TWGSS
OI Team

2022/23 Team Formation Test

INFORMATION

Date	15 October 2022
Time	14:10 - 17:40 (3 hours 30 minutes)
Full Score	400

PROBLEMS

Problem		Limit		Score	
		Time	Memory	Subtasks	Total
TF231	School Corridor	1s	256MB	7 + 14 + 9	30
TF232	Ridiculous Price	1s	256MB	11 + 14 + 5	30
TF233	Seating Plan	1s	256MB	2 + 5 + 8 + 15 + 10	40
TF234	Chemcraft Time	1s	256MB	7 + 13 + 24 + 6	50
TF235	Problem Descriptor	1s	256MB	8 + 9 + 8 + 17 + 28	70
TF236	King's Army	1s	256MB	13 + 7 + 12 + 8 + 20 + 17 + 12 + 1	90
TF237	Blackboard Escape	1s	256MB	2 + 3 + 4 + 5 + 6 + 20 + 50	90

REMINDERS

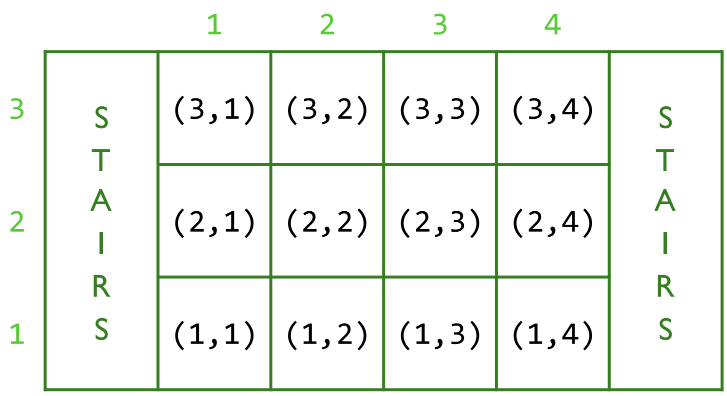
- All problems are divided into subtasks. All test cases in a subtask must be passed to get points.
- You may attempt the problems using C++20 or Python 3 (secondary language). It is not guaranteed that the problems are solvable using secondary language.
- Unless otherwise specified, your output will be automatically fixed as follows:
  - Trailing spaces in each line will be removed.
  - An end-of-line character will be added to the end of the output if not present.
- 64-bit integers may be required in some problems, use `long long` in C++ if needed.
- All stories are hypothetically written. In case of coincidences, they are coincidences.

TF231 School Corridor

Score	30
Time Limit	1s
Memory Limit	256MB

Today is Thursday, Daniel is writing a Chinese essay! As this takes up 1 hour and 30 minutes while the length of two Chinese lessons is only 1 hour and 20 minutes, 10 minutes would be taken from his recess, leaving only 5 minutes for him to move between classrooms (and go to toilet). In order to be punctual and avoid being scolded by Mr. Yee for being late, he needs to rush to the Biology Laboratory immediately after finishing his composition.

As an S6 student, he is familiar with the floor plan of the school. The school consists of H floors. Each floor have W classrooms. **TWO** stairs are located at the two ends of each floor, one before the leftmost classroom and one after the rightmost classroom. Both stairs have access to all floors. Moving up or down 1 floor by the stairs or to the adjacent stair or classroom takes 1 second. The figure below shows the school with $H = 3$, $W = 4$.



Given the classroom positions where Daniel writes his Chinese essay and the Biology Laboratory respectively, please help him to find the minimum possible time needed to rush to the Biology Laboratory.

INPUT

The first line consists of two space-separated integers H and W.

The second line consists of two space-separated integers h_c and w_c, meaning the classroom where Daniel writes his Chinese essay is located on the h_c-th floor and is the w_c-th classroom counting from the left on that floor.

The third line consists of two space-separated integers h_b and w_b, meaning the Biology Laboratory is located on the h_b-th floor and is the w_b-th classroom counting from the left on that floor.

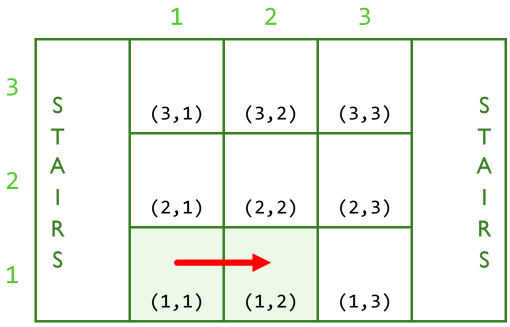
OUTPUT

Output an integer, the minimum possible time (in seconds) for Daniel to move to the Biology Laboratory.

