Problem A. Dancing Zombie

On a dark and stormy night, Dr. Zomboss dispatched a vanguard of n Dancing Zombies to breach Dave's defenses. When the upbeat music begins, the Dancing Zombies will appear one by one on the number line in a predetermined rhythm and start spreading out in all directions.

The specific rules are as follows:

- The i-th Dancing Zombie will appear at position b_i on the number line at the a_i-th second.
- Every Dancing Zombie located at coordinate x will, one second after its appearance and every second thereafter, summon a new Dancing Zombie at each of its two adjacent integer coordinates $x-1$ and $x+1$. The newly summoned Dancing Zombies also possess the aforementioned summoning ability, and multiple Dancing Zombies can exist at the same coordinate simultaneously.

The goal of Dr. Zomboss is to cover the entire battlefield with Dancing Zombies. What is the minimum number of seconds required to ensure that every integer coordinate in the interval $[0, k]$ on the number line contains at least one Dancing Zombie?

Input

The first line contains two integers n, k ($1 \le n \le 10^5$, $0 \le k \le 10^{12}$), representing the total number of Dancing Zombies and the right endpoint of the area to be covered, respectively.

The next n lines each contain two integers a_i, b_i ($0 \le a_i, b_i \le k$), representing the time and initial position of the ith Dancing Zombie, respectively.

Output

Output a single integer on a single line, representing the time required for Dancing Zombies to cover all integer coordinates in the interval $[0, k]$ on the number line.

Examples

standard input	standard output
3 5 0 0 0 5 1 2	2
2 102 0 102 6 66	72

Note

For the first example, at the end of each second, the number of Dancing Zombies in the interval $[0, 5]$ on the number line is as follows:

- Second 0: $\{1, 0, 0, 0, 0, 1\}$;
- Second 1: $\{1, 1, 1, 0, 1, 1\}$;
- Second 2: $\{2, 3, 2, 2, 2, 2\}$.

At second 2, there is at least one Dancing Zombie at every integer coordinate in the interval $[0, 5]$ on the number line.

Problem B. Galaxy of Stars in a Dream

Everyone has their own galaxy of stars that they long for but cannot reach.

In a dream, Xinghe has a non-negative integer x. He needs to find three non-negative integers a, b, c such that $a \times b + c = x$. Let the maximum among these three numbers be $M = \max(a, b, c)$, and the minimum be $m = \min(a, b, c)$. His goal is to make the range $M - m$ as small as possible. Please help him determine the minimum possible value of $M - m$ among all triples (a, b, c) satisfying the condition.

Input

The first line contains an integer T ($1 \le T \le 10^5$), denoting the number of test cases.

For each test case, there is one line containing an integer x ($0 \le x \le 10^9$), as described in the statement.

Output

For each test case, output one integer on a separate line, representing the minimum value of $M - m$.

Example

standard input	standard output
4 1 12 1000 729320	1 0 8 5

Problem C. Binary String

Given a binary string $S = S_1S_2 \dots S_n$ of length n, you need to insert a bitwise operator between every two adjacent digits S_i and S_{i+1} (there are $n-1$ such positions), so that the final value of the resulting expression is 0.

There are three operators to choose from, with the following rules:

- Bitwise AND (`&`): for the expression a `&` b, its value is 1 if and only if $a = b = 1$; otherwise, it is 0.
- Bitwise XOR (`^`): for the expression a `^` b, its value is 1 if and only if $a = 1, b = 0$ or $a = 0, b = 1$; otherwise, it is 0.
- Bitwise OR (`|`): for the expression a `|` b, its value is 0 if and only if $a = b = 0$; otherwise, it is 1.

Note that in this problem, the precedence of bitwise operators is the same as in C, according to the following rules:

1. Bitwise AND (`&`) has the highest precedence.
2. Bitwise XOR (`^`) has lower precedence than bitwise AND, but higher precedence than bitwise OR.
3. Bitwise OR (`|`) has the lowest precedence.
4. Operators with the same precedence are evaluated from left to right.

