Задача A. Древняя цивилизация

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

Марсианские ученые исследуют Ганимед, один из многочисленных спутников Юпитера. Недавно на нем были обнаружены руины древней цивилизации. Ученые доставили на Марс несколько табличек с надписями на неизвестном науке языке.

Было установлено, что жители Ганимеда использовали алфавит из двух букв, причем каждое слово имело фиксированную длину — ровно ℓ букв. Поэтому ученые решили записывать каждое слово этого языка как целое положительное число от 0 до $2^\ell - 1$. Первой букве алфавита соответствуют нули в двоичной записи числа, а второй букве алфавита — единицы.

Одно и то же слово может иметь в языке разные формы. Тогда необходимо восстановить начальную форму слова. Ниже описано, как это делается.

Назовем *расстоянием* между двумя словами количество позиций, в которых эти слова различаются. Например, расстояние между словами 1001_2 и 1100_2 (в двоичной записи) равно двум, т. к. они имеют разные буквы на второй и четвертой позициях, если считать слева направо. Будем в дальнейшем обозначать расстояние между словами x и y как $d(x, y)$.

Пусть слово имеет n форм, i-я из которых описывается целым числом x_i. Все x_i не обязательно различны, т. к. две разные формы слова могут записываться одинаково. Рассмотрим некоторое слово y. Тогда *близость* слова y равна сумме расстояний до каждой из форм исследуемого слова, т. е. сумме $d(x_i, y)$ по всем $1 \leqslant i \leqslant n$.

Начальной формой является слово y с минимально возможной близостью.

Вам необходимо помочь ученым и написать программу, которая находит начальную форму слова по всем его известным формам. Обратите внимание, что начальная форма **не обязательно** должна совпадать с какой-либо из известных n форм слова.

Формат входных данных

В первой строке входных данных находится два целых числа n и ℓ ($1 \leqslant n \leqslant 3 \cdot 10^5$, $1 \leqslant \ell \leqslant 30$) — количество форм слова и количество букв в одной форме.

Во второй строке находится n целых чисел x_i ($0 \leqslant x_i \leqslant 2^\ell - 1$) — формы слова. Числа не обязательно являются различными.

Формат выходных данных

Выведите одно целое число — начальную форму слова, т. е. такое y ($0 \leqslant y \leqslant 2^\ell - 1$), что сумма $d(x_i, y)$ по всем $1 \leqslant i \leqslant n$ минимально возможная. Обратите внимание, что y не обязано совпадать с каким-либо числом из x_i.

Если вариантов восстановить начальную форму слова несколько, выведите любой из них.

Система оценки

№	Дополнительные ограничения	Баллы за подзадачу	Необходимые подзадачи
1	$n \leqslant 3000$, $\ell \leqslant 2$	8	
2	$n \leqslant 3000$, $\ell \leqslant 12$	15	1
3	$\ell \leqslant 12$	19	1 - 2
4	$\ell \leqslant 18$	27	1 - 3
5	Нет дополнительных ограничений	31	1 - 4

Примеры

стандартный ввод	стандартный вывод
3 5 18 9 21	17
3 5 18 18 18	18

Замечание

Рассмотрим примеры из условия.

В первом примере слова в двоичной записи записываются как $x_1 = 10010_2$, $x_2 = 01001_2$ и $x_3 = 10101_2$. Пусть $y = 10001_2$. Тогда $d(x_1, y) = 2$ (различие в четвертой и пятой позиции), $d(x_2, y) = 2$ (различие в первой и второй позиции), $d(x_3, y) = 1$ (различие в третьей позиции). Получаем, что близость равна $2 + 2 + 1 = 5$. Можно показать, что меньшего значения близости добиться невозможно.

Во втором примере все формы слова одинаковы и равны 18 (10010_2 в двоичной записи), поэтому начальная форма тоже равна 18. Как нетрудно догадаться, близость в таком случае равна нулю.

