Homework

Panov Ivan, M3139

21-12-2019

Задача 1

Задача 2

Задача 3

Для подсчета ответа воспользуемся методом динамического программирования. Будем считать $dp(mask)$ – два значения($count$, $capacity$): минимальное количество рюкзаков, чтобы взять предметы с номерами равным единичным битам из $mask$ и заполненость текущего рюкзака. Рассмотрим переход: переберем последний предмет, проверим есть ли бит с таким номером в $mask$, если есть и $dp(mask).count > dp(newmask).count$ или $(dp(mask).count = dp(newmask).count)$ and $(dp(mask).capacity > dp(newmask).capacity + w_i)$ обновим $dp(mask)$, обновленное значение равно минимуму по всем $newmask$: если $dp(newmask).capacity + w_i \leq S$, то $[dp(newmask).count, dp(newmask).capacity + w_i]$, иначе $[dp(newmask).count + 1, w_i]$, где i – индекс перебираемого предмета, $newmask$ - $mask$ с нулем в i-ом бите, w_i – вес i-го предмета, минимум пары определен, как сравнение сначала по $count$ потом по $capacity$. Ответ хранится в $dp(ansmask).count$, $ansmask$ – маска с единицами во всез битах от 0 до $n - 1$. Итоговое время работы $\mathcal{O}(2^n n)$.

Задача 4

Допустим мы пытаемся построить дерево, которое дано в условие. Изначально, есть только корень и неравенство выполнаяется, если для любой вершины добавить двух сыновей, то сумма не изменится, так как одно слагаемое превратится в два, которые в два раза меньше. Если же мы добавили всего одного ребенка, то член в сумме уменьшился в двое, что не может нарушить неравенство. Заметим, что данное неравенство становится равенстом при следующем условии – у каждой вершины либо один ребенок, либо два.

Задача 5

Рассмотрим начальную и конечную вершину (x и y), найдем их наименьшего общего предка LCA. Расстояние от x до LCA и от y до LCA не больше $\mathcal{O}(\log(n))$. Все вершины, которые лежат между x и y точно были посещены, так как мы постоянно искали следующую вершину и все из этих вершин больше x и меньше y, следовательно всего их k, каждую из этих вершин мы посетили не больше 3 раз: спустились в первый раз, перед переходом в правую вершину и перед возвращением в предка. Следовательно общая сложность – $\mathcal{O}(\log(n) + k)$.