You need to construct a valid sequence of operators such that the final value of the whole expression is 0. If there are multiple valid solutions, output any of them. It can be proved that for every valid input, at least one valid construction always exists.

Input

The first line contains a positive integer n ($2 \le n \le 10^5$), which denotes the length of the string.

The second line contains a string S of length n consisting only of 0 and 1.

Output

Output one line containing a string of length $n-1$, representing the operators inserted in order. If there are multiple valid solutions, output any of them.

Example

standard input	standard output
3 000	&&

Note

In the sample, the original string is `000`. Two bitwise AND operators can be inserted in the middle to make the expression `0&0&0`, whose result is `0`. Therefore, output `&&`.

Problem D. Bracket Convex Hull

Given n points on a two-dimensional plane, each point is labeled with a bracket, which may be either a left bracket `(` or a right bracket `)`.

We say that a string S is a valid bracket sequence if and only if it can be generated by the following recursive rules:

- The empty string is a valid bracket sequence;
- If A is a valid bracket sequence, then the string `(` A `)` is also a valid bracket sequence;
- If both A and B are valid bracket sequences, then AB is also a valid bracket sequence;
- Any string that cannot be generated by the above rules is not a valid bracket sequence.

For example, `()`, `(())`, and `()()` are valid bracket sequences, while `)(`, `((`, and `())(` are not valid bracket sequences.

Now you need to choose some distinct points from the given points as vertices to form a convex polygon.

In this problem, a polygon formed by m points $p_{a_1}, p_{a_2}, \ldots, p_{a_m}$ is called a convex polygon if and only if the following conditions are satisfied:

- $m \geq 3$;
- Connecting these points in the order $a_1, a_2, \ldots, a_m$, and then connecting a_m and a_1, forms a simple polygon;
- For every edge of the polygon, all other vertices lie strictly on the same side of the line containing that edge.

In other words, the selected points must appear exactly in the cyclic order of their own convex hull, and all interior angles must be strictly less than 180°; that is, the convex polygon has no three collinear points.

For a convex polygon, choose any vertex on its boundary as the starting point, and traverse all vertices along the polygon boundary in either clockwise or counterclockwise direction, stopping just before returning to the starting point. This yields a bracket sequence of length m: if the current vertex is labeled with a left bracket, write `(`; if it is labeled with a right bracket, write `)`.

Your task is to find a convex polygon containing at least one left bracket such that, for any vertex on the polygon labeled with a left bracket, the bracket sequence obtained by starting from that vertex and traversing the polygon boundary in either direction is a valid bracket sequence.

If such a convex polygon exists, output any one of them; otherwise, report no solution.

Input

The first line contains an integer T ($1 \leq T \leq 400$), the number of test cases.

For each test case, the first line contains an integer n ($1 \leq n \leq 2000$), the number of points.

The next n lines each contain three integers x_i, y_i, t_i ($0 \leq x_i, y_i \leq 10^9, t_i \in \{0, 1\}$), representing the coordinates and bracket type of the i-th point. Here, $t_i = 0$ denotes a left bracket, and $t_i = 1$ denotes a right bracket.

It is guaranteed that, within each test case, all points are distinct and no three points are collinear.

It is also guaranteed that the sum of n over all test cases does not exceed 2000.

Output

For each test case, if there is no solution, output a single integer -1 on one line.

Otherwise, output the number of vertices m of the selected convex polygon on the first line.

On the second line, output m pairwise distinct integers $a_1, a_2, \ldots, a_m$, representing the indices of the selected points.

These points must be output in the order along the boundary of the convex polygon, either clockwise or counterclockwise.

