Задача 1. Башня из кубиков

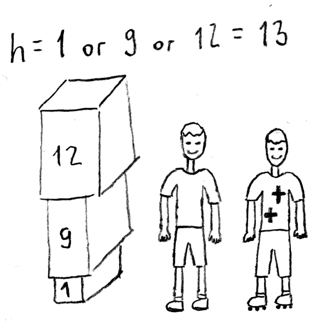
Имя входного файла:	`стандартный ввод`
Имя выходного файла:	`стандартный вывод`
Ограничение по времени:	1 секунда
Ограничение по памяти:	256 мегабайт

Мальчик Петя и его друг, робот Petya++, строят башни из кубиков. У них в распоряжении имеется n кубиков, каждый из которых имеет *высоту* a_i.

Чтобы построить башню, ребята могут взять некоторое непустое подмножество из имеющихся кубиков и поставить их друг на друга. Поскольку кубики не совсем обычные, высота башни будет считаться как побитовое ИЛИ высот всех кубиков. Например, если Петя и Petya++ построят башню из кубиков с высотами 1, 9 и 12, то тогда высота башни составит $1 \text{ or } 9 \text{ or } 12 = 13$.

Ребята интересуются, какую высоту может иметь построенная башня. Для этого они сначала строят башню высоты ровно 0, затем — башню высоты ровно 1, затем — башню высоты ровно 2, затем — башню высоты ровно 3, и так далее. Они заканчивают, как только не могут построить башню некоторой высоты i.

Ребята уже устали строить башни из кубиков, потому им стало интересно, башню какой наименьшей неотрицательной высоты они не смогут построить. Petya++ быстро смог решить эту задачу, но не уверен в своем ответе. Ему необходимо, чтобы кто-то еще решил эту задачу и смог проверить его ответ.



Формат входных данных

В первой строке входных данных задано целое число n ($1 \leqslant n \leqslant 2 \cdot 10^5$) — количество кубиков в распоряжении у ребят.

Во второй строке входных данных заданы n целых чисел $a_1, a_2, \ldots, a_n$ ($0 \leqslant a_i \leqslant 2^{60} - 1$) — высота i-го кубика.

Формат выходных данных

Выведите одно целое число — минимальная неотрицательная высота башни, которую не смогут построить ребята.

Система оценки

№	Дополнительные ограничения	Баллы за подзадачу	Необходимые подзадачи
1	$n \leqslant 10$, $a_i \leqslant 2^{10} - 1$	4	
2	$n \leqslant 20$	4	1
3	$n \leqslant 2000$, $a_i \leqslant 2^{10} - 1$	16	1
4	Нет дополнительных ограничений	76	1 - 3

Примеры

стандартный ввод	стандартный вывод
6 0 1 2 3 4 5	8
3 3 0 1	2
3 1 1 1	0

Замечание

ИЛИ — операция, которая на вход принимает два бита и возвращает бит 0, если оба бита на входе равны нулю и 1 в противном случае. Например: $1 \text{ or } 0 = 1$, $0 \text{ or } 0 = 0$. Для применения побитового ИЛИ двух чисел эти числа сначала переводят в двоичную систему счисления, а затем применяют ИЛИ к каждому из разрядов. Например, $6 \text{ or } 3 = 7$, поскольку $6 = 110_2$, $3 = 11_2$. Применив операцию ИЛИ поразрядно, получаем $7 = 111_2$:

$$\begin{array}{rr} & 110_2 \\ \text{or} & 11_2 \\ \hline & 111_2 \end{array}$$

Теперь перейдем к первому примеру из условия.

В этом примере башни всех высот от 0 до 5 можно собрать ровно из одного кубика соответствующей высоты. Башню высоты 6 можно собрать из кубиков высоты 4 и 2, поскольку $4 \text{ or } 2 = 6$. Башню высоты 7 можно собрать из кубиков высоты 5 и 3, поскольку $5 \text{ or } 3 = 7$. Существуют и другие способы собрать башню высоты 7. Нетрудно убедиться, что башню высоты 8 собрать невозможно.