SAMPLE TESTS

	Input	Output
1	3 3 1 1 1 2	1

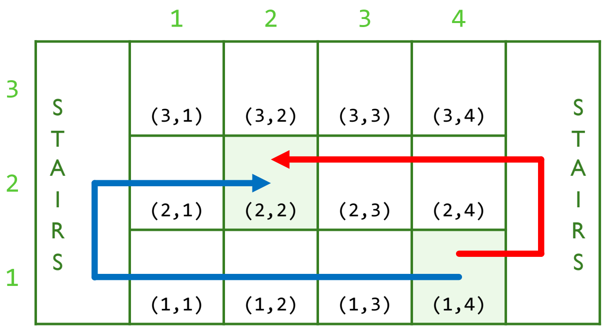
Daniel could just move to the adjacent classroom, so the minimum possible time is 1 second.



	Input	Output
2	3 4 1 4 2 2	5

- If Daniel used the left stairs, the time taken would be $4 + 1 + 2 = 7$.
- If Daniel used the right stairs, the time taken would be $1 + 1 + 3 = 5$.

Therefore, the minimum possible time is 5 seconds.



CONSTRAINTS

$1 \leq H, W \leq 2005$

$1 \leq h_c, h_b \leq H$

$1 \leq w_c, w_b \leq W$

Daniel never writes his Chinese essay at the Biology Laboratory.

SUBTASKS

	Points	Constraints
1	7	$H = 1$
2	14	$W = 2$
3	9	No additional constraints

TF232 Ridiculous Price

Score	30
Time Limit	1s
Memory Limit	256MB

After the Biology lesson, Daniel moves downstairs to the tuck shop. As he needs to revise for his Chemistry Uniform Test, he decides to have a bag of *lomien* (撈麵) for his lunch! Standing in the queue, he discovers something weird: 1 bag of *lomien* costs $8, whereas 2 bags of *lomien* cost $20.

"Why would anyone buy 2 bags of *lomien* then?" doubts Daniel. "The pricing is so **ridiculous**!" exclaims Daniel. (Sadly, he doesn't realise the fact that he can buy *lomien* for a girl he loved at the back of the queue so that she won't miss her lunch. Anyways.)

Hypothetically, there is a situation where the tuck shop sells at most N bags of *lomien* at a time, where k bag(s) of *lomien* is sold for $\$a_k$ $(1 \leq k \leq N)$.

The unit price when buying k bags of *lomien* is defined to be $\$\frac{a_k}{k}$. **A pricing is *normal* if the unit prices are non-increasing.** That is, for all integers $1 \leq x < y \leq N$, the unit price of x bags of *lomien* is lower or equal to the unit price of y bags of *lomien*. If any pair of (x, y) violates the above condition, then the pricing is considered as ***ridiculous***.

As a friend of Daniel, you decided to help him to determine whether a *lomien* pricing where is ***normal*** or ***ridiculous***.

INPUT

The first line consists of an integer N.

The second line consists of N space-separated integers $a_1, a_2, \dots, a_N$.

OUTPUT

Output either `Ridiculous` or `Normal` on the first and only line.

SAMPLE TESTS

	Input	Output
1	2 8 20	Ridiculous

- The unit price for 1 bag of *lomien* is $\frac{8}{1} = \$8$.
- The unit price for 2 bags of *lomien* is $\frac{20}{2} = \$10 > \$8$.

Therefore, this *lomien* pricing is ***ridiculous***.

	Input	Output
2	4 5 9 12 16	Normal

- The unit price for 1 bag of *lomien* is $\frac{5}{1} = \$5$.
- The unit price for 2 bags of *lomien* is $\frac{9}{2} = \$4.5$.
- The unit price for 3 bags of *lomien* is $\frac{12}{3} = \$4$.
- The unit price for 4 bags of *lomien* is $\frac{16}{4} = \$4$.

As the unit prices are non-increasing ($\$5 \geq \$4.5 \geq \$4 \geq \$4$), this *lomien* pricing is ***normal***.

	Input	Output
3	5 5 9 12 16 21	Ridiculous

- The unit price for 4 bags of *lomien* is $\frac{16}{4} = \$4$.
- The unit price for 5 bags of *lomien* is $\frac{21}{5} = \$4.2 > \$4$.

Therefore, this *lomien* pricing is ***ridiculous***.

