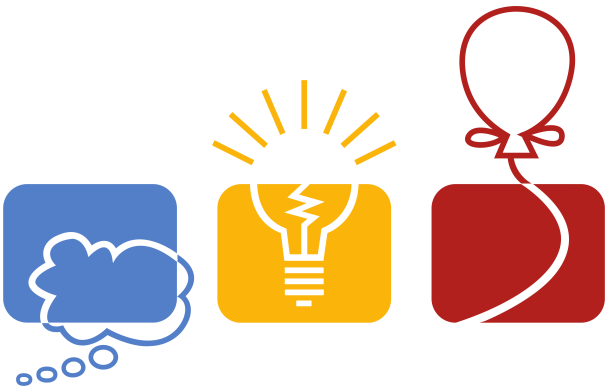
United Kingdom and Ireland Programming Contest 2023



Problems

- **A** Assessment Disruption
- **B** Boat Commuter
- **C** Clearing Space
- **D** Delivery Forces
- **E** Enchanted Fortress
- **F** Fast Forward
- **G** Glacier Travel
- **H** History in Numbers
- **I** International Travel
- **J** Journey of Recovery
- **K** Kernel Scheduler
- **L** Last One Standing
- **M** Mini-Tetris 3023
- **N** Naming Wine Bottles

Problems are not ordered by difficulty.
Do not open before the contest has started.

This page is intentionally left (almost) blank.

Problem A
Assessment Disruption

It is finally time to submit your essays on economics for assessment!

Each essay is characterised by its word count w and its quality q. The required word count is W, so the closer w to W is in your essay, the better mark you may expect. And, surely, the higher the quality, the better. However, one essay can have a better quality and a bigger deviation from W than another, so it is not clear which one is better.

As an economics student, you may know that this kind of situation is captured by *Pareto dominance*. Formally, essay A is said to *dominate* essay B *in the Pareto sense* if $|w_A - W| \leq |w_B - W|$, $q_A \geq q_B$, and at least one of these inequalities is strict.

The professor is known to use this relation to mark the essays. First, she finds all the best essays: those that are not dominated by any other essay. These essays receive the same mark, which is the highest among this year's students (but still can fall below their expectations!). Then she removes the marked essays and repeats the procedure, but the mark will be lower this time, and so on. More precisely, she uses the following algorithm:

- All the essays are numbered from 1 to N.
- Each essay can be either *in work*, *postponed*, or *marked*. Initially, all essays are *in work*.
- A variable r captures the *rank* of an essay, the lower, the better. Initially, $r \leftarrow 1$.
- While there are any essays that are *in work*:
  - Iterate over all essay numbers from 1 to N. If the i-th essay is *in work*:
    - Iterate over all essay numbers from $i + 1$ to N. If the j-th essay is *in work*, **compare** essays i and j for dominance:
      - If i dominates j, turn j into *postponed*.
      - If j dominates i, turn i into *postponed* and break the loop.
    - If the i-th essay is still *in work*, it receives rank r and turns into *marked*.
  - All *postponed* essays become *in work* again.
  - The rank is increased: $r \leftarrow r + 1$.

You are afraid that you, and everyone else, will get low marks, but someone told you that if it took the professor too long to perform the entire assessment, the department would take it over, and the final marks would be based on a simple multiple-choice quiz. By rigorous computations you found out that the number of essay **comparisons** should be at least $N^3/20$ for this to happen. Find the way to disrupt the assessment procedure.

Input

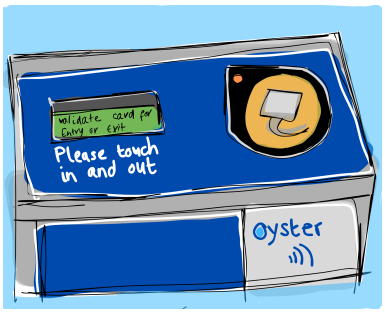
The first and only line of the input file contains two numbers, N ($2 \leq N \leq 10^3$) and W ($0 \leq W \leq 10^4$), separated by a whitespace.

Output

Output N lines. The i-th line, $1 \leq i \leq N$, should contain the word count w_i and the quality q_i of the i-th essay. They should be integers that satisfy $0 \leq w_i \leq 10^4$ and $0 \leq q_i \leq 10^3$. No two essays should be described by the same pair of numbers, because this counts as collusion, which you would want to avoid at all costs.

