array for each of these inputs. In all three cases, the answer is unique.

(a) 1 0 1 1 1 1 0 0

(b) 1 0 1 1 0 1 1 1 0 0 1 0

(c) 1 1 1 1 0 1 1 0 1 0 1 1 0 1 0

---

3. Given a number N between 2 and 9, consider the integers that can be formed using each digit in the set {1, 2, . . . , N} exactly once. For instance, if N is 3, the integers we can form are 123, 231, 312, 132, 321 and 213. If we arrange these in increasing order, we get the list {123, 132, 213, 231, 312, 321}. The fourth number in this list is 231. In general, given two numbers N and K, the task is to compute the number at position K when all integers formed using the digits {1, 2, . . . N} exactly once are arranged in ascending order. For instance, the example worked out above corresponds to N = 3 and K = 4.
Compute the answer for the following values of N and K.

(a) N = 5, K = 76 (b) N = 7, K = 4197 (c) N = 9, K = 191082