Проблема останова

Имя входного файла:	`стандартный ввод`
Имя выходного файла:	`стандартный вывод`
Ограничение по времени:	2 секунды
Ограничение по памяти:	512 мегабайт

Проблема останова — это одна из ключевых задач в теории вычислимости. Её формулировка такова: по данному тексту программы и входным данным необходимо определить, остановится ли программа на этих входных данных. В этой задаче вам предлагается решить её расширенную версию: по данному тексту программы определить, останавливается ли она на всех возможных входных данных.

Программа написана на простом языке, который может оперировать k регистрами, каждый из которых содержит 0 или 1. Регистры нумеруются с 0. Язык поддерживает следующие команды:

- `begin` — начало программы. Программа всегда начинается с этой команды.
- `and to from` — применить логическое «И» к значениям в регистрах с номерами `to` и `from` и сохранить полученное значение в регистр с номером `to`.
- `or to from` — применить логическое «ИЛИ» к значениям в регистрах с номерами `to` и `from` и сохранить полученное значение в регистр с номером `to`.
- `xor to from` — применить логическое «исключающее ИЛИ» к значениям в регистрах с номерами `to` и `from` и сохранить полученное значение в регистр с номером `to`.
- `move to from` — скопировать значение из регистра с номером `from` в регистр номер `to`. Значение регистра с номером `from` остаётся неизменным.
- `set to x` — присвоить регистру с номером `to` значение `x` ($x \in \{0, 1\}$).
- `not to` — применить логическое «НЕ» к значению в регистре с номером `to`.
- `jump reg line` — если в регистре с номером `reg` содержится 1, то перейти на строку программы с номером `line`, иначе перейти на следующую строку. Строки программы нумеруются с 0.
- `end` — конец программы. Программа всегда заканчивается этой командой.

Входные данные состоят из k битов и в начале исполнения лежат в регистрах. По данному тексту программы определите, останавливается ли она на всех входных данных, а если существуют такие входные данные, на которых программа будет работать вечно, выведите их.

Формат входных данных

В первой строке входных данных содержатся два числа n и k ($2 \leqslant n \leqslant 1000$, $1 \leqslant k \leqslant 12$) — количество строк в программе и количество регистров, соответственно.

В следующих n строках содержится текст программы в формате, указанном в условии. Гарантируется, что он корректен, то есть `begin` содержится только в первой строке, `end` только в последней, все номера регистров находятся в пределе от 0 до $k - 1$ включительно, второй аргумент команды `set` может быть равен только 0 или 1, а аргумент `line` команды `jump` находится в пределах от 0 до $n - 1$ включительно.

Формат выходных данных

В случае, если программа всегда останавливается, выведите `Yes`.

В противном случае, в первой строке выведите `No`. Во второй строке выведите последовательность из k чисел x_i ($x_i \in \{0, 1\}$) без пробелов, где x_i равняется значению i-го регистра во входных данных, на которых программа не прекратит работу.

Если входных данных, на которых программа будет работать вечно, несколько, выведите любой пример.

Примеры

стандартный ввод	стандартный вывод
3 6 begin not 0 end	Yes
8 3 begin set 1 1 xor 0 1 xor 0 1 or 0 2 and 0 1 jump 0 6 end	No 100
5 6 begin move 2 1 move 3 2 jump 3 1 end	No 010000
7 6 begin set 0 1 or 1 0 jump 1 6 set 0 1 jump 0 4 end	Yes

Замечание

В первом тестовом примере в программе нет ни одной команды `jump`, поэтому она завершится на любых входных данных.

Разберём работу программы из второго тестового примера на двух входных данных: `000` и `100`.

Для `000`:

1. Присваиваем регистру номер 1 значение 1. Значения регистров: `010`
2. Делаем логическое «исключающее ИЛИ» с регистрами номер 0 и 1, результат записываем в регистр 0. Значения регистров: `110`
3. Делаем еще раз то же самое, значения регистров: `010`
4. Делаем логическое «ИЛИ» с регистрами номер 0 и 2, результат записываем в регистр 0. Значения регистров: `010`
5. Делаем логическое «И» с регистрами номер 0 и 1, результат записываем в регистр 0. Значения регистров: `010`
6. Так как в регистре номер 0 хранится 0, не переходим на строку номер 6 (то есть на текущую), переходим на следующую. Значения регистров: `010`
7. Дошли до последней строки с командой `end`, программа завершилась.

Для `100`:

1. Присваиваем регистру номер 1 значение 1. Значения регистров: `110`
2. Делаем логическое «исключающее ИЛИ» с регистрами номер 0 и 1, результат записываем в регистр 0. Значения регистров: `010`
3. Делаем еще раз то же самое, значения регистров: `110`
4. Делаем логическое «ИЛИ» с регистрами номер 0 и 2, результат записываем в регистр 0. Значения регистров: `110`
5. Делаем логическое «И» с регистрами номер 0 и 1, результат записываем в регистр 0. Значения регистров: `110`
6. Так как в регистре номер 0 хранится 1, переходим на строку номер 6 (то есть на текущую). После этого нам нужно будет снова выполнить эту строку, и таким образом мы вошли в бесконечный цикл.

В третьем тестовом примере проиллюстрирована работа `move`.

В четвёртом тестовом примере в тексте программы есть бесконечный цикл, но он недостижим, а программа останавливается на всех входных данных.

Система оценки

Тесты к этой задаче состоят из четырёх групп. Баллы за каждую группу ставятся только при прохождении всех тестов группы и всех тестов некоторых из предыдущих групп. Обратите внимание, прохождение тестов из условия не требуется для некоторых групп.

Подзадача	Баллы	Доп. ограничения	Необходимые подзадачи	Комментарий
		k		
0	0	–	–	Тесты из условия.
1	18	–	–	Помимо `begin` и `end` в программе содержатся только команды `jump`.
2	17	$k = 1$	–	
3	28	$k \leqslant 6$	0, 2	
4	37	–	0–3	