Sample Input 1	Sample Output 1
3 2500	2500 528 2480 543 2520 511

Problem B
Boat Commuter



The Bulgarian city of Nodnol runs a boat service to ferry its residents between the trendy areas in which they live and the large metallic structures in which they work on the next recession.

TFN (Transport For Nodnol) has issued m travel cards (known affectionally as "Retsyo"), which are numbered from 1 to m. Each pier has a card terminal at which passengers are required to tap "in" when starting the trip and to tap "out" when finishing it.

As there is only one card terminal on each pier, passengers use the same device to tap in and to tap out.

Trip cost depends on the distance travelled and is determined as follows:

- if the trip started at the pier i and finished at the pier j ($i \neq j$), then its cost is $|i - j|$ pounds;
- if the trip started somewhere and was not finished with a tap out, then it costs £100;
- if the trip started and finished in the same place, then it also costs £100, as it is interpreted as an attempt to game the system.

You are given a sequence of tapping events — for each you have the pier p_i and card number c_i recorded. You are to determine how much the transport authority should charge each of the cards

Input

- One line containing three integer numbers: the number of piers n, the number of travel cards m, and the number of events k ($2 \leq n \leq 50$, $1 \leq m, k \leq 10^5$).
- k further lines, each describing tap events in chronological order.
  - The i-th event is described by two integers p_i and c_i ($1 \leq p_i \leq n$, $1 \leq c_i \leq m$).

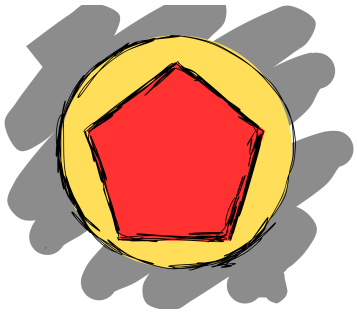
Output

Output m integers separated by spaces — the i-th integer giving the total charge to be applied to the i-th card.

Sample Input 1	Sample Output 1
3 3 5 1 1 1 2 1 2 3 1 2 3	2 100 100

This page is intentionally left (almost) blank.

Problem C
Clearing Space



You are putting up an event space in Nottingham's Sherwood Forest by erecting a fence in a circular-shaped clearing you found that is exactly one kilometre in radius. You will put some fence posts in the trees around the edge of the clearing and then connect them together with fencing later.

You would like to put the fence around as much of the event space as possible. However, the ground is only suitable in a few places around the border, and you only have so many fence posts to put in the ground, so you'll have to choose carefully if you want to maximise area.

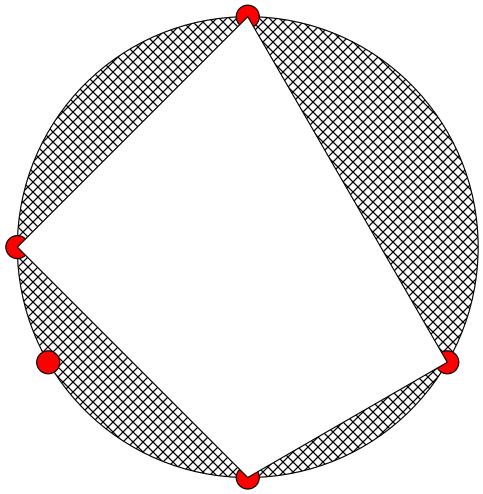


Figure C.1: An illustration of using 4 posts to capture the maximum area in sample input 1.

Knowing the safe places to put fence posts, and the number of posts you have, what is the maximum area of clearing you can enclose?

Input

- One line containing the integer number of safe points around the 1km-radius clearing, n ($3 \leq n \leq 100$).
- One line containing the integer number of fence posts you have, p ($3 \leq p \leq n$).
- One line containing n distinct real numbers $a_1, \ldots, a_n$ in ascending order, the angles in degrees of each of the safe places to add fence posts ($0 \leq a_i < 360$).

Output

Output the maximum area you can capture with a polygonal clearing made using at most p fence posts, in square metres.

The output must be accurate to an absolute or relative error of 10^-6.

As a reminder, the radius of the clearing is 1km.

