Задача A. Вторичная структура РНК

Имя входного файла:	`rna.in`
Имя выходного файла:	`rna.out`
Ограничение по времени:	2 секунды
Ограничение по памяти:	256 мегабайт

РНК (Рибонуклеиновая кислота) — одна из важнейших макромолекул для всех известных форм жизни. Она состоит из длинной цепочки нуклеотидов. Всего есть четыре различных вида нуклеотидов. Они обозначаются буквами `A`, `C`, `G` и `U`. Первичная структура РНК представляется в виде последовательности из этих четырёх символов. Вторичная структура РНК определяется взаимодействиями пар нуклеотидов. Более точно, основание `A` может соединиться с `U`, а `C` — с `G`. Стабильность вторичной структуры зависит от количества сформировавшихся пар. Итоговая структура будет содержать максимально возможное число пар.

Представим первичную структуру строкой из символов "`ACGU`". Введём правила формирования вторичной структуры:

- Основание `A` может сформировать пару с основанием `U`;
- Основание `C` может сформировать пару с основанием `G`;
- Каждое основание может находиться не более, чем в одной паре;
- Пусть $w < x$, $y < z$ и $w < y$. Если основание в позиции w образует пару с основанием в позиции x и основание в позиции y образует пару с основанием в позиции z, то верно одно из двух утверждений: $y > x$ или $z < x$;
- Может быть не более K пар `C`–`G`.

Вам дана первичная структура РНК особи и вам нужно найти количество пар в итоговой вторичной конфигурации в соответствии с указанными выше правилами формирования пар.

Первичная структура будет дана в формате, использующем сжатие длин серий (RLE). В этом формате идущие подряд одинаковые символы заменяются на пару из символа и числа, соответствующего количеству повторов символа. Например, "`AAAACCGAAUUG`" будет представлено как "`A4C2G1A2U2G1`". Это значит, что первичная структура будет дана в формате $\langle c_1 f_1 c_2 f_2 c_3 f_3 \dots c_n f_n \rangle$, где c_i — один из символов `ACGU`, а f_i — положительное целое число.

Особи, с которыми мы работаем, обладают следующими свойствами:

1. $f_1 + f_2 + f_3 + \ldots + f_n \leqslant 10050$
2. $f_1 \leqslant 5000$
3. $f_n \leqslant 5000$
4. $f_2 + f_3 + \ldots + f_{n-1} \leqslant 50$

Формат входных данных

В первой строке входного файла задано целое число T ($T \leqslant 200$) — количество тестов. Каждый тест описывается двумя строками. В первой из них задана первичная структура РНК с использованием кодирования длин серий. Во второй строке задано число K ($0 \leqslant K \leqslant 20$) — ограничение на число пар `C`–`G` во вторичной структуре.

Формат выходных данных

Для каждого теста, выведите его номер и максимальное число сформированных пар нуклеотидов во вторичной структуре. Смотрите примеры для более точного понимания формата вывода.

Примеры

rna.in	rna.out
3 A3C1G1C1U4A2U1 1 A3C1G1C1U4A2U1 0 A100U200 2	Case 1: 6 Case 2: 5 Case 3: 100

Задача B. Дубы

Имя входного файла:	`oaks.in`
Имя выходного файла:	`oaks.out`
Ограничение по времени:	1.5 секунды
Ограничение по памяти:	256 мегабайт

На аллее перед зданием Министерства Обороны в ряд высажены n дубов. В связи с грядущим приездом главнокомандующего, было принято решение срубить несколько деревьев для придания аллее более милитаристического вида.

Внутренние распорядки министерства позволяют срубать дуб только в двух случаях:

- если и ближайший дуб слева, и ближайший дуб справа строго ниже, чем данный дуб;
- если и ближайший дуб слева, и ближайший дуб справа строго выше, чем данный дуб.

В частности, согласно этому правилу, нельзя срубить крайний левый и крайний правый дубы.

Министр хочет выработать такой план вырубки, чтобы в итоге осталось несколько дубов, высоты которых образуют неубывающую последовательность, то есть чтобы каждый дуб был не ниже, чем все дубы, стоящие слева от него. При этом, как человек любящий флору, министр хочет, чтобы было срублено минимальное возможное количество деревьев.

Помогите сотрудникам министерства составить оптимальный план вырубки аллеи или выяснить, что срубить дубы соответствующим образом невозможно.

Формат входных данных

Первая строка входного файла содержит целое число n — количество дубов, растущих на аллее ($2 \leqslant n \leqslant 200$). Вторая строка содержит n чисел — высоты дубов, приведённые слева направо. Высоты дубов — положительные целые числа, не превышающие 1 000.