Задача 2. Галактика удачи

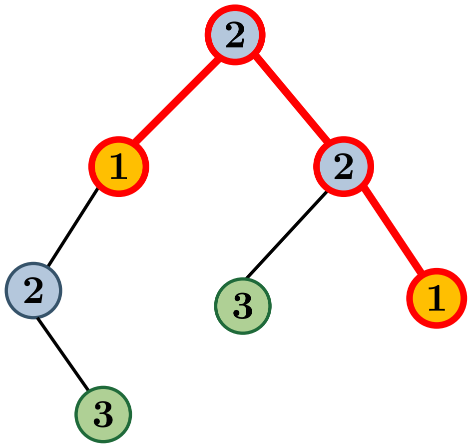
Имя входного файла:	`стандартный ввод`
Имя выходного файла:	`стандартный вывод`
Ограничение по времени:	1 секунда
Ограничение по памяти:	256 мегабайт

Известное во всей Солнечной системе казино «Галактика удачи» решило установить новый игровой автомат.

Смысл игры, которую предлагает игровой автомат, заключается в следующем. На экране показывается дерево, состоящее из n вершин. Напоминаем, что деревом называется связный граф, не содержащий циклов. На каждой вершине дерева записана некоторая буква венерианского алфавита. Венерианский алфавит состоит из c букв, которые для простоты будем обозначать целыми числами от 1 до c.

Когда посетитель казино хочет сыграть в новую игру, он кидает монетку в автомат. Затем автомат выбирает некоторые две различные вершины u и v в дереве и составляет строку из всех букв на пути от u до v. Если эта строка оказалась палиндромом, то игрок выигрывает крупный приз. Напомним, что строка называется палиндромом, если она одинаково читается как слева направо, так и справа налево.

Пусть, к примеру, игровой автомат содержит дерево, изображенное на рисунке ниже. Тогда строка на отмеченном пути равна 1221. Она является палиндромом, поэтому игрок выигрывает.



Может получиться так, что автомат ни при каких значениях u и v не выдаст строку-палиндром. Такой автомат называется *несправедливым*. Проблема несправедливых автоматов заключается в том, что никто не захочет в них играть, поскольку никак нельзя выиграть. Несправедливые автоматы представляют большую проблему для казино, поэтому руководство хочет оценить, насколько велик шанс получить несправедливый автомат. Необходимо решить следующую задачу: посчитать количество способов расставить буквы на заданном дереве таким образом, чтобы построенный автомат был несправедливым. Ответ на задачу может быть поистине огромным, поэтому необходимо найти лишь его остаток от деления на число $10^9 + 7$.

Формат входных данных

В первой строке входных данных находится два целых числа n и c ($1 \leqslant n \leqslant 3 \cdot 10^5$, $2 \leqslant c \leqslant 10^9$) — количество вершин в дереве и количество букв в венерианском алфавите.

В каждой из следующих $n-1$ строк находятся два целых числа u_i и v_i ($1 \leqslant u_i, v_i \leqslant n$, $u_i \neq v_i$) — ребра в заданном дереве.

Формат выходных данных

Выведите одно целое число — количество способов расставить буквы таким образом, чтобы получить несправедливый автомат, по модулю $10^9 + 7$.

Система оценки

№	Дополнительные ограничения	Баллы за подзадачу	Необходимые подзадачи
1	$u_i = i$, $v_i = i + 1$	5	
2	$u_i = 1$, $v_i = i + 1$	5	
3	$c = 2$	1	
4	$c \leqslant 3$	4	3
5	$n \leqslant 8$, $c \leqslant 8$	6	
6	$n \leqslant 10$	12	5
7	$n \leqslant 15$	9	5, 6
8	$n \leqslant 2000$	21	5 - 7
9	Нет дополнительных ограничений	37	1 - 8

Примеры

стандартный ввод	стандартный вывод
4 3 1 2 1 3 3 4	6
2 10 1 2	90

Замечание

В первом примере несправедливыми будут автоматы, в вершинах которых записаны буквы $\{1, 2, 3, 2\}$, $\{1, 3, 2, 3\}$, $\{2, 1, 3, 1\}$, $\{2, 3, 1, 3\}$, $\{3, 1, 2, 1\}$, и $\{3, 2, 1, 2\}$. Итого получаем шесть вариантов.