Задача B. Пауки на дереве

Имя входного файла:	`стандартный ввод`
Имя выходного файла:	`стандартный вывод`
Ограничение по времени:	2 секунды
Ограничение по памяти:	256 мегабайт

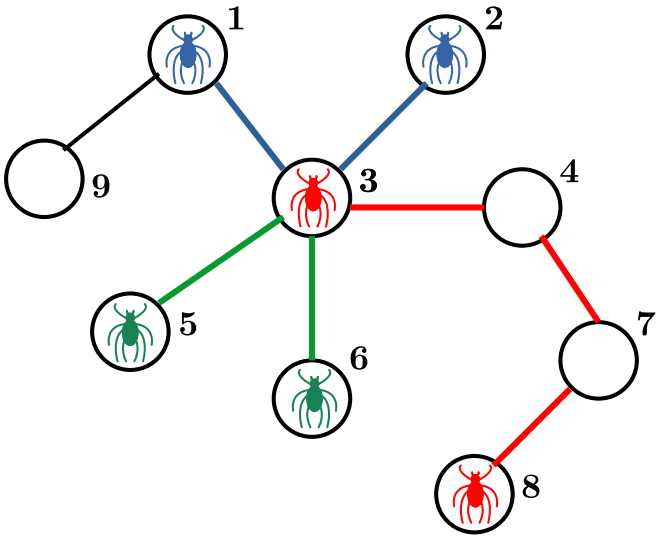
Пока ученые занимаются исследованием языка древней цивилизации с Ганимеда, Вы продолжили наблюдение за Двоичными пауками.

Двоичные пауки обычно обитают на деревьях. А дерево, как известно из теории графов, является связным ациклическим графом. Иными словами, дерево состоит из n вершин и $n-1$ ребра между ними. Каждое ребро соединяет некоторые две вершины дерева. Из любой вершины дерева можно добраться по ребрам до любой другой. Также дерево не содержит циклов, т. е. нельзя построить путь по ребрам из некоторой вершины и вернуться в нее, не пройдя по одному и тому же ребру дважды.

Известно, что пауки объединяются в группы. Каждая группа пауков живет в двух гнездах, не обязательно расположенных близко друг к другу. Гнездо должно располагаться в вершине дерева, причем в каждой вершине должно находиться не более одного гнезда.

На дереве может жить несколько групп пауков. После расселения по дереву каждая группа должна построить путь между своими гнездами по ребрам дерева. Пауки из разных групп договариваются между собой так, чтобы не мешать друг другу, т. е. чтобы никакие два пути не имели общего ребра. При этом допускается пересечение путей в вершине. Если пауки не смогут договориться, то им придется воевать друг другом за территорию, а Вы хотите избежать ненужного кровопролития.

Рассмотрим пример:



Здесь первая группа пауков (обозначена синим) имеет гнезда в вершинах 1 и 2, вторая группа пауков (обозначена зеленым) — в вершинах 5 и 6, и, наконец, третья группа пауков имеет гнезда в вершинах 3 и 8. Цвет ребра соответствует группе пауков, у которой путь содержит это ребро, а остальные ребра покрашены в черный чвет. Обратите внимание, что все пути проходят через вершину 3, и конфликта при этом не происходит, поскольку пути все еще не пересекаются по ребрам.

Вы отметили на дереве q вершин, причем q — четное число. В каждой из отмеченных вершин будет находиться гнездо паука. Теперь Вам необходимо разместить $\frac{q}{2}$ групп пауков по вершинам таким образом, чтобы они не воевали между собой.

Формат входных данных

В первой строке входных данных находится два целых числа n и q ($2 \leqslant n \leqslant 4 \cdot 10^5$, $2 \leqslant q \leqslant n$, q четно) — общее количество вершин в дереве и количество отмеченных вершин.

Во второй строке входных данных находится q различных целых чисел m_i ($1 \leqslant m_i \leqslant n$) — номера отмеченных вершин.