Sample Input 1	Sample Output 1
5 4 0 120 180 240 270	1866025.40378443866

Problem D
Delivery Forces

Gry finally becomes the Executive Courier Officer in "Universe Express". He has n subordinate couriers with some delivery strength f_i. The delivery strength of a team of three people is the median of their strength, i.e., the middle element after the sorting. Please help Gry to split the couriers into k teams of three people in order to maximize the total delivery strength of "Universe Express". The total strength is the sum of the strength of these k teams.

Input

- One line containing the number of couriers in the company, n ($1 \leq n \leq 10^6$), where n is a multiple of 3.
- One line containing the strengths of the n couriers $f_1 \ldots f_n$ ($1 \leq f \leq 10^6$).

Output

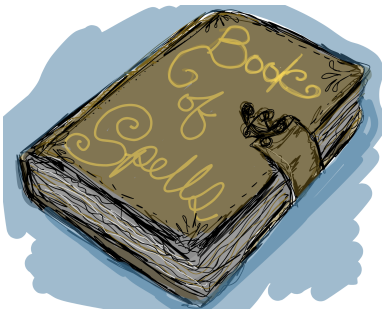
The sole line of the output should contain the maximal strength of "Universe Express".

Sample Input 1	Sample Output 1
3 1 2 3	2

Sample Input 2	Sample Output 2
6 5 6 2 3 1 4	8

This page is intentionally left (almost) blank.

Problem E
Enchanted Fortress



Elisabeth the Efficient, a famous magician, was called by Emmanuel the Empowered, the mighty lord of the East Embarkmentlands, to enhance the magical protection of his enchanted fortress.

After arriving at the site and studying the structure of the fortress and the existing spells that protect its integrity, Elisabeth found out the following properties of the spell to design:

- it should consist of symbols taken from the string s;
- all symbols should be unique;
- the strength of the spell depends only on which symbols are chosen, but not on their order;
- if symbols s_i and s_j, $i \le j$, both exist in the spell, then its strength increases by $d_{i,j}$.

For instance, if $s =$ `ABC` and

$$d = \begin{pmatrix} 1 & -1 & 2 \\ - & 2 & -3 \\ - & - & 1 \end{pmatrix}$$

then a spell `A` has strength 1, `ABC` has strength 2, and `AC` has strength 4.

However, the problem appeared to be too difficult to Elisabeth, because she has only little experience of working with computers. Can you help her to find the strongest spell?

Input

The first line of the input file contains the string s, that will not be empty and may contain only big Latin letters or symbols `!`, `?`, `@`, `*`. All symbols will be pairwise different. Its length $n = |s|$ will thus not exceed 30.

The following n lines contain the description of the matrix d. The i-th of these lines ($1 \le i \le n$) contains $n + 1 - i$ integer numbers $d_{i,i}$, $d_{i,i+1}$, …, $d_{i,n}$, separated by single whitespace symbols. These numbers do not exceed 10^6 by the absolute values.

Output

Output the length of the strongest spell in the first line. In the second line, output the spell itself. If there are multiple equally strongest spells, output any of them.

Sample Input 1	Sample Output 1
ABC 1 -1 2 2 -3 1	2 AC

Sample Input 2

```
@
-1
```

Sample Output 2

```
0
```

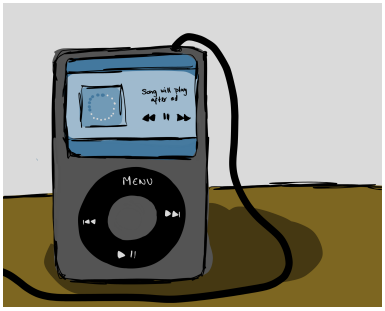
Sample Input 3

```
ABDFHORSU!?
1 -1 1 1 1 1 1 1 1 -1
-1 -1 -1 -1 -1 -1 -1 -1 -1 -1
1 1 1 1 1 1 1 1 -1
1 1 1 1 1 1 1 -1
1 1 1 1 1 1 -1
1 1 1 1 1 -1
1 1 1 1 -1
1 1 1 -1
1 1 -1
1 -1
-1
```

Sample Output 3

```
9
FUSRODAH!
```

Problem F
Fast Forward



Gry has started to use the new Expify song streaming platform. Since, Gry does not want to spend money Expify forces him to listen to advertisements. An advertisement can be played only after some song (it cannot be played in the middle) and only if the time from the end of the previous advertisement is at least c seconds.