Во втором примере несправедливыми будут автоматы, на вершинах которых записаны две разные буквы. Получаем $10^2 - 10 = 90$ вариантов.

Задача 3. Двоичная сбалансированная яблоня

Имя входного файла:	`стандартный ввод`
Имя выходного файла:	`стандартный вывод`
Ограничение по времени:	4 секунды
Ограничение по памяти:	256 мегабайт

Если вам еще не надоели задачи, в которых фигурируют деревья, то держите еще одну.

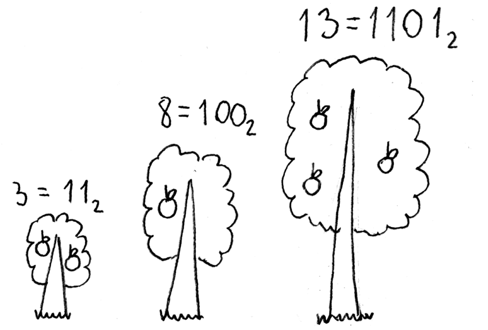
Марсианский садовник Казимир Казимирович выращивает у себя в саду двоичную сбалансированную яблоню. Этот вид деревьев достаточно неприхотлив к погоде, не требует много удобрений и дает хороший урожай.

У садовника есть n двоичных сбалансированных яблонь, которые растут в ряд и пронумерованы целыми числами от 1 до n слева направо. Изначально i-я яблоня имеет высоту a_i. Затем каждый день происходит одно из двух событий следующего вида:

1. Над садом Казимира Казимировича проходит дождь, в результате чего все яблони с номерами от l до r включительно увеличивают свою высоту на k.

2. Садовник собирает яблоки со всех яблонь от l до r включительно. Ввиду того, что яблони двоичные и сбалансированные, количество яблок, которые можно собрать с дерева, вычисляются необычным образом: для i-й яблони оно равно количеству единичных бит в двоичной записи a_i, где a_i — высота этой яблони. Суммарное количество яблок, которое соберет Казимир Казимирович, будет равно суммарному количеству единичных бит в двоичной записи всех чисел $a_l, a_{l+1}, \ldots, a_r$.

Садовник хочет запланировать, когда ему выгодно продавать яблоки. Поэтому Казимир Казимирович хочет знать для каждого события второго типа, сколько яблок он соберет. Для этого садовнику понадобится Ваша помощь: вы должны написать программу, которая сможет обработать все события быстро и эффективно.



Формат входных данных

В первой строке входных данных находится два целых числа n и q ($1 \leqslant n, q \leqslant 2^{18}$) — количество двоичных сбалансированных яблонь в саду у Казимира Казимировича и количество событий.

Во второй строке входных данных находится n целых чисел $a_1, a_2, \ldots, a_n$ ($0 \leqslant a_i \leqslant 2^{18} - 1$) — высота i-й яблони.

В каждой из следующих q строк находится описание i-го события. Описание начинается с целого числа t_i ($1 \leqslant t_i \leqslant 2$) — тип i-го события. Далее, в зависимости от типа события, входные данные имеют следующий вид:

- Если $t_i = 1$, то далее записаны три целых числа l_i, r_i, k_i ($1 \leqslant l_i \leqslant r_i \leqslant n$, $0 \leqslant k_i \leqslant 2^{18} - 1$). Это обозначает, что высота всех яблонь с l_i по r_i включительно увеличилась на k_i.

- Если $t_i = 2$, то далее записаны два целых числа l_i и r_i ($1 \leqslant l_i \leqslant r_i \leqslant n$). Это обозначает, что Казимир Казимирович собрал яблоки со всех яблонь с l_i по r_i включительно.