В каждой из следующих $n-1$ строк входных данных находится два целых u_i и v_i ($1 \leqslant u_i, v_i \leqslant n$, $u_i \neq v_i$). Это означает, что между вершинами u_i и v_i в дереве проведено ребро.

Гарантируется, что ребра во входных данных образуют дерево.

Формат выходных данных

Выходные данные должны содержать $\frac{q}{2}$ строк.

В каждой строке должно содержаться два целых числа x_i и y_i ($1 \leqslant x_i, y_i \leqslant n$) — номера вершин, в которых будут располагаться гнезда i-й группы пауков.

Все числа в выходных данных должны быть различны. Каждая отмеченная вершина должна содержать ровно одно гнездо, остальные вершины не должны содержать гнезд. Расположение гнезд должно быть выбрано таким образом, чтобы пауки не воевали, т. е. чтобы было возможно построить непересекающиеся по ребрам пути между каждой парой гнезд, принадлежащих одной группе.

Если разместить пауков требуемым образом невозможно, выведите вместо этого число -1.

Система оценки

№	Дополнительные ограничения	Баллы за подзадачу	Необходимые подзадачи
1	$x_i = i$, $y_i = i + 1$	8	
2	$x_i = 1$, $y_i = i + 1$	8	
3	$n \leqslant 3000$, $q \leqslant 6$	15	
4	$n \leqslant 3000$, $q = n$	13	
5	$n \leqslant 3000$	16	3 - 4
6	$q = n$	12	4
7	$n \leqslant 2 \cdot 10^5$	17	3 - 5
8	Нет дополнительных ограничений	11	1 - 7

Пример

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

Замечание

Пример из условия разобран на иллюстрации выше.

Задача C. Галактический сокол

Имя входного файла:	`стандартный ввод`
Имя выходного файла:	`стандартный вывод`
Ограничение по времени:	2 секунды
Ограничение по памяти:	256 мегабайт

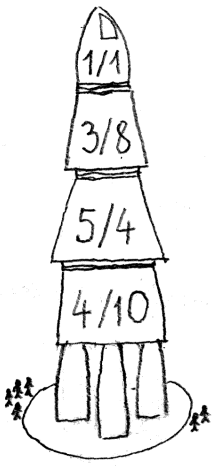
После долгих экспериментов ученые наконец получили то, к чему так долго стремились — новый, быстрый двигатель для ракет. Чтобы испытать двигатель в реальных полетах, было решено построить для него ракету под названием «Галактический сокол».

«Галактический сокол» будет состоять из нескольких ступеней, которые стоят одна на другой. Всего было выпущено n ступеней. Все выпущенные ступени пронумерованы целыми числами от 1 до n. Ступень с номером i имеет *массу* m_i и *прочность* p_i.

Ракета должна быть надежной. А для этого необходимо, чтобы прочность каждой ступени была не меньше, чем суммарная масса всех ступеней, расположенных выше нее.

К сожалению, конструкторы «Галактического сокола» очень торопились и не успели рассчитать, можно ли из имеющихся ступеней собрать ракету. Поэтому они решили использовать при построении ракеты как можно больше ступеней из имеющихся, чтобы ракета при этом была надежной. Понятно, что каждую ступень можно использовать не более одного раза.

Весь Марс ожидает запуска «Галактического сокола», поэтому необходимо как можно скорее решить, из каких ступеней будет состоять ракета. Вам поручили решить эту непростую задачу.



Формат входных данных

В первой строке входных данных находится целое число n ($1 \leqslant n \leqslant 3 \cdot 10^5$) — количество ступеней.

В каждой из следующих n строк находится два целых числа m_i и p_i ($1 \leqslant m_i \leqslant 10^9, 0 \leqslant p_i \leqslant 10^9$) — масса и прочность i-й ступени.

Формат выходных данных

В первой строке выходных данных выведите одно целое число k ($1 \leqslant k \leqslant n$) — максимально возможное количество ступеней в ракете «Галактический сокол».