Gry has a circular playlist with n songs where the duration of the i-th song is d_i seconds. He wants to minimize the number of advertisements, so, he wants to find out how many advertisements will be if he starts listening to his whole playlist from i-th song, i.e., the circular playlist stops playing after n songs.

We suppose that there is an advertisement right before Gry starts listening. Neither this advertisement nor the one, after the playlist stops, count.

Input

- One line containing the number of songs in the playlist n, and the refresh time between advertisements c ($1 \leq n \leq 10^6$, $1 \leq c \leq 10^9$)
- One line containing the n durations of the songs $d_1 \ldots d_n$ ($1 \leq d_i \leq 10^3$)

Output

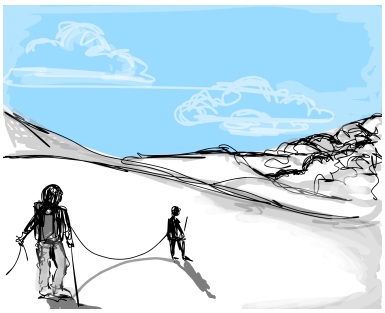
Output the number of advertisements if Gry starts listening to the playlist from the i-th song.

Sample Input 1	Sample Output 1
7 7 1 1 1 1 1 1 1	0 0 0 0 0 0 0

Sample Input 2	Sample Output 2
3 3 1 1 3	0 1 1

This page is intentionally left (almost) blank.

Problem G
Glacier Travel



Glaciers are vast rivers of slowly-flowing ice, fraught with crevasses which hide under thin layers of snow and for unsuspecting walkers to step into and fall in. To reduce the danger, hikers usually go in teams tied together with a thick rope to reduce the consequences of a fall–if one person falls in, the other person may yet hold them from a safe distance.

Today, you are roped up to cross a glacier with your partner. Your plan is to follow the exact same route, at the same speed, the first starting earlier and the second beginning to trace steps once you are exactly x metres apart. Were you to follow a completely straight path, you would thus then remain exactly x metres apart at all time.

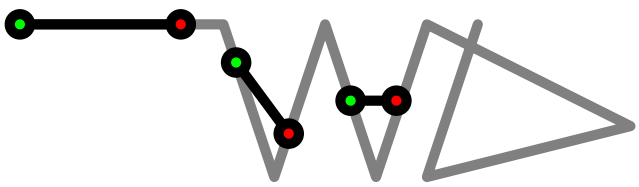


Figure G.1: An illustration of the path taken in the 2nd sample case, taken from above. This could also be a particularly festive diagram of someone falling into a crevasse.

However, the twisting nature of the course as you avoid obstacles means that you may not always remain exactly x metres apart. What is the closest that you shall actually come while both of you are walking on the path?

Input

- One line containing a real number: the separation distance along the path in metres, s ($1 \leq s \leq 1000$).
- One line containing the number of points in the path, n ($2 \leq n \leq 10^6$).
- n further lines, the ith of which contains a pair of integers giving the ith coordinate on the track $x_i y_i$ ($-10^6 \leq x, y \leq 10^6$) in metres from the origin.

Every pair of adjacent points on the track are distinct from one another, although the track may cross over or repeat itself. The track is guaranteed to have a length of at least s.

Output

Output the minimum distance between the two walkers at any point on the route, ignoring any time after the first walker has finished, or before the second walker has started.

The output must be accurate to an absolute or relative error of at most 10^{-4}.

Sample Input 1

```
5
4
20 0
10 0
10 10
0 10
```

Sample Output 1

```
3.5355339
```

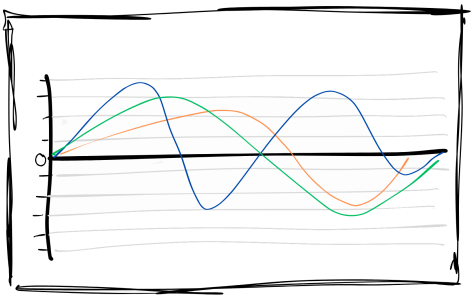
Sample Input 2

```
3.16227766
9
-2 4
2 4
3 1
4 4
5 1
6 4
10 2
6 1
7 4
```

Sample Output 2

```
1
```

Problem H
History in Numbers



Algorithms and data structures find their place in various arts, crafts, and sciences. In this problem, we are discussing one potential application in history.