Гарантируется, что высота каждой яблони в ходе обработки событий не превысит $2^{18}-1$.

Формат выходных данных

Для каждого события второго типа выведите целое число в отдельной строке — количество яблок, которое собрал Казимир Казимирович. Ответы необходимо выводить в том же порядке, в котором заданы события во входных данных.

Система оценки

№	Дополнительные ограничения	Баллы за подзадачу	Необходимые подзадачи
1	$n, q \leqslant 2000$, a_i всегда не превосходят $2^{16}-1$	4	
2	$n, q \leqslant 10^4$, a_i всегда не превосходят $2^{16}-1$	11	1
3	не более 100 событий с $t_i = 1$	6	
4	не более 100 событий с $t_i = 2$	6	
5	$n, q \leqslant 2^{16}$, a_i всегда не превосходят $2^{16}-1$	42	1, 2
6	Нет дополнительных ограничений	31	1 - 5

Заметьте, что фразу «a_i всегда не превосходят $2^{16}-1$» следует читать как «$a_i \leqslant 2^{16}-1$ и после каждого события первого типа высота каждой яблони не будет превосходить $2^{16}-1$».

Пример

стандартный ввод	стандартный вывод
5 6 5 3 8 13 7 2 2 4 1 1 3 2 2 1 5 1 3 5 1 2 2 5 2 1 3	6 13 9 8

Замечание

Рассмотрим пример из условия.

В первый день Казимир Казимирович собирает яблоки. С яблони высоты 3 он получит два яблока, посольку $3 = 11_2$. Яблоня высоты 8 даст одно яблоко, т. к. $8 = 1000_2$. А яблоня высоты 13 даст три яблока, поскольку $13 = 1101_2$. Итого садовник получит $2 + 1 + 3 = 6$ яблок.

После второго дня высоты яблонь станут равны 7 5 10 13 7.

В третий день Казимир Казимирович собирает яблоки со всех яблонь. Поскольку $7 = 111_2$, $5 = 101_2$, $10 = 1001_2$, $13 = 1101_2$, $7 = 111_2$, то он получает $3 + 2 + 2 + 3 + 3 = 13$ яблок.

После четвертого дня высоты яблонь станут равны 7 5 11 14 8.

В пятый день Казимир Казимирович получит два яблока с яблони высоты 5, три яблока с яблони высоты 11, три яблока с яблони высоты 14 и одно яблоко с яблони высоты 8. Итого $2+3+3+1 = 9$.

В шестой день Казимир Казимирович получит три яблока с яблони высоты 7, два яблока с яблони высоты 5 и три яблока с яблони высоты 11. Итого $3 + 2 + 3 = 8$ яблок.

Задача 4. Антинейтринная ракета

Имя входного файла:	inputX.txt
Имя выходного файла:	outputX.txt
Ограничение по времени:	отсутствует
Ограничение по памяти:	отсутствует

После долгих и мучительных экспериментов марсианские ученые наконец-то смогли разработать первый антинейтринный двигатель. Дело осталось за малым — собрать ракету, которая поддерживает этот тип двигателей, а затем провести испытания и удостовериться в надежности построенной ракеты.

Задача проектирования ракеты была поручена уже известным Вам инженерам, Тсетноку и Ачадазу.

Инженеры решили, что двигательный отсек ракеты будет представлять из себя матрицу, состоящую из n строк и m столбцов. В каждой ячейке матрицы может помещаться двигатель одного из пяти типов, которые для удобства пронумерованы целыми числами от 0 до 4. Известно, что двигатель типа i имеет *мощность* p_i.

