CS 161 (Stanford, Winter 2026) Section 7

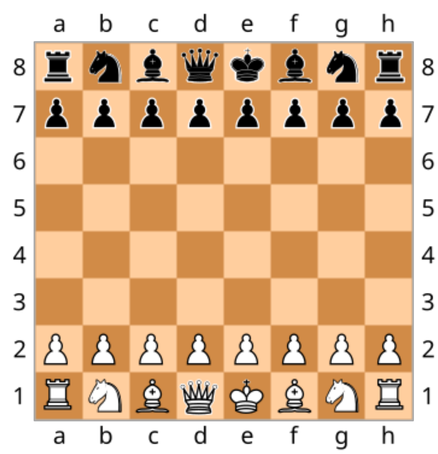
1 Encoding

Suppose we encode lowercase letters into a numeric string as follows: we encode *a* as 1, *b* as 2, . . . , and *z* as 26. Given a numeric string S of length n, develop an $O(n)$ algorithm to find how many letter strings this can correspond to. For example, for the numeric string 123, the algorithm should output 3 because the letter strings that map to this numeric string are *abc* (decoded as "1", "2", "3") *lc*, (decoded as "12", "3") and *aw* (decoded as "1", "23").

2 Knight Moves

Given an 8×8 chessboard and a knight that starts at position $a1$, devise an algorithm that returns how many ways the knight can end up at position xy after k moves. Knights move ± 1 squares in one direction and ± 2 squares in the other direction. In other words, knights move in a pattern similar to a L.

Note: on a chessboard, rows are labeled from 1 to 8 and columns are labeled from *a* to *h*, as seen below.



3 String Cutting

Suppose we have a string of length k, where k is a positive integer. We would like to cut the string into integer-length segments such that we maximize the *product* of the resulting segments' lengths. Multiple cuts may be made. For example, if $k = 8$, the maximum product is 18 from cutting the string into three pieces of length $3, 3$, and 2. Write an algorithm to determine the maximum product for a string of length k.