CONSTRAINTS

$1 \leq N \leq 2 \times 10^5$

$1 \leq a_i \leq 10^9$

It is guaranteed that the prices are strictly increasing (i.e. $a_1 < a_2 < \cdots < a_N$).

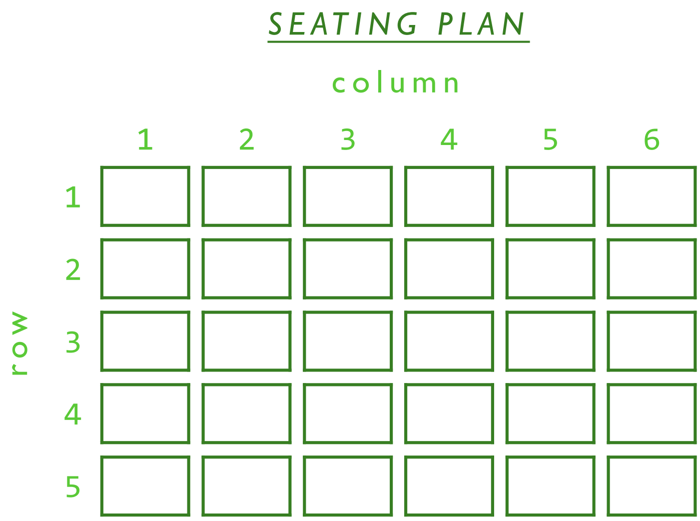
SUBTASKS

	Points	Constraints
1	11	$N = 2$ $1 \leq a_i \leq 100$
2	14	$1 \leq N \leq 10$ $1 \leq a_i \leq 100$
3	5	No additional constraints

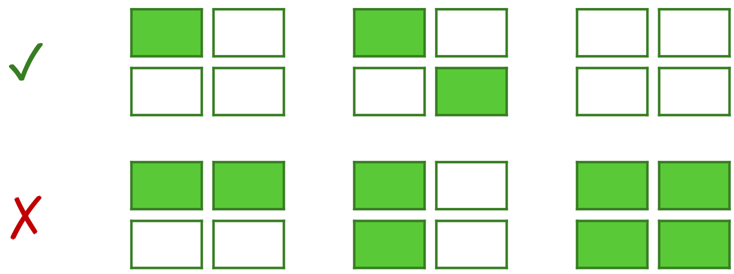
TF233 Seating Plan

Score	40
Time Limit	1s
Memory Limit	256MB

It is now 3:30p.m., time for the Chemistry Uniform Test! Daniel feels nervous.



Suddenly, a new policy is announced by Mr. Yim to reduce virus spreading – not only the tables in the classroom should remain separated, all students must not sit next to, directly behind or directly in front of other students during the Chemistry Uniform Test! If a seating plan satisfies the above requirement, we call it ***virus-proof***.



Suppose there are N rows and M columns of tables (and chairs) in the classroom. As the IT helper, Daniel is being asked by Miss Yhan to construct a ***virus-proof*** seating plan for K Chemistry students. Is this possible, or a larger classroom is needed to contain all students?

INPUT

The first and the only line contains three space-separated integers: N, M and K.

OUTPUT

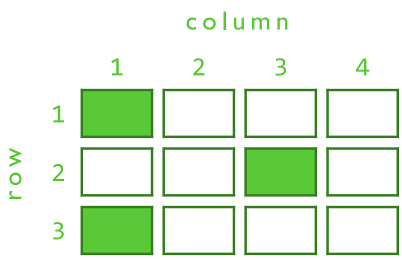
If it is not possible to construct a ***virus-proof*** seating plan using the given data, output `Impossible` on the first and only line.

Otherwise, on the first line, output `Possible`.
On the next N lines, your program should output a string of length-M each, representing the seating plan. The j-th character on the i-th line should be `#` if in the seating plan, a student sits on position (i, j) (i.e. the seat on the i-th row and the j-th column), otherwise output `.` if no student sits on that position. If there are multiple ***virus-proof*** seating plans, you may output any one of them.

SAMPLE TESTS

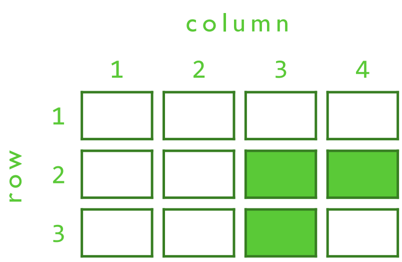
	Input	Output
1	3 4 3	Possible #... ..#. #...

This is one of the ***virus-proof*** seating plans for the input.



	Input	Output
2	3 4 3	Possible .... ..## ..#.

