Задача A. Рефрен

Имя входного файла:	`refrain.in`
Имя выходного файла:	`refrain.out`
Ограничение по времени:	3 секунды
Ограничение по памяти:	64 мегабайта

Рассмотрим последовательность n целых чисел от 1 до m. Подпоследовательность подряд идущих чисел называется *рефреном*, если произведение ее длины на количество вхождений в последовательность максимально.

По заданной последовательности требуется найти ее рефрен.

Формат входного файла

Первая строка входного файла содержит два целых числа: n и m ($1 \leqslant n \leqslant 150\,000$, $1 \leqslant m \leqslant 10$).

Вторая строка содержит n целых чисел от 1 до m.

Формат выходного файла

Первая строка выходного файла должна содержать произведение длины рефрена на количество ее вхождений. Вторая строка должна содержать длину рефрена. Третья строка должна содержать последовательность которая является рефреном.

Пример

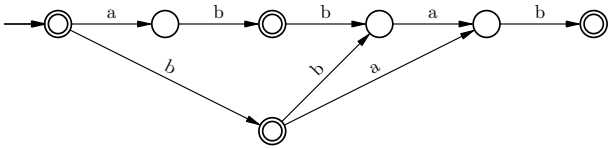
refrain.in	refrain.out
9 3 1 2 1 2 1 3 1 2 1	9 3 1 2 1

Задача B. Суффиксный автомат

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

Суффиксным автоматом для строки w называется детерминированный конечный автомат A, который допускает язык $Suff(w)$ — множество суффиксов слова w. Например, суффиксный автомат для слова *abbab* должен допускать в точности следующие слова: $\{abbab, bbab, bab, ab, b, \varepsilon\}$. Мы также потребуем, чтобы суффиксный автомат не имел недостижимых состояний, и не было состояний, из которых не достижимы допустимые. Других ограничений, например, минимальности, накладывать не будем.

На рисунке показан суффиксный автомат для слова *abbab*.



По заданному *скелету* суффиксного автомата некоторого слова требуется восстановить суффиксный автомат. А именно — вам даны состояния, переходы, начальное состояние и допускающие состояния. Но пометки на ребрах удалены.

Вам следует расставить пометки на ребрах заданного суффиксного автомата, так чтобы он стал суффиксным автоматом некоторого слова w, а также найти это слово. Для простоты будем считать, что размер алфавита ничем не ограничен, вы можете использовать в качестве символов числа от 1 до k (k вы можете выбрать сами).

Формат входного файла

Первая строка входного файла содержит три целых числа: n, m и t — количество состояний, количество переходов, и количество допускающих состояний, соответственно ($2 \leqslant n \leqslant 200$, $1 \leqslant m \leqslant 1000$, $1 \leqslant t \leqslant n$). Вторая строка содержит t целых чисел — номера допускающих состояний (состояния пронумерованы с 1, начальное состояние имеет номер 1).

Следующие m строк описывают переходы: каждая строка содержит два целых числа s_i и t_i и описывает переходы из s_i в t_i.

Формат выходного файла

На первой строке выходного файла выведите два целых числа: l и k — длину слова w и размер алфавита. Используйте числа $\{1, \ldots, k\}$ как элементы алфавита. k не должно превышать m.

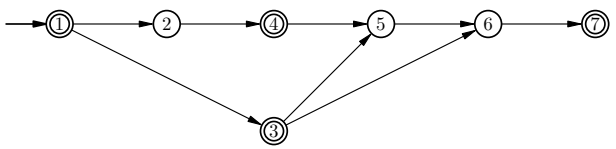
Вторая строка должна содержать l целых чисел — слово w.

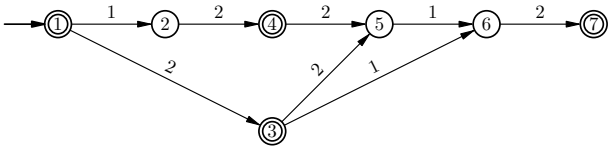
Наконец, третья строка должна содержать m целых чисел — метки на переходах скелета автомата, в том порядке, в котором они описаны во входном файле.

Гарантируется, что ответ всегда существует.

Пример

suffix.in	suffix.out
7 8 4 1 3 4 7 1 2 1 3 2 4 3 5 3 6 4 5 5 6 6 7	5 2 1 2 2 1 2 1 2 2 2 1 2 1 2



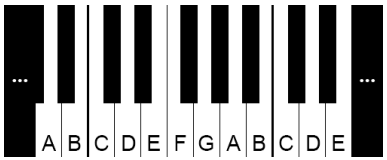


Задача C. Древняя мелодия

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

Археолог Вася нашел в недавно обнаруженном древнем саркофаге президента Атлантиды несколько листов с непонятными знаками. После внимательного рассмотрения обнаружилось, что эти знаки имеют строгие закономерности, и записывают мелодию для национального атлантского клавесина. Этот клавесин отличается от стандартных музыкальных инструментов тем, что на нем отсутствуют клавиши, которые на обычных современных инструментах имеют черный цвет.

Современные клавишные инструменты имеют такой вид:



Соотвественно, каждая клавиша (и черная, и белая) соответствует одному *полутону*. Древние Атланты использовали только белые клавиши, поэтому расстояние между некоторыми парами соседних клавиш было равно не одному полутону, а двум.

Способ записи мелодий в древней Атлантиде основывался на записи расстояний между звуками в полутонах. Каждый знак в мелодии записывает количество полутонов, необходимое для перехода от звука к следующему. При этом, естественно, что все клавиши должны присутствовать на национальном клавесине. Например, расстояние от звука *до* (`C`) до звука *фа* (`F`) равно 5, а обратное расстояние от `F` до `C` равно -5.

К сожалению, Вася столкнулся с несколькими проблемами при расшифровке мелодии. Во-первых, он не знает, с какой ноты мелодия могла бы начинаться. Во-вторых, неизвестен порядок листочков, и все ли они составляют одну мелодию.

В качестве первого эксперимента Вася решил составить мелодию из максимально возможного количества листочков. При этом должен быть способ проиграть всю мелодию, начиная с какого-то звука, и используя только клавиши национального клавесина (начальный звук тоже должен соответствовать клавиши на клавесине). Выбранные листочки можно расположить в произвольном порядке, но при этом нельзя использовать листочек более одного раза.

Ваша задача — написать программу, которая подберет оптимальный набор листочков для этого эксперимента.

Формат входного файла

В первой строке входного файла задается количество листочков N ($1 \leqslant N \leqslant 19$). В последующих N строках задается содержимое листочков. Каждая из этих строк содержит количество чисел на данном листочке $1 \leqslant k_i \leqslant 30$, а затем k_i чисел в пределах от $-10\,000$ до $10\,000$. Клавиатура древнего клавесина считается бесконечной и зацикленной.

Формат выходного файла

В первой строке выходного файла выведите максимальное количество листочков M, которое можно использовать для экспериментов, и идентификатор ноты, с которой следует начинать мелодию (латинскую букву от `A` до `G`). Во второй строке выведите M чисел, разделенных пробелами — номера листочков в порядке, в котором их следует проигрывать.

Пример

melody.in	melody.out
2 7 2 2 1 2 2 2 1 7 2 1 2 2 2 1 2	1 C 1
4 4 2 2 2 2 2 1 1 4 5 -4 12 -11 10 0 2 -2 5 -1 -4 2 -2 7 -2	1 C 4
3 5 2 3 9 1 7 5 -1 -7 -2 -3 -9 7 1 2 3 4 2 3 9	2 D 1 2