Во второй строке выходных данных выведите k различных целых чисел a_i ($1 \leqslant a_i \leqslant n$) — номер ступени, которая будет стоять в ракете на i-й позиции. Позиции в ракете нумеруются сверху вниз, т. е. первая позиция соответствует самой верхней ступени, а k-я — самой нижней.

Построенная Вами ракета должна быть надежной.

Если существует несколько построить ракету, то разрешается вывести любой из них.

Система оценки

№	Дополнительные ограничения	Баллы за подзадачу	Необходимые подзадачи
1	$n \leqslant 9$	7	
2	$n \leqslant 21$	12	1
3	$n \leqslant 40$	5	1 - 2
4	$n \leqslant 300$	19	1 - 3
5	$n \leqslant 3000$, $m_i = 1$	6	
6	$n \leqslant 3000$	16	1 - 5
7	$m_i = 1$	8	5
8	Нет дополнительных ограничений	27	1 - 7

Примеры

стандартный ввод	стандартный вывод
6 1 1 4 10 2 2 6 3 5 4 3 8	4 1 6 5 2
4 1 0 1 0 1 0 1 0	1 1

Замечание

Рассмотрим первый пример. Построенная ракета показана на рисунке выше.

Можно показать, что эта ракета является надежной. Будем нумеровать ступени в порядке сверху вниз:

- Выше второй ступени находится ступень с массой 1, а ее надежность равна $8 \geqslant 1$.
- Выше третьей ступени находятся ступени с массой 1 и 3, а ее надежность равна $4 \geqslant 1 + 3$.
- Выше четвертой ступени находятся ступени с массой 1, 3 и 5, а ее надежность равна $10 \geqslant 1 + 3 + 5$.

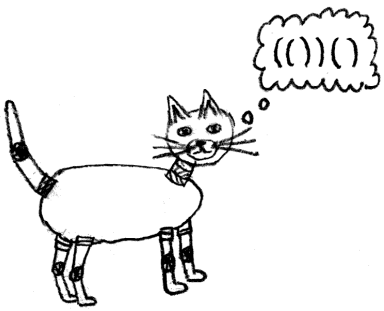
Во втором примере мы не можем взять больше одной ступени. У всех ступеней прочность равна нулю, а это значит, что мы не можем поместить одну ступень поверх другой.

Задача D. Обновление котов

Имя входного файла:	`стандартный ввод`
Имя выходного файла:	`стандартный вывод`
Ограничение по времени:	6 секунд
Ограничение по памяти:	256 мегабайт

Компания «Interplanetary Software, Inc.» вместе с компанией «Robots of Cydonia, Ltd.» разработала и выпустила роботов-котов. Эти электронные питомцы умеют мяукать, ловить мышей и всячески развлекать хозяина.

Недавно разработчики из «Interplanetary Software, Inc.» решили выпустить обновление программного обеспечения для этих роботов. После обновления коты должны уметь решать задачи, связанные со скобочными последовательностями. Одна из таких задач приведена ниже.



Сначала немного погрузимся в теорию скобочных последовательностей. Будем рассматривать строки, состоящие из символов «`(`», «`)`» и «`.`». Назовем строку *правильной скобочной последовательностью (ПСП)*, если из нее можно получить пустую строку путем операций удаления одиночных символов «`.`» либо подряд идущих символов «`()`». Например, строка «`((()(.))`» является ПСП, т. к. с ней можно проделать следующую цепочку удалений:

«`((()(.))`» → «`((())`» → «`(())`» → «`()`» → «».

Мы получили пустую строку, а это значит, что исходная строка является ПСП. В то же время, строка «`)(`» не является ПСП, поскольку к ней нельзя применить описанные выше операции удаления.

Будем называть ПСП *простой*, если эта ПСП непуста, не начинается на «`.`» и не заканчивается на «`.`».