This is **NOT** a ***virus-proof*** seating plan, as students at $(2, 3)$ and $(2, 4)$ are sitting next to each other, while students at $(2, 3)$ and $(3, 3)$ are also sitting next to each other. This output scores 40% in all subtasks.



3	`3 4 7`	`Impossible`

It can be proven that there is no way to construct a ***virus-proof*** seating plan for 7 students in a 3×4 grid.

CONSTRAINTS

$1 \leq N, M \leq 5002$

$0 \leq K \leq N \times M$

SUBTASKS

	Points	Constraints
1	2	$N = 1$ $M = 2$ $K = 2$
2	5	$K = 0$
3	8	$N = M = 2$ $0 \leq K \leq 3$
4	15	M is even
5	10	No additional constraints

SCORING

In all subtasks, you can obtain a **partial score** if you can determine whether it is possible to have a ***virus-proof*** seating plan. For each subtask:

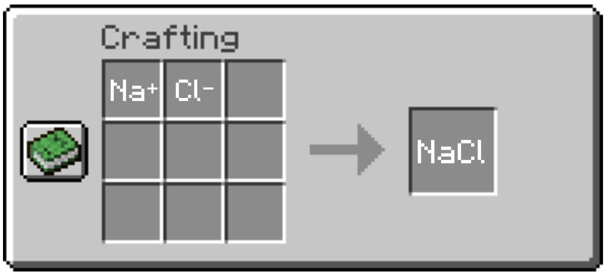
- You score 100% if your output is correct in all test cases within the subtask; otherwise
- You score 40% if the first line of your output is correct in all test cases within the subtask; otherwise
- You score 0.

TF234 Chemcraft Time

Score	50
Time Limit	1s
Memory Limit	256MB

It is finally time for Chemistry Uniform Test. **"O_2 NO"**, exclaims Daniel, **"NO_2"**!

Upon receiving the test paper, Daniel realises that he knows nothing about Chemistry – he just has played some *Chemcraft* yesterday – a game on crafting chemical compounds – and knew that particles can be crafted from some other simpler particles. For instance, he knows that one can combine **Na^+** (sodium cation) and **Cl^-** (chloride anion) to form **NaCl** (sodium chloride, common salt).



Daniel remembers that in *Chemcraft*, there are N kinds of particles. **It is known that particle 1 is always the simplest, particle 2 is slightly more complicated, etc., while particle N is the most complicated.** Each particle i naturally comes with a formation time T_i and a list of K_i **simpler** particles required to craft that particle.

- If $K_i = 0$, then the particle **CAN** be obtained from the nature. T_i units of time are required to obtain 1 unit of that particle.
- If $K_i > 0$, then the particle **CANNOT** be obtained from the nature. T_i units of time and 1 unit of all the K_i **simpler** particles are required to produce 1 unit of that particle. The K_i particles disappear after the new particle is formed. **Note that the K_i particles are not necessarily distinct.**

As Daniel is an experienced player of *Chemcraft*, he wishes to find the crafting time of each of the N particles – forget about the Chemistry UT – the time needed to produce 1 unit of that particle.

INPUT

The first line contains an integer N.

The next N lines describe how the particles are formed. Each line starts with T_i and K_i, the formation time and the number of particles required to form particle i. This is followed by a list of K_i space-separated integers, $p_{i,1}, p_{i,2}, \ldots, p_{i,K_i}$ **(note $p_{i,j} < i$, that is, particle $p_{i,j}$ is simpler than particle i for $1 \le j \le K_i$)**, meaning particle(s) $p_{i,1}, p_{i,2}, \ldots, p_{i,K_i}$ are required to form particle i.

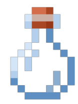
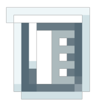
OUTPUT

