Problem A. Alice and Bob (and string)

Можно заметить, что суффиксный автомат будет в точности графом описанной игры. Обходом в глубину можно определить выигрышности всех состояний. Говорим, что состояние выигрышное, если из него есть хотя бы один переход в проигрышное состояние. Далее можно динамикой по ориентированному ациклическому графу посчитать, сколько существует путей, которые ведут в какое-то выигрышное состояние. Далее, используя эту информацию, найти k-ый лексикографически выигрышный путь за O(длина пути). Символы, записанные на нём - ответ на задачу.

Problem B. Alice and Bob (and string) 2

Разворачиваем строку S, строим по ней автомат и считаем выигрышности как в прошлой задаче. Далее пользуемся тем фактом, что дерево суффиксных ссылок образует дерево развернутой (раз мы уже развернули строку один раз, то в данном случае исходной) строки. При этом выигрышными или проигрышными одновременно являются все строки, которые лежат на одном и том же ребре (см. часть о соответствии между состояниями автомата и рёбрами дерева с лекции). Пользуясь этой информацией можно найти k-ую выигрышную строку обходом суффиксного дерева.

Problem C. Substring

Пусть суммарная длина списка слов L. Это означает, что количество различных длин слов не больше O(sqrt(L)). Построим автомат Ахо-Корасик для данного списка слов. Дополнительно к стандартным суффиксным ссылкам, из каждого состояния проведём ссылку (будем её называть терминальной ссылкой) в ближайшее терминальное состояние в его суффиксном пути. Далее в каждом состоянии создадим список префиксов текста, где строка, соответствующая состоянию, встречается как суффикс. Изначально все такие списки создадим пустыми.

Будем пошагово “кормить” ему данный текст. После добавления каждого символа мы получили некоторое состояние в автомате спустимся по пути из терминальных ссылок и в список префиксов добавим текущий. Таким образом мы для каждого слова найдём список его вхождений в текст. Далее пробежимся по запросам и для каждого запроса обратимся в соответствующий ей список и проверим на наличие подходящих позиций бинарным поиском.

Problem D. Refrain

Для каждого состояния суффиксного автомата строки требуется посчитать мощность правого контекста (число вхождений строк из данного сосояния в исходную строку). После чего взять максимум по всем состояниям произведения максимальной длины строки в нём и мощности правого контекста. Это будет ответом на задачу.

Problem E. Average Common Prefix

На разборе прошлого контеста вам рассказывали, как с помощью хешей за log сравнить две строки. С помощью такого сравнения мы можем отсортировать массив всех циклических сдвигов за n * log n * log n. После этого для каждой пары соседних строк в отсортированном массиве нужно за log n найти наибольший общий префикс (снова хешами).

Problem F. Bacon's Cypher

Эта задача является учебной и дана для закрепления. Она является позадачей в одной из задач прошлого контеста.

Problem G. Mnemonics and Palindromes 3

Мы не будем разбирать эту задачу, т.к. она очень простая. Таких строк не больше 6, вы можете получить их перебором на листочке или программно.