Также будем считать, что *подстрока* строки s — это ее последовательный подотрезок. В частности, $s[l \dots r] = s_l s_{l+1} \dots s_r$, где s_i — i-й символ строки s.

Теперь перейдем к формулировке задачи. Вам дана строка s, изначально состоящая из символов «`(`» и «`)`». Необходимо отвечать на следующие запросы:

1. Даны индексы l и r ($1 \leqslant l < r \leqslant n$). Гарантируется, что l-й символ строки равен «`(`», r-й символ строки равен «`)`», а остальные символы строки равны «`.`». Тогда требуется сделать l-й и r-й символ равными «`.`».

2. Даны индексы l и r ($1 \leqslant l < r \leqslant n$), причем гарантируется, что подстрока $s[l \dots r]$ — простая ПСП. Требуется найти количество подстрок в $s[l \dots r]$, которые являются простыми ПСП. Иными словами, необходимо найти количество пар индексов i, j таких, что $l \leqslant i < j \leqslant r$ и $s[i \dots j]$ является простой ПСП.

Вы работаете в «Interplanetary Software, Inc.» и Вам поручили непростое задание: научить котов после обновления решать описанную выше задачу.

Формат входных данных

В первой строке входных данных находится два целых числа n и q ($1 \leqslant n \leqslant 3{\cdot}10^5$, $1 \leqslant q \leqslant 3{\cdot}10^5$) — длина строки и количество запросов.

Во второй строке находится строка s, состоящая из n символов «(» и «)».

В каждой из следующих q строк находится по три числа t, l и r ($t \in \{1, 2\}$, $1 \leqslant l < r \leqslant n$) — запросы, которые требуется обработать. Гарантируется, что все запросы корректны и удовлетворяют условию задачи.

Формат выходных данных

Для каждого запроса второго типа выведите по одному целому числу в отдельной строке — количество подстрок, которые являются простыми ПСП. Ответы требуется выводить в том же, порядке, в котором приведены сами запросы.

Система оценки

№	Дополнительные ограничения	Баллы за подзадачу	Необходимые подзадачи
1	$n \leqslant 300$, $q = 1$, $t_i = 2$	4	
2	$n \leqslant 3000$, $q = 1$, $t_i = 2$	7	1
3	$q = 1$, $t_i = 2$	14	1 - 2
4	$n, q \leqslant 10^5$, $t_i = 2$	12	1 - 2
5	$n, q \leqslant 10^5$	13	1 - 2, 4
6	$n, q \leqslant 2 \cdot 10^5$, $t_i = 2$	14	1 - 2, 4
7	$n, q \leqslant 2 \cdot 10^5$	13	1 - 2, 4 - 6
8	$t_i = 2$	12	1 - 4, 6
9	Нет дополнительных ограничений	11	1 - 8

Пример

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

Замечание

Рассмотрим пример из условия.

Ответ на первый запрос равен 3, поскольку существует всего три требуемых подстроки: $s[3 \ldots 6]$, $s[3 \ldots 4]$ и $s[5 \ldots 6]$.

Ответ на второй запрос равен 4. Подходящие подстроки — $s[3 \ldots 6]$, $s[3 \ldots 4]$, $s[5 \ldots 6]$ и $s[2 \ldots 7]$.

После третьего запроса строка станет равной «)(..())()».

Ответ на четвертый запрос равен 2. Подходящие подстроки — $s[5 \ldots 6]$ и $s[2 \ldots 7]$. Обратите внимание, что $s[3 \ldots 6]$ больше не является простой ПСП, т. к. начинается с «.».

Ответ на пятый запрос равен 4. Подходящие подстроки — $s[5 \ldots 6]$, $s[2 \ldots 7]$, $s[8 \ldots 9]$ и $s[2 \ldots 9]$.

После шестого запроса строка станет равной «)(....)()».

После седьмого запроса строка станет равной «).....()».

Ответ на восьмой запрос равен 1. Подходящая подстрока — $s[8 \ldots 9]$.