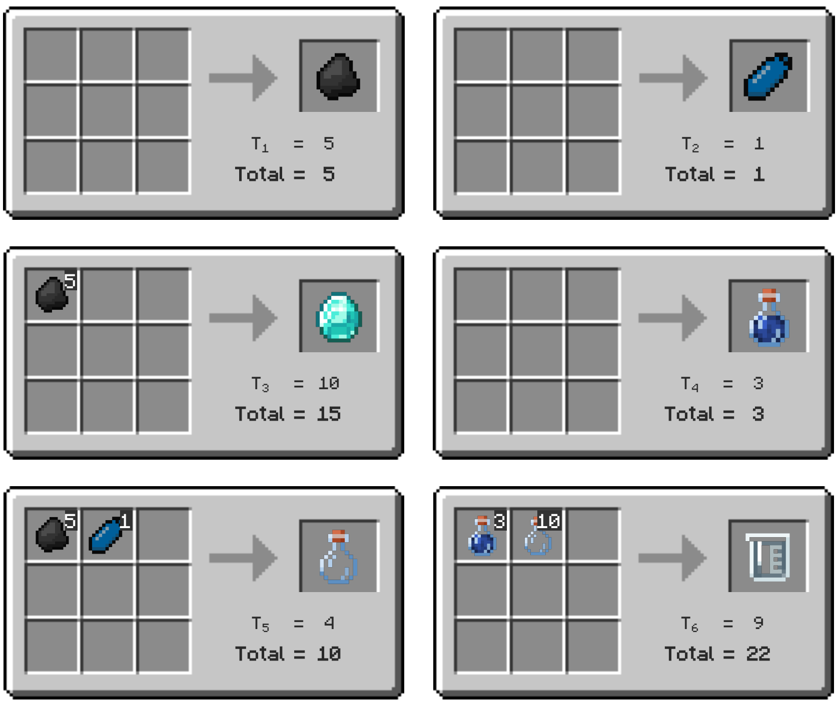
Output a list of N space-separated integers, the crafting time of particle i for $1 \le i \le N$. As the answer can be quite large, please output your answer modulo $10^9 + 7$.

SAMPLE TESTS

Input	Output
6 5 0 1 0 10 1 1 3 0 4 2 1 2 9 2 4 5	5 1 15 3 10 22

In sense of Chemistry, we may name the particles as followings:

Particle 1	Particle 2	Particle 3	Particle 4	Particle 5	Particle 6
					
C	O_2	C(diamond)	H_2O	CO_2	H_2CO_3
Carbon	Oxygen	Diamond	Water	Carbon dioxide	Carbonic acid



- For carbon (C), it can be obtained naturally with $T_1 = 5$. Therefore, the crafting time of carbon is 5.
- For oxygen (O_2), it can be obtained naturally with $T_2 = 1$. Therefore, the crafting time of oxygen is 1.
- For diamond (C), it requires 1 unit of carbon to form with $T_3 = 10$. Therefore, the crafting time of diamond is $10 + 5 = 15$ (i.e. T_3 **PLUS** the crafting time of carbon).
- For water (H_2O), it can be obtained naturally with $T_4 = 3$. Therefore, the crafting time of water is 3.
- For carbon dioxide (CO_2), it requires 1 unit of carbon and 1 unit of oxygen to craft with $T_5 = 4$. Therefore, the crafting time of carbon dioxide is $4 + 5 + 1 = 10$ (i.e. T_5 **PLUS** the crafting time of carbon **PLUS** the crafting time of oxygen).
- For carbonic acid (H_2CO_3), it requires 1 unit of water and 1 unit of carbon dioxide to craft with $T_6 = 9$. Therefore, the crafting time of carbonic acid is $9 + 3 + 10 = 22$ (i.e. T_6 **PLUS** the crafting time of water **PLUS** the crafting time of carbon dioxide).

CONSTRAINTS

$1 \leq N \leq 10^5$

$1 \leq T_i \leq 10^9$

$0 \leq K_i \leq 9$ for $1 \leq i \leq N$

If particle a is used to make particle b, then it is guaranteed that $a < b$.

SUBTASKS

	Points	Constraints
1	7	$K_i = 0$ for $1 \leq i \leq N$
2	13	$K_1 = 0,\ K_i = 1$ for $2 \leq i \leq N$ $p_{i,1} = i - 1$ for $2 \leq i \leq N$ $1 \leq T_i \leq 1000$
3	24	$1 \leq N \leq 10$ $1 \leq T_i \leq 5$
4	6	No additional constraints

TF235 Problem Descriptor

Score	70
Time Limit	1s
Memory Limit	256MB

"What problem do you have?"

"What problem?"

"No, I ask you, what problem do you have?"

"No, what problem do you have, what problem do I have?"

*"I ask you, what **problem** do you have?"*

"Har?"

Daniel hears this legendary conversation after the Chemistry UT on bus route 42! He realises that for all the problems on the judge, there is no description for them – unlike the *Bus Route Category*. As his friend Jack has already done the problem *Bus Route Category*, Daniel decides to ask Jack to make a program that can automatically generate the description from the problem ID. As Jack is busy, he decides to ask you for help instead (*free labour!*).