Historians are studying the urban development of a city over a period of n years. The urban development is quantified using an aggregate measure they call an urban development index (UDI), which could take integer values, including negative. There is an initial estimation e_i of this index for each of the years. However, during the research process, due to the new documents and evidence discovered, these estimates are revised and updated. More formally, throughout the historians' work process, they could want to update the UDI estimates for all years between s_j and f_j (inclusive) by adding d_j to each of them.

The main question historians are interested in is if the UDI has been increasing over a certain period of time. However, due to the noisy nature of estimates, we will not be using the standard definition. Instead, the following procedure is used:

- we collapse all consequent equal numbers into one. For instance, `1 1 2 2 2 3 3 3` becomes `1 2 3`;
- the UDI is considered *increasing* from year i to year j if the sequence of local minima in the UDI sequence after the above transformation is strictly increasing. An element p_i is considered to be a local minimum if it is less than both of its neighbors (one neighbor for the first or last element).

You are to implement the system able to store the UDI estimates and process the requests to update them and to check if the UDI has been increasing over a certain period.

Input

- One line containining an integer n ($1 \le n \le 3 \cdot 10^5$) denoting the length of the time period being considered.
- One line containing n space separated integers e_i ($-10^8 \le e_i \le 10^8$).
- One line containing an integer number m ($1 \le m \le 3 \cdot 10^5$).
- m further lines each describing one request in one of the following two formats:
  - `update` followed by three integers s_j, f_j, and d_j ($1 \le s_j \le f_j \le n$, $|d_j| \le 10^8$).
  - `check` followed by two integers s and f ($1 \le s \le f \le n$).

Output

For each `check` request output `YES` if the UDI has been increasing during the corresponding period and `NO` otherwise.

Sample Input 1

```
5
10 4 10 6 10
5
check 1 5
update 2 3 1
check 1 5
update 2 3 1
check 1 5
```

Sample Output 1

```
YES
YES
NO
```

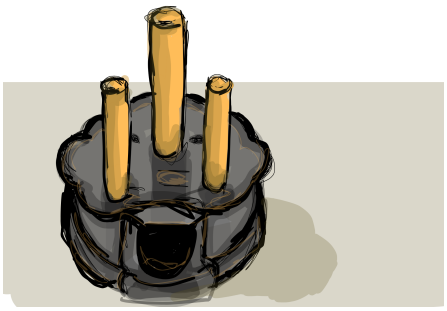
Sample Input 2

```
8
10 -5 -5 -5 11 6 6 12
1
check 1 8
```

Sample Output 2

```
YES
```

Problem I
International Travel



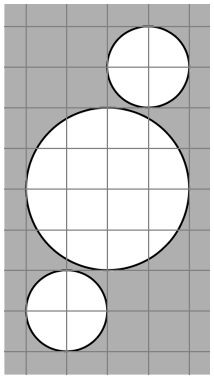
When you arrive into a new country, one of the first questions is “can I charge my phone here?”. Nowadays, the question of voltage is often secondary, as modern IT equipment is mostly voltage insensitive. The shapes of outlets and plugs, however, are still a problem.

In this problem we give you descriptions of electrical plugs and sockets, and you must determine whether one can insert such a plug into such a socket. In our simplified world, each plug pin is a cylinder, and each socket hole is cylindrical too. Since sockets have springs to ensure a tight contact, the diameter of a pin may be smaller than the diameter of the matching hole. We also neglect pin lengths and hole depths, so both plug and socket are described by three circles.

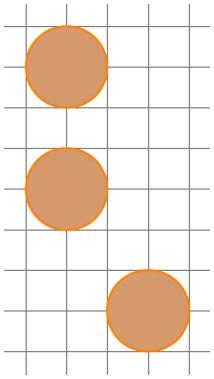
In both, the plug and the socket, one of these three circles corresponds to the earth wire, so they have to match. The other circles represent different power phases, and since most countries use alternating current (AC), the corresponding plug pins are allowed to match the socket holes in either way.