Формат выходных данных

Если оставить последовательность дубов с неубывающими высотами невозможно, выходной файл должен содержать только одно число -1.

В случае, если искомый план существует, в первую строку выходного файла выведите целое число m — минимальное количество дубов, которые необходимо срубить. В следующие m строк выведите оптимальный план вырубки деревьев — номера дубов в том порядке, в котором их следует срубать, по одному номеру на строке.

Дубы нумеруются слева направо натуральными числами от 1 до n.

Если планов с наименьшим числом срубаемых дубов несколько, выведите любой из них.

Примеры

oaks.in	oaks.out
5 3 2 4 8 5	2 2 4

Задача C. Пастбища

Имя входного файла:	`pasture.in`
Имя выходного файла:	`pasture.out`
Ограничение по времени:	1 секунда
Ограничение по памяти:	64 мегабайта

Фермер Джон решил снабдить каждую из своих коров сотовым телефоном. Для этого ему требуется установить сотовые станции на его N пастбищах, последовательно пронумерованных от 1 до N.

Ровно $N - 1$ пара пастбищ являются соседними, и для любых двух пастбищ A и B имеется последовательность соседних пастбищ таких, что A — первое пастбище этой последовательности, а B — последнее. Сотовые станции размещаются только на пастбищах и имеют достаточный радиус действия, чтобы обеспечить связью это пастбище и все соседние.

Помогите фермеру Джону определить, какое минимальное количество станций он должен установить, чтобы обеспечить связью все пастбища.

Формат входных данных

На первой строке входного файла находится одно целое число N ($1 \leqslant N \leqslant 100\,000$). Далее следуют $N - 1$ строк, каждая из которых содержит пару целых чисел — очередную пару соседних пастбищ A и B ($1 \leqslant A \leqslant N$; $1 \leqslant B \leqslant N$; $A \neq B$).

Формат выходных данных

Выведите в выходной файл одно число — минимальное достаточное количество станций.

Примеры

pasture.in	pasture.out
5 1 3 5 2 4 3 3 5	2

Задача D. План лекций

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

В процессе подготовки ЛКШ перед преподавателями стоит ряд задач: проверка вступительной, распределение школьников по параллелям и многое другое. Одной из таких задач является составление плана лекций на будущую смену.

С каждым годом популярность и применимость на олимпиадах тех или иных тем меняется, поэтому каждый год необходимо заново выбирать самые полезные темы для лекций. Это выглядит простой задачей, но всё осложняется тем, что некоторые темы нельзя рассказывать раньше других. Например, в младших параллелях первой всегда идёт лекция по языку Python, так как это необходимо для понимания всех последующих лекций. Также нельзя читать лекцию по алгоритму Дейкстры раньше, чем будут рассказаны способы хранения графов.

Чтобы сделать задачу более наглядной, преподаватели используют дерево зависимостей — граф, вершинами которого являются темы, а рёбрами — зависимости. Про каждую тему кроме первой известен номер темы p_i, которую необходимо рассказать перед данной (не обязательно именно в предыдущий день). Помимо этого, про каждую тему известна её полезность w_i. Чтение одной и той же более одного раза обладает нулевой полезностью (мы надеемся).

За много лет было установлено, что граф зависимостей всегда является подвешенным деревом, то есть каждая тема кроме корня дерева зависит ровно от одной другой темы. Это сильно упрощает задачу, так как корень дерева — единственная тема, для которой не ничего не требуется знать, а значит, именно её нужно рассказывать в первый день. Осталось только определить остальные лекции.

Вам предстоит написать программу для нахождения максимальной суммарной полезности лекций в наборе, который можно прочитать. Конечно, читать лекцию v_i можно только если все лекции, в которых содержится материал, необходимый для понимания v_i, уже прочитаны.

Формат входных данных

В первой строке ввода записано два числа n и k ($1 \leqslant k < n \leqslant 4000$) — количество вершин в графе зависимостей и количество лекций соответственно.

Во второй строке записано $n - 1$ число, i-е из которых соответствует номеру предка темы $i + 1$ в дереве зависимостей. Корнем дерева является тема номер 1.

В третьей строке находятся n целых чисел w_i ($0 \leqslant w_i \leqslant 100\,000$) — полезности лекций.

Формат выходных данных

Выведите одно целое число — максимальную суммарную полезность k различных лекций, которую можно достичь.