All problem ID consists of a prefix and different prefixes have different structures follow them:

Prefix	Structure
`X`, `B`, `A`, `W`	`<count>`
`TF`	`<year><count>`
`TW`, `TC`	`<year><round><count>`
Others	Invalid

Each prefix corresponds to a *problem type representation*:

Prefix	Problem Type Representation
`X`	`Original`
`B`	`Basic`
`A`	`Algorithm`
`W`	`Workshop`
`TF`	`Team Formation Test`
`TW`	`Operation Imaginary`
`TC`	`Mini Contest`

<count> represents the problem being the <count>-th problem in that category or contest. The *full count representation* is Problem <count>. Problems with different prefixes have different valid lengths for <count>:

Prefix	Valid Length of <count>
X, B, A, W	3 digits
TF, TW, TC	1 digit

Note that 000 and 0 is a valid 3-digit and 1-digit <count> respectively.

<year> represents the school year that the contest is in. It is defined to be the last two digits of the latter year in the school year. It is given that the school year is between 2021/22 and 2120/21. The *full school year representation* is <full former year>/<last two digits of latter year>, here are some examples:

<year>	Full School Year Representation
22	2021/22
23	2022/23
47	2046/47
00	2099/00
05	2104/05
21	2120/21

<round> means that the contest is the <round>-th round of that kind of contest in the school year. A valid <round> lies between 1 and 9.
For prefix TW, the *full round number representation* is <round><ordinal suffix>,
for prefix TC, the *full round number representation* is Round <roman numeral of round>:

<round>	Full Round Number Representation	
	for Prefix TW	for Prefix TC
1	1st	Round I
2	2nd	Round II
3	3rd	Round III
4	4th	Round IV
5	5th	Round V
6	6th	Round VI
7	7th	Round VII
8	8th	Round VIII
9	9th	Round IX
0	Invalid	

The problem description is constructed by joining specific types of data in the following order:

- (for `TF`, `TW`, `TC` only) *Full school year representation*
- (for `TW` only) *Full round number representation*
- *Problem type representation*
- (for `TC` only) *Full round number representation*
- *Full count representation*

A **single space** should be used to separate each type of data above.

INPUT

The first and the only line contains a string, representing the problem ID.

OUTPUT

If it is not possible to construct a problem description from the problem ID according to the rules given, output `Invalid`.

Otherwise, output the problem description constructed from the problem ID using the rules given on the first and only line.

SAMPLE TESTS

	Input	Output
1	`X007`	`Original Problem 007`
2	`B9`	`Invalid`
3	`TF235`	`2022/23 Team Formation Test Problem 5`
4	`TF1810`	`Invalid`
5	`TW3721`	`2036/37 2nd Operation Imaginary Problem 1`
6	`TC1130`	`2110/11 Mini Contest Round III Problem 0`
7	`TC1101`	`Invalid`

CONSTRAINTS

It is guaranteed that the length of input would not exceed 10 characters, while it only contains digits or upper-case letters (`0` - `9`, `A` - `Z`).

SUBTASKS

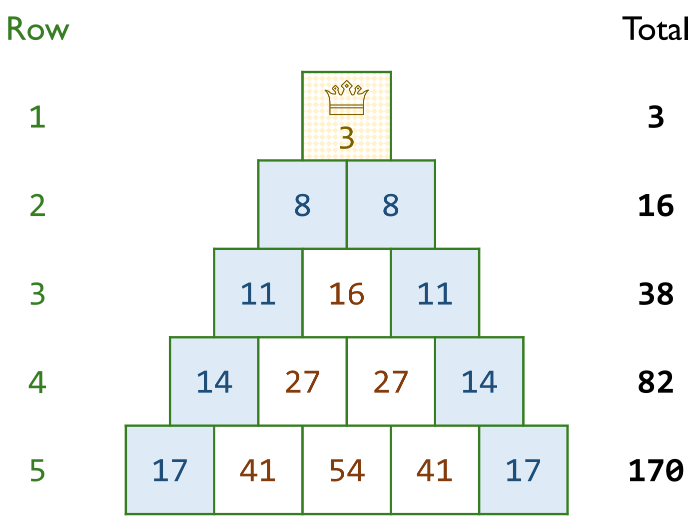
	Points	Constraints
1	8	The problem ID has the prefix of `X`, `B`, `A` or `W` and is guaranteed to be valid
2	9	The problem ID has the prefix of `TF` and is guaranteed to be valid
3	8	The problem ID has the prefix of `TW` and is guaranteed to be valid
4	17	The problem ID is guaranteed to be valid
5	28	No additional constraints

TF236 King's Army

Score	90
Time Limit	1s
Memory Limit	256MB

Back to home, Daniel decides to watch his friend Duncan's stream on the game "*Team Fight Territory (TFT)*".

