Технокубок

Технокубок 2017 - Отборочный Раунд 1

Разбор задач первого отборочного раунда Технокубка 2016/2017

727A - Превращение: из A в B

Будем решать задачу в обратную сторону — попытаемся получить из числа B число A.

Заметим, что если число B заканчивается на 1, то последней операцией, которую использовал Василий — приписать к числу справа 1. Поэтому удалим последнюю цифру из B и перейдем в новому числу.

Если же последняя цифра чётная — то последней операцией, которую использовал Василий — умножить число на 2. Поэтому поделим B пополам и перейдем к новому числу.

Если же B заканчивается на нечетную цифру отличную от 1, то ответ «NO».

После того как мы перешли к новому числу, нужно вновь выполнить описанный алгоритм. Если на каком-то шаге мы получили число равное A, то мы нашли ответ, а если же мы получили число, меньшее A, то ответ «NO».

727B - Сумма чека

В данной задаче нужно было аккуратно сделать то, что написано в условии. Можно было сначала выделить все последовательности из подряд идущих цифр и точек, которые являлись ценами.

Затем нужно было выделить целое количество рублей из каждой цены и отдельно считать сумму всех целых цен в переменную r. То же нужно было делать для копеек из каждой цены и складывать их в переменную c.

После обработки всех цен, нужно было перевести копейки в рубли, то есть прибавить к r величину $c / 100$ (целая часть от деления c на 100), а c

присвоить значению $c\%100$ (остаток от деления c на 100). После этого осталось только аккуратно вывести ответ, не забыв, что если $c < 10$, то сначала для копеек нужно вывести 0, а затем c, так как количество копеек по условию обязательно должно состоять из двух цифр.

727C - Восстановление массива

Изначально сделаем три запроса на сумму чисел $a1 + a2 = c1$, $a1 + a3 = c2$ и $a2 + a3 = c3$.

После этого мы получаем систему из трех уравнений с тремя неизвестными $a1, a2, a3$. После простых вычислений получим, что $a3 = (c3 - c1 + c2) / 2$. После этого легко находятся $a1$ и $a2$. Теперь мы знаем значения $a1, a2, a3$, потратив на это 3 запроса.

Затем для всех i от 4 до n нужно сделать запрос на сумму $a1 + ai$. Если очередная сумма равна ci, то $ai = ci - a1$ (напомним, что мы уже знаем значение $a1$).

Таким образом можно восстановить весь массив, потратив на это ровно n запросов.

727D - Распределение футболок

Пусть в массиве cnt хранится сколько в типографии есть футболок каждого из размеров.

Изначально раздадим футболки тем, кто точно хочет футболку одного размера, уменьшая при этом соответствующее значение в массиве cnt. Если в какой-то момент футболки не хватило, то ответа не существует.

Теперь осталось раздать футболки тем, кто хочет футболку одного из двух размеров. Поступим жадным образом. Раздадим по максимуму футболки размера `S` тем, кто хочет их или футболки размера `M`. После этого перейдем к футболкам размера `M` и сначала раздадим их по максимуму тем, кто хочет футболки размера `S` или `M`, но кому не хватило футболок `S`, а затем, если остались футболки размера `M`, раздадим их

тем, кто хочет их или футболки размера `L`. Аналогичным образом раздадим футболки оставшихся размеров.

Если после всех операций с футболками у кого-то футболки не оказалось, значит ответа не существует, в противном случае, ответ найден.

727E - Игры на диске

С помощью алгоритма Ахо-Корасика построим суффиксное дерево на множестве названий игр так, что в вершине дерева, соответствующей названию некоторой игры, (вершине на глубине k) будем хранить номер этой игры.

Построенный бор позволяет приписывать к некоторой строке символы один за другим и определять вершину в нем, соответствующую наидлиннейшему префиксу из всех префиксов названий игр, совпадающему с суффиксом нашей строки. Если длина этого префикса равна k, то суффикс совпадает с некоторым названием игры.

Запишем строку из входных данных дважды и посчитаем $idxi$ — индекс игры, название которой совпадает с подстрокой удвоенной строки с индекса $i - k + 1$ по индекс i включительно (если такой нет, то -1).

Теперь остается перебрать индекс символа, который является последним в записи названия некоторой игры на диск. Очевидно, этот индекс можно перебирать от 0 до $k - 1$. С фиксированным индексом f достаточно проверить, что все названия с последними символами в индексах $f + ik \mod nk$ для $0 \le i < n$ являются различными (для этого смотрим, что среди $idx(f + ik)\%nk + nk$ нет -1 и все они различны). Если это выполняется — выводим `YES` и легко восстанавливаем ответ. Если ни для одного f условия не выполнились, выводим `NO`.

Асимптотика решения — $O(nk + \sum |t_i|)$

727F - Задачи Поликарпа

Для начала решим задачу для одного значения Q. Нетрудно показать, что оптимальным поведением является следующее: добавляем в множество оставленных задач очередную задачу качества ai; пока значение настроения (сумма качеств и Q) является отрицательным, удаляем из множества оставленных задач задачу с наихудшим качеством. Качество такой задачи обязательно будет отрицательным, поэтому мы не испортим значения настроения на предыдущих задач. Такое моделирование просто осуществляется с помощью структур `std::set` или `std::priority_queue`.

Соображение выше позволяет нам отвечать на запрос за $O(n \log n)$, однако $O(mn \log n)$ не укладывается в ограничение по времени. Поэтому нужно заметить, что при увеличении Q количество удаленных задач не увеличивается, а возможных таких количеств всего n. Таким образом, на задачу нужно взглянуть с обратной стороны: для $0 \le x \le n$ посчитать, какое наименьшее значение может иметь Q так, что количество удаленных задач не превосходит x. Эта задача просто решается для каждого x с помощью бинпоиска за $O(n \log n \log MAXQ)$, в сумме по всем x получим $O(n^2 \log n \log MAXQ)$. Если еще и учесть, что нас интересуют только m значений Q, то бинпоиск осуществлять можно только по ним и получить $O(n^2 \log n \log m)$

По сохраненным значениям для каждого ответа x остается только найти первый ответ, наименьшее значение Q для которого не больше значения Q в запросе. Это можно делать наивно за $O(n)$ или же с помощью бинарного поиска за $O(\log n)$ (значения Q для ответов не возрастают), получая $O(mn)$ или $O(m \log n)$ в сумме.

Наилучшая асимптотика составит $O(n^2 \log n \log m + m \log n)$, однако решения за $O(n^2 \log n \log MAXQ + mn)$ тоже проходят все тесты.