Примеры

стандартный ввод	стандартный вывод
5 3 1 1 2 3 1 1 2 2 1	4
7 3 1 1 2 3 4 4 1 2 1 2 1 2 1	5

Замечание

В первом тесте из условия оптимальное решение — выбрать лекции 1, 2 и 3. В таком случае суммарная полезность лекций составит 4.

Во втором тесте из условия оптимальное решение — выбрать лекции 1, 2 и 4. В таком случае суммарная полезность лекций составит 5.

Задача E. Покраска дерева

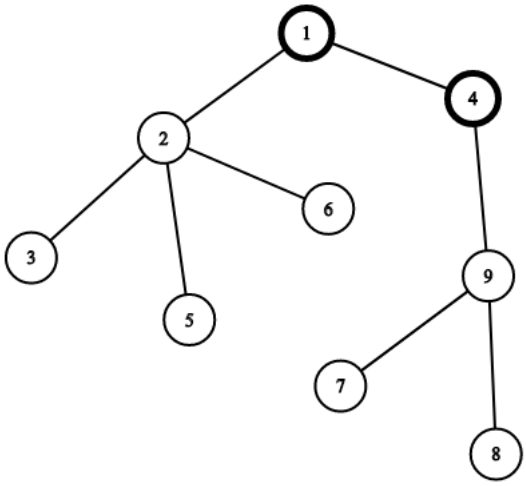
Имя входного файла:	`стандартный ввод`
Имя выходного файла:	`стандартный вывод`
Ограничение по времени:	1 секунда
Ограничение по памяти:	256 мегабайт

Вам задано дерево, состоящее из n вершин. Вы играете в игру на этом дереве.

Изначально все вершины белые. На первом ходу игры Вы выбираете одну вершину и красите её в черный. Затем на каждом ходу Вы выбираете белую вершину, смежную с **любой** чёрной вершиной и красите её в чёрный.

Каждый раз, когда вы выбираете вершину(даже во время первого хода), Вы получаете количество очков, равное размеру компоненты связности, состоящей только из белых вершин, содержащей выбранную вершину. Игра заканчивается, когда все вершины покрашены в чёрный цвет.

Рассмотрим следующий пример:



Вершины 1 и 4 уже покрашены в чёрный цвет. Если Вы выберете вершину 2, Вы получите 4 очка за компоненту связности, состоящую из вершин $2, 3, 5$ и 6. Если Вы выберете вершину 9, Вы получите 3 очка за компоненту связности, состоящую из вершин $7, 8$ и 9.

Ваша задача — максимизировать количество очков, которое Вы получите.

Формат входных данных

Первая строка содержит одно целое число n — количество вершин в дереве ($2 \leqslant n \leqslant 2 \cdot 10^5$).

Каждая из следующих $n - 1$ строк описывает ребро дерева. Ребро i описывается двумя целыми числами u_i и v_i, номерами вершин, которые оно соединяет ($1 \leqslant u_i, v_i \leqslant n, u_i \neq v_i$).

Гарантируется, что заданные рёбра образуют дерево.

Формат выходных данных

Выведите одно целое число — максимальное количество очков, которое вы получите, если будете играть оптимально.

Примеры

стандартный ввод	стандартный вывод
9 1 2 2 3 2 5 2 6 1 4 4 9 9 7 9 8	36
5 1 2 1 3 2 4 2 5	14

Задача F. Сеть

Имя входного файла:	`network.in`
Имя выходного файла:	`network.out`
Ограничение по времени:	1.3 секунды
Ограничение по памяти:	256 мегабайт

В компьютерной сети вашей фирмы n компьютеров. В последнее время свитч, к которому они подключены, сильно барахлит, и потому не любые два компьютера могут связаться друг с другом. Кроме того, если компьютер a обменивается информацией с компьютером b, то никакие другие компьютеры не могут в это время обмениваться информацией ни с a, ни с b. Вам необходимо вычислить максимальное количество компьютеров, которые могут одновременно участвовать в процессе обмена информацией.

Формат входных данных

В первой строке файла задано число n ($1 \leqslant n \leqslant 18$). Далее идут n строк по n символов, причём j-й символ i-й строки равен '`Y`', если i-й и j-й компьютеры могут обмениваться информацией, иначе он равен '`N`'. Верно, что i-й символ i-й строки всегда равен '`N`' и, кроме того, матрица символов симметрична.

Формат выходных данных

Выведите максимальное количество компьютеров, которые могут одновременно участвовать в процессе обмена информацией.