In the game, Duncan is fighting against *King Charles IV* and *Queen Consort Nicoria*. As 知彼知己，百戰不殆, Duncan wants to know the fighting abilities of *King Charles IV*'s army.



The army consists of R rows arranged in a triangular shape. Only *King Charles IV* is in the 1st row (as he needs to lead the army). For the remaining rows, there are n soldiers in the n-th row, for example there are 2 soldiers in the 2nd row, and 3 soldiers in the 3rd row.

The fighting abilities of *King Charles IV* and the soldiers are defined as follows:

- The fighting ability of *King Charles IV* would be his leadership level L.
- The fighting ability of each soldier in the 2nd row is the sum of base ability B and *King Charles IV*'s leadership level L.
- From the 3rd row onwards, the fighting ability of the leftmost/rightmost soldiers in the n-th row is the sum of the fighting ability of the leftmost/rightmost soldier in the $(n-1)$-th row and *King Charles IV*'s leadership level L.
- For each of the remaining soldiers, his fighting ability is the sum of the fighting abilities of the two soldiers standing directly in front of him.

Since Duncan only needs to face one row in the army at each moment, he wants to know the total fighting abilities of some specific rows in the army.

INPUT

The first line contains 4 space-separated integers, R, L, B and Q, representing the number of rows in the army, the leadership level of *King Charles IV*, the base ability of solders in the 2nd row and the number of queries respectively.

For the next Q lines, the i-th $(1 \leq i \leq Q)$ line contains a single integer r_i, representing the row which Duncan wants to know its total fighting ability.

OUTPUT

For each of the Q queries, output the total fighting ability of the r_i-th row on a separate line, modulo $10^9 + 7$.

SAMPLE TESTS

Input	Output
5 3 5 5 1 2 3 4 5	3 16 38 82 170

This sample corresponds to the diagram shown in the problem statement.

CONSTRAINTS

$2 \leq R \leq 10^{18}$

$0 \leq L, B \leq 10^9$

$1 \leq Q \leq 2 \times 10^5$

$1 \leq r_i \leq R$

SUBTASKS

	Points	Constraints
1	13	$L = 0$ $B = 1$ $1 \le R, Q \le 10$
2	7	$L = 0$ $0 \le B \le 10$ $1 \le R, Q \le 10$
3	12	$L = 1$ $B = 0$ $1 \le R, Q \le 10$
4	8	$0 \le L \le 10$ $B = 0$ $1 \le R, Q \le 10$
5	20	$0 \le L, B \le 10$ $1 \le R, Q \le 10$
6	17	$1 \le R, Q \le 5000$
7	12	$1 \le R, Q \le 10^5$
8	1	No additional constraints

TF237 Blackboard Escape

Score	90
Time Limit	1s
Memory Limit	256MB

After watching the fierce fighting of Duncan against *King Charles IV* in the game, Daniel feels tired and quickly falls asleep. Suddenly, Daniel realises he is being trapped in the classroom (alone) in his dream! He quickly finds that he has the ability to control his dream. However, he is forced to finish a mysterious game before escaping the classroom. In order to get out and gain freedom, he needs to finish the game quickly.

There is a long blackboard in the classroom. Initially, the blackboard contains two numbers 0 and 1 only. In each round, he can choose two numbers x and y (not necessarily distinct nor positive) from the blackboard and write down a new number $2y - x$ onto the blackboard. All the numbers are preserved. He will be able to escape from the classroom once he writes down the integer N.

Daniel calls you for help, as he has a mobile phone with him (*Re:Zero logic*). Can you help him to minimise the number of rounds K needed, so that he can escape from the classroom as soon as possible?

INPUT

The first and the only line contains a positive integer: N.

OUTPUT

On the first line, output the number of operations K you used.
On the next K lines, each output two space-separated integers: x and y, as specified above.

For each test case, the output from your program must satisfy:

- The number of rounds K does not exceed 10^5.
- The numbers x and y are between -10^9 and 10^9.
- The numbers x and y are present on the blackboard at the round you use them.

If you violate any of the above rules, you will receive zero score and the "Wrong Answer" verdict.

If there are multiple solutions, you can print anyone of them.

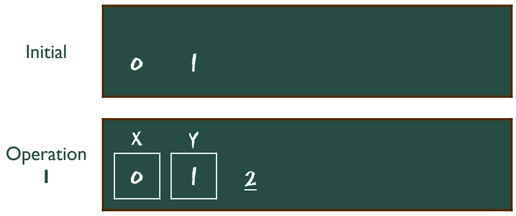
SAMPLE TESTS

1

Input	Output
2	1 0 1

Initially, there are two integers on the blackboard, 0 and 1.