Given the descriptions of a plug and a socket, is it possible to insert this plug into this socket?

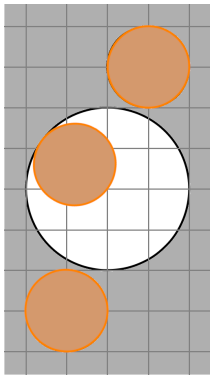
An illustration for the second sample test comes below.



The socket



The plug



How to insert the plug

Input

The input file consists of six lines. The first three lines describe the plug, the next three lines describe the socket. The i-th line contains three positive integers x_i, y_i, r_i, defining the coordinates of the center of the pin or the hole and its radius. All these values are integers that do not exceed 10^4.

In both the plug and the socket, no two circles intersect or touch. The first pin or hole corresponds to the earth wire.

Output

If it is impossible to insert the plug into the socket, output `NO`.

Otherwise, output `YES` in the first line. In the next three lines, print the new coordinates of centres of the plugs after the necessary translations and/or rotations, which would allow the plug to be inserted into the socket. Your answer is accepted if the following all hold:

- the pairwise distances between the plug pins in the input file match those in the output file with an absolute precision of 10^{-6};
- one can indeed obtain the configuration in the output file by translating and/or rotating the plug described in the input file;
- if the plug pin radius needs to be reduced by at most 10^{-6} to completely fit the hole.

If the answer is `NO`, it is guaranteed that even all the plug pins become smaller by $2 \cdot 10^{-6}$, it is still impossible to fit the plug into the socket.

Sample Input 1

```
1 1 1
4 1 1
1 4 1
1 1 1
1 4 1
4 1 1
```

Sample Output 1

```
YES
1 1
4 1
1 4
```

Sample Input 2

```
1 4 1
3 1 1
1 7 1
2 4 2
1 1 1
3 7 1
```

Sample Output 2

```
YES
1.2 4.6
1 1
3 7
```

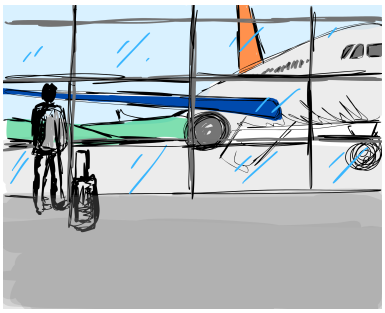
Sample Input 3

```
1 4 1
3 1 1
1 7 1
1 1 1
1 4 1
4 1 1
```

Sample Output 3

```
NO
```

Problem J
Journey of Recovery



You are making an international trip with several stops to blow off steam and celebrate your progression onto the NWERC. Since your flights are often booked with low-cost airlines, you always run the risk of your flights being cancelled last minute leaving you stuck in the airport. Normally this is no problem—take the next flight—but you have to arrive at the NWERC on time.

If any one of your flights is cancelled at the same moment you are about to depart, and all others operate as planned, you will book a new itinerary from there to your final destination. Assuming you always plot the fastest route, by how much will you be delayed in the worst case?

Input

- One line containing the number of flight connections overall, n ($1 \leq n \leq 10^6$).
- n further lines, the ith of which contains four space-separated fields:
  - The code of the departure airport, s_i ($1 \leq |s| \leq 20$)
  - The time of departure in days, minutes, and hours, in the format **dd`hh:mm`** ($1 \leq d \leq 365$, $0 \leq hh \leq 23$, $0 \leq mm \leq 59$).
  - The code of the arrival airport, t_i ($1 \leq |s| \leq 20$)
  - The time of arrival in days, minutes, and hours, in the format **dd`hh:mm`** ($1 \leq d \leq 365$, $0 \leq hh \leq 23$, $0 \leq mm \leq 59$).
- One line containing the number of flight connections in your itinerary, m ($1 \leq m \leq n$).
- One line containing the m indices $f_1 \ldots f_m$ of flight connections, in the order you plan to take them.

Flights always go between different airports and always strictly forward in time. For every consecutive pair u, v in your itinerary, the arrival time of flight u is guaranteed to be less than or equal to the departure time of flight v.