Изначально двигательный отсек пуст. За одно действие инженеры могут поместить двигатель в любую из ячеек отсека. Если в данной ячейке уже находится двигатель, то он предварительно удаляется, а затем ставится новый. При этом должно выполняться важное условие: чтобы поставить двигатель типа i в некоторую ячейку, в соседних по стороне ячейках должен находиться хотя бы один двигатель каждого из типов $0, 1, \ldots, i-1$. Таким образом, двигатель типа 0 можно установить в любую ячейку, а для установки двигателя типа 4 необходимо, чтобы все четыре соседних по стороне ячейки были заняты двигателями типов 0, 1, 2 и 3. Заметьте, что это правило не обязательно должно выполняться в дальнейшем, важно лишь, чтобы оно было выполнено при установке двигателя.

Конечно же, чтобы ракета набирала как можно большую скорость, суммарная мощность всех двигателей должна быть как можно больше. Тсетнок и Ачадаз уже два дня ломают голову, как оптимально разместить двигатели. Поэтому они обратились за помощью к Вам.

Формат входных данных

Это задача с открытыми тестами.

Входные данные находятся в файлах input1.txt, input2.txt, …, input10.txt.

Первая строка входных данных содержит целое число t — номер теста (для примера из условия $t = 0$).

Вторая строка входных данных содержит три целых положительных числа n, m и w_{max} — размеры двигательного отсека и желательная мощность построенной ракеты (про эту величину смотрите подробнее в разделе «Система оценивания»).

Третья строка входных данных содержит пять целых положительных чисел p_0, p_1, p_2, p_3, p_4 ($p_0 \leqslant p_1 \leqslant p_2 \leqslant p_3 \leqslant p_4$) — мощность двигателя i-го типа.

Формат выходных данных

На проверку необходимо сдать выходные файлы с названиями output1.txt, output2.txt, …, output10.txt, где выходной файл outputX.txt должен соответствовать входному файлу inputX.txt.

В первой строке выходных данных выведите два целых положительных числа w и k ($1 \leqslant k \leqslant 5 \cdot 10^5$) — суммарная мощность двигателей в Вашем решении и количество действий, необходимое для достижения заданной мощности.

В каждой из следующих k строк выходных данных выведите три целых числа x_i, y_i и t_i ($1 \leqslant x_i \leqslant n$, $1 \leqslant y_i \leqslant m$, $0 \leqslant t_i \leqslant 4$). Это обозначает, что i-м действием Вы помещаете двигатель типа t_i в ячейку двигательного отсека с координатами (x_i, y_i).

Система оценки

Если выходной файл не соответствует указанному формату выходных данных, то Вы получите 0 баллов за тест.

Если Вы нарушили правила установки двигателя, т. е. вы установили двигатель типа i при условии, что рядом с ним не находится двигатель некоторого типа j $(j < i)$, то ракета взрывается, а Вы получаете 0 баллов за тест.

Иначе Ваш балл равен

$$\min\left(10, 10 \cdot \left(\frac{w}{w_{max}}\right)^2\right),$$

где w — суммарная мощность двигателей ракеты в построенном Вами решении, а w_{max} — желательная суммарная мощность ракеты, заданная во входных данных.

Баллы за каждый тест округляются вверх до сотых и суммируются. Правила округления таковы, что, например, при округлении числа 10.112 вверх до сотых получаем число 10.12.

Гарантируется, что возможно расположить двигатели согласно описанным выше правилам таким образом, чтобы их суммарная мощность была равна w_{max}, а количество действий не превосходило $5 \cdot 10^5$.

Пример

inputX.txt	outputX.txt
0 2 2 30 3 7 10 15 18	27 5 1 1 0 1 2 0 2 1 1 2 2 2 1 1 1

Замечание

В примере выше итоговая расстановка двигателей будет выглядеть следующим образом:

```
10
12
```

Заметим, что около двигателя типа 1 с координатами $(2, 1)$ по соседству нет двигателя типа 0. Но это не означает, что требование на соседние двигатели нарушено, поскольку оно должно выполняться только при установке двигателя.

Можно заметить, что суммарная мощность двигателей равна $3 + 7 + 7 + 10 = 27$, а w_{max} во входных данных равен 30. По этой причине такой ответ получит $10 \cdot \left(\frac{27}{30}\right)^2 = 8.1$ балла за тест.