In operation 1, $x = 0$, $y = 1$, $2 \times 1 - 0 = 2$ is written.



2

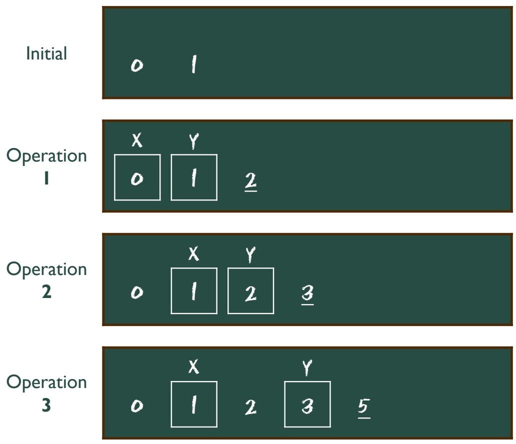
Input	Output
5	3 0 1 1 2 1 3

Initially, there are two integers on the blackboard, 0 and 1.

In operation 1, $x = 0$, $y = 1$, $2 \times 1 - 0 = 2$ is written.

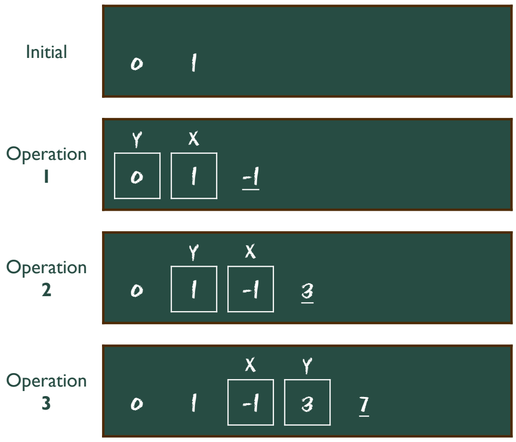
In operation 2, $x = 1$, $y = 2$, $2 \times 2 - 1 = 3$ is written.

In operation 3, $x = 1$, $y = 3$, $2 \times 3 - 1 = 5$ is written.



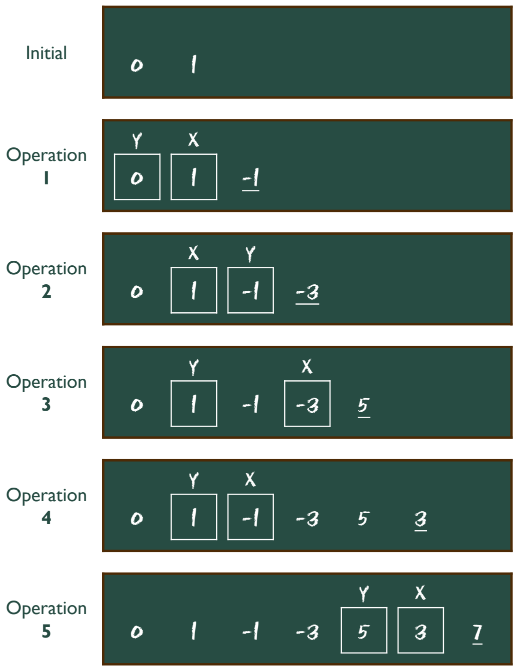
3

7	3 1 0 -1 1 -1 3



4

7	5 1 0 1 -1 -3 1 -1 1 3 5



CONSTRAINTS

$2 \le N \le 5 \times 10^8$

SUBTASKS

	Points	Constraints
1	2	$N = 2$
2	3	$N = 3$
3	4	$N = 4$
4	5	$N = 5$
5	6	$N = 6$
6	20	$2 \le N \le 10^5$
7	50	No additional constraints

SCORING

In subtask 7, your score is determined by the maximum number of queries $K_{\max}$ you used in all test cases within the subtask.

$$score = \begin{cases} 50 & \text{if } K_{\max} \le 30 \\ 35 + \dfrac{60 - K_{\max}}{2} & \text{if } 30 < K_{\max} \le 60 \\ 25 + \sqrt{\dfrac{5000}{K_{\max}}} & \text{if } 60 < K_{\max} \le 6000 \\ 12 + 10 \log \dfrac{10^5}{K_{\max}} & \text{if } 6000 < K_{\max} \le 10^5 \\ 0 & \text{if } K_{\max} > 10^5 \end{cases}$$