Transfers are instantaneous—that is to say, arriving at an airport and departing from it in the same minute is possible. Likewise, if one planned flight is cancelled, you may board another departing at exactly the same time.

Output

Output the maximum amount by which you could be delayed if any one of the given flights is cancelled at its moment of boarding. If you would not be delayed at all in any case (or can even arrive early) simply output 0.

If you cannot always make it to the destination at all, output `stranded` instead.

Sample Input 1

```
8
egnx 0d00:10 delft 0d01:00
delft 0d01:00 zad 0d09:00
zad 0d09:01 prg 0d15:30
prg 0d20:00 delft 1d02:15
prg 0d22:00 delft 1d04:15
zad 2d00:00 delft 3d00:00
egnx 2d00:00 delft 2d02:00
egnx 2d00:00 delft 2d02:00
4
1 2 3 4
```

Sample Output 1

```
2745
```

Sample Input 2

```
3
ork 101d00:00 noc 101d00:01
ork 100d23:59 noc 101d00:02
dub 100d00:00 ork 101d00:00
2
3 1
```

Sample Output 2

```
stranded
```

Sample Input 3

```
2
lax 0d00:30 hnl 0d06:20
lax 0d00:30 hnl 0d06:20
1
2
```

Sample Output 3

```
0
```

Problem K
Kernel Scheduler

You are developing the scheduling module for the new operating system. This module takes n tasks to be executed and the dependencies between them and then puts them in a certain order for execution.

More formally, there are n tasks numbered from 1 to n. You are also given m dependencies numbered from 1 to m; i-th of them is described by two numbers — a_i and b_i, meaning that the task a_i should be executed before the task b_i.

In some cases, there are *cyclical dependencies* — situations when according to the dependencies given some task t_1 should be executed before t_2, t_2 before t_3, ..., and t_{k-1} before t_k and t_k before t_1. Cyclical dependencies create a problem for scheduling, so you decided to remove some of the given dependencies in such a way that the resulting set does not contain any cyclical ones.

However, you still need to keep at least $m/2$ original dependencies to preserve some of the original information. You are to write the program performing this task.

Input

- One line containing the numbers n and m ($2 \leq n \leq 10^5$, $1 \leq m \leq 3 \cdot 10^5$).
- m further lines, each containing two numbers a_i and b_i ($1 \leq a_i, b_i \leq n$, $a_i \neq b_i$), describing the corresponding dependency between two tasks a_i and b_i.

Output

The first line should should contain `YES` in case the desired subset of dependencies exists, and `NO` otherwise.

In the `YES` case second line should contain the number k of the selected dependencies (please note that k should be at least $m/2$) and the third line should contain k numbers — the ids of the selected dependencies. They are numbered from 1 to m in the order given in the input.

Sample Input 1	Sample Output 1
3 3 1 2 2 3 3 1	YES 2 1 2

Sample Input 2

```
2 5
1 2
1 2
1 2
2 1
2 1
```

Sample Output 2

```
YES
3
1 2 3
```

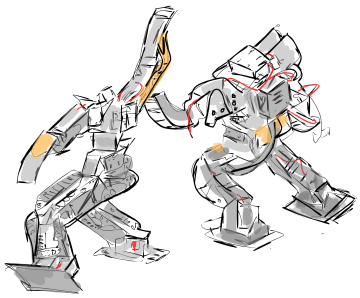
Sample Input 3

```
4 4
1 2
2 3
2 4
3 4
```

Sample Output 3

```
YES
4
1 2 3 4
```

Problem L
Last One Standing



In a computer game units are described by their health h, damage d, and time to reload t.

When such a unit fires a missile at an opposing one — the opponent's health is decreased by d 0.5 seconds after the missile is fired. The time between consecutive missile launches for the same unit should be at least t seconds.

For simplicity, we assume the missile supply to be infinite for all units in the game.

Two players — one controlling a unit with health h_1, damage d_1 and time to reload t_1, and the second with a unit described by h_2, d_2 and t_2 — have engaged in a fight in this computer game. Both units are fully reloaded at the beginning of the fight and can fire missiles immediately.

The unit is destroyed when its health becomes zero or negative. A player wins if there is a moment in time such that the opponent's unit is destroyed, while theirs is not.

Since it takes 0.5 seconds for a missile to reach its target, it is possible for both units to fire missiles at the same time and ultimately destroy each other.

