www.artofproblemsolving.com/community/c1795620
by kapilpavase

Q1 Given is a $n \times n$ grid with all squares on one diagonal being forbidden. You are allowed to start from any square, and move one step horizontally, vertically or diagonally. You are not allowed to visit a forbidden square or previously visited square. Your goal is to visit all non forbidden squares. Find, with proof, the minimum number of times you will have to move one step diagonally

---

Q2 Given is an array A of $2n$ numbers, where n is a positive integer. Give an algorithm to create an array $prod$ of length $2n$ where

$$prod[i] = A[i] \times A[i+1] \times \cdots \times A[i+n-1],$$

($A[x]$ means $A[x \bmod 2n]$) in $O(n)$ time **without** using division. Assume that all binary arithmetic operations are $O(1)$

Q3 A positive sequence is a finite sequence of positive integers. Sum of a sequence is the sum of all the elements in the sequence. We say that a sequence A can be embedded into another sequence B, if there exists a strictly increasing function

$$\phi : \{1, 2, \ldots, |A|\} \to \{1, 2, \ldots, |B|\},$$

such that $\forall i \in \{1, 2, \ldots, |A|\}$,

$$A[i] \le B[\phi(i)],$$

where $|S|$ denotes the length of a sequence S. For example, $(1, 1, 2)$ can be embedded in $(1, 2, 3)$, but $(3, 2, 1)$ can not be in $(1, 2, 3)$

Given a positive integer n, construct a positive sequence U with sum $O(n \log n)$, such that all the positive sequences with sum n, can be embedded into U.

Q4 Let $a_1, a_2, \ldots a_n$ be positive real numbers. Define $b_1, b_2, \ldots b_n$ as follows.

$$b_1 = a_1$$
$$b_2 = max(a_1, a_2)$$
$$b_i = max(b_{i-1}, b_{i-2} + a_i) \text{ for } i = 3, 4 \ldots n$$

Also define $c_1, c_2 \ldots c_n$ as follows.

$$\begin{aligned} c_n &= a_n \\ c_{n-1} &= max(a_n, a_{n-1}) \\ c_i &= max(c_{i+1}, c_{i+2} + a_i) \text{ for } i = n-2, n-3 \ldots 1 \end{aligned}$$

Prove that $b_n = c_1$.

Q5 Given a string of length $2n$, we perform the following operation:

-Place all the even indexed positions together, and then all the odd indexed positions next. Indexing is done starting from 0.

For example, say our string is “abcdef”. Performing our operation yields “abcdef” $\rightarrow$ “acebdf”. Performing the operation again yields “acebdf” $\rightarrow$ “aedcbf”. Doing this repeatedly, we have:

“abcdef” $\rightarrow$ “acebdf” $\rightarrow$ “aedcbf” $\rightarrow$ “adbecf” $\rightarrow$ “abcdef”.

You can assume that the characters in the string will be unique. It can be shown that, by performing the above operation a finite number of times we can get back our original string.

Given n, you have to determine the minimum number of times the operation must be performed to get our original string of length $2n$ back.

In the example given above, $2n = 6$. The minimum steps required is 4.

Q6 Some bugs are sitting on squares of 10×10 board. Each bug has a direction associated with it **(up, down, left, right)**. After 1 second, the bugs jump one square in **their associated** direction. When the bug reaches the edge of the board, the associated direction reverses (up becomes down, left becomes right, down becomes up, and right becomes left) and the bug moves in that direction. It is observed that it is **never** the case that two bugs are on same square. What is the maximum number of bugs possible on the board?