Пример

network.in	network.out
5 NYYYY YNNNN YNNNY YNNNY YNYYN	4

Задача G. Хэдмастеры

Имя входного файла:	`linear.in`
Имя выходного файла:	`linear.out`
Ограничение по времени:	2 секунды
Ограничение по памяти:	64 мегабайта

Уже скоро должен состояться финальный бой между трансформерами. Все вокруг затихло и ждет последнего сражения.

Давайте рассмотрим то, как готовятся к бою хэдмастеры. N хэдмастеров решили расположиться на N целых точках числовой прямой с координатами $1, 2, \ldots, N$. В каждой точке должен оказаться ровно один робот. Единственная загвоздка заключается в том, что M различных пар роботов должны быть соединены специальными кабелями. Кабеля являются очень дорогостоящими, поэтому стратегически важно минимизировать их суммарную длину.

Если робот в точке с координатой x должен быть соединен с роботом, который находится в точке с координатой y, то для их соединения потребуется $|x-y|$ метров кабеля. Помогите хэдмастерам найти минимальное количество кабеля, которое необходимо потратить при оптимальном расположении роботов в указанных точках.

Формат входных данных

В первой строке входного файла записано два числа N ($2 \leqslant N \leqslant 20$) — количество хэдмастеров. Во второй строке находится одно целое число M — количество пар хэдмастеров, которые должны быть соединены. В следующих M строках заданы пары хэдмастеров, которые должны быть соединены. Пара задается ровно двумя натуральными числами, не превышающими N — номерами роботов. В каждой строке содержится ровно одна такая пара. Никакие две пары не совпадают.

Формат выходных данных

Выведите в первую строку выходного файла выведите единственное число — минимальное количество кабеля, которое придется потратить хэдмастерам.

Примеры

linear.in	linear.out
5 3 1 2 2 3 4 5	3

Замечание

Одним из возможных оптимальных расположений роботов может быть следующий: $4, 5, 1, 2, 3$.

Задача H. Раскраска графа

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

Дан граф из n вершин, раскрасьте его в минимально возможное число цветов так, чтобы никакие две вершины, соединенные ребром, не были одного цвета.

Формат входных данных

В первой строке содержится число t — количество тестовых примеров ($1 \leqslant t \leqslant 5$).

Далее содержится t тестовых случаев, заданных в следующем формате:

В первой строке записаны числа n и m — количество вершин и ребер соответственно ($1 \leqslant n \leqslant 17$, $0 \leqslant m \leqslant \frac{n \cdot (n-1)}{2}$).

Затем идет m строк, в которых содержится по два числа v_i u_i, что означает, что вершины v_i и u_i соеденены ребром ($1 \leqslant v_i, u_i \leqslant n, v_i \neq u_i$).

Гарантируется, что все ребра в каждом тестовом случае различны.

Формат выходных данных

Для каждого тестового случая в первой строке выведите минимальное число цветов k.

Во второй строке выведите n чисел a_i — цвета вершин ($1 \leqslant a_i \leqslant k$).

Примеры

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

Задача I. Сокровища

Имя входного файла:	`dowry.in`
Имя выходного файла:	`dowry.out`
Ограничение по времени:	2 секунды
Ограничение по памяти:	256 мегабайт

Дочь короля Флатландии собирается выйти за прекрасного принца. Принц хочет подарить принцессе сокровища, но он не уверен какие именно бриллианты из своей коллекции выбрать.

В коллекции принца n бриллиантов, каждый характеризуется весом w_i и стоимостью v_i. Принц хочет подарить наиболее дорогие бриллианты, однако король умен и не примет бриллиантов суммарного веса больше R. С другой стороны, принц будет считать себя жадным всю оставшуюся жизнь, если подарит бриллиантов суммарным весом меньше L.

Помогите принцу выбрать набор бриллиантов наибольшей суммарной стоимости, чтобы суммарный вес был в отрезке $[L, R]$.

Формат входных данных

Первая строка содержит число n ($1 \leqslant n \leqslant 32$), L и R ($0 \leqslant L \leqslant R \leqslant 10^{18}$). Следующие n строк описывают бриллианты и содержит по два числа — вес и стоимость соответствующего бриллианта ($1 \leqslant w_i, v_i \leqslant 10^{15}$).

Формат выходных данных

Первая строка вывода должна содержать k — количество бриллиантов, которые нужно подарить принцессе. Вторая строка должна содержать номера даримых бриллиантов.

Бриллианты нумеруются от 1 до n в порядке появление во входных данных.

Если составить подарок принцессе невозможно, то выведите 0 в первой строке вывода.

Примеры

dowry.in	dowry.out
3 6 8 3 10 7 3 8 2	1 2