You are to determine who wins in case both players act optimally.

Input

- One line containing the integer numbers h_1, d_1 and t_1 ($1 \leq h_1, d_1, t_1 \leq 1000$).
- One line containing the integer numbers h_2, d_2 and t_2 ($1 \leq h_2, d_2, t_2 \leq 1000$).

Output

Output the phrase `player one` if the first player wins, `player two` if the second player wins, or `draw` if neither player wins.

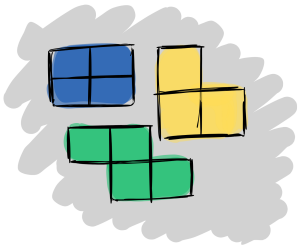
Sample Input 1	Sample Output 1
30 10 10 30 15 19	player two

Sample Input 2	Sample Output 2
30 15 19 30 10 10	player one

Sample Input 3	Sample Output 3
100 20 10 100 12 5	draw

This page is intentionally left (almost) blank.

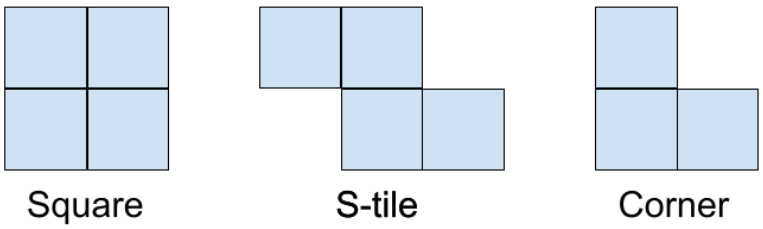
Problem M
Mini-Tetris 3023



A guy named Gry found a new game called "Mini-Tetris 3023". This small version of Tetris is played on a very long grid only 2 cells high and has just three types of tile:

- A `square` made out of 4 tiles in a 2×2 grid.
- An `S-tile` made out of 4 tiles, 2 on one row and 2 slightly offset on the other
- A `corner` made out of 3 tiles, 1 on one row and 2 on the other

Tiles may be rotated 0, 90, 180, or 270 degrees to fit amongst each other, however, they cannot overlap or go outside the vertical boundary of the grid.



This game provides a squares, b S-tiles, and c corners. Gry would like to beat the high score by creating the largest-possible contiguous $2 \times n$ rectangle out of some or all of the provided tiles, without any tiles overlapping or sticking out of the rectangle.

Input

- The sole line of input contains three integers a, b, and c ($0 \leq a, b, c \leq 50$) — the number of squares, S-tiles, and corners, respectively.

Output

Output the maximum possible width of the grid, n, that can be perfectly filled by some or all of the given tiles without overlapping or overstepping the boundaries.

Sample Input 1	Sample Output 1
2 2 2	11

Sample Input 2	Sample Output 2
1 1 1	2

Sample Input 3	Sample Output 3
0 0 0	0

This page is intentionally left (almost) blank.

Problem N
Naming Wine Bottles

Wine is a sophisticated business. Bottling alone encompasses the art of label design, the physics of vacuum sealing, the craft of glassforming, and the calculus of volumetric shapes.

Today we will deal with literature: every size of wine bottle has a name. You may know that a "Standard" bottle holds 0.75L. Did you also know that a 15L bottle is a "Nebuchadnezzar", and a 12L is a mighty "Balthazar"?

Usually wine bottle sizes come in multiples of 1.5L. You have some other bottles in non-standard sizes and you will need to create impressive names for them as well. Note that any two bottles of the same size must have consistent names.

Input

- On the first line, the number of bottles n ($1 \leq n \leq 10\,000$)
- On each of the following n lines, the volume of a bottle v ($0.0 \leq v_i \leq 10^4$) with up to 10 digits after the decimal point, followed by an uppercase letter "L"

Output

For each given line, output one lower-case word, with Latin letters only, giving a name for this size of bottle.

Sample Input 1	Sample Output 1
6 15L 0.88L 1.0L 1L 1000L 1024L	nebuchadnezzar hammurabi standard standard giganebriator gibinebriator

Sample Input 2	Sample Output 2
3 0.03L 0.031L 0.03L	small smallish small

This page is intentionally left (almost) blank.