Example

standard input	standard output
5 4 1 1 0 2 4 1 3 9 0 4 16 1 5 1 1 0 2 4 0 3 9 0 4 16 1 5 25 1 6 47 58 0 30 23 0 27 34 1 35 7 1 10 30 1 1 25 1 8 10 5 0 10 16 0 1 5 0 24 9 0 6 2 0 6 12 0 7 18 1 3 13 1 1 0 0 0	4 1 2 3 4 -1 4 1 3 2 4 -1 -1

Problem E. High-Dimensional Geometry

When a geometric problem is extended to three dimensions or even higher dimensions, its difficulty can increase significantly. To address complex n-dimensional geometric structures, researchers have proposed a model in which the 2^n key positions in space are abstracted as vertices of a graph, and specific geometric constraints between these positions are abstracted as edges in the graph.

You now have a model that has been abstracted into a graph, in which complex geometric features have been transformed into a simple undirected graph containing 2^n vertices with no duplicate edges or self-loops. However, for certain reasons, you cannot directly observe the edges in the graph. You want to know the total number of edges, m.

You may use a "subset detection detector" which works as follows:

- You input a subset S containing k vertices into the detector. The detector requires that the size of each set examined must be **exactly** half the total number of vertices in the graph, i.e., $k = 2^{n-1}$.
- The detector scans these vertices and their associated edges, and returns the total number of edges in the graph where **at least one endpoint** belongs to the set S.

Due to the high cost, the detector can be used at most 2^{n+1} times. Given the limited number of queries, find the total number of edges m in the graph.

Interaction Protocol

First, read an integer n from standard input ($1 \le n \le 10$). This means the graph contains 2^n vertices, numbered from 1 to 2^n.

To make a query, output `? s` to standard output, where s is a binary string of length 2^n. The i-th character should be:

- `1`, if vertex i is included in S;
- `0`, otherwise.

You must ensure that the number of `1`s in the string is exactly 2^{n-1}. If you make an invalid query or exceed the query limit, the interactor will terminate your program.

After each query, you need to read an integer d from standard input, which denotes the number of edges that have at least one endpoint in the subset S.

To output the answer, output `! m` to standard output, where m is the total number of edges. After outputting the answer, you must terminate the program immediately.

The interactor is **non-adaptive**. This means that the graph is fixed before the interaction begins and does not change according to your queries.

Note: After each output, you must print a newline and flush the standard output buffer. Use the following methods to flush:

- For C or C++, use `fflush(stdout)` or `cout.flush()`
- For Java, use `System.out.flush()`
- For Python, use `stdout.flush()`

Example

standard input	standard output
2 1 0	 ? 1010 ? 0101 ! 1

Problem F. Counting

Little L has a non-negative integer array $a = [a_1, a_2, \ldots, a_n]$ of length n. The elements in the array may repeat, and each element can be any non-negative integer (that is, $a_i \geq 0$, with no upper bound). Little L chooses an unknown positive integer m, and performs a modulo operation on every element of array a, obtaining a new array b, where $b_i = a_i \bmod m$.

Now Little L tells you the array b, but both the original array a and the modulus m are unknown. He wants to know: among all possible original arrays a and modulus m, how many different values can its mex take?

The definition of mex: the smallest non-negative integer that does not appear in the array. For example, $\text{mex}\{0, 1, 3\} = 2$, and $\text{mex}\{1, 2, 3\} = 0$.

Input

The first line contains an integer n ($1 \leq n \leq 10^5$), the length of the array.

The second line contains n integers $b_1, b_2, \ldots, b_n$ ($0 \leq b_i \leq 10^9$), representing the array b given by Little L.

Output

Output an integer representing the number of possible values of mex of the original array a.

Example

standard input	standard output
6 0 1 2 3 1 2	5

Note

For the example, all possible mex values are $\{0, 1, 2, 3, 4\}$. For example, when mex $= 3$, one possible construction is $m = 5$, $a = \{0, 1, 2, 8, 6, 7\}$.

Problem G. Passing Ball Problem

A football team consists of n players (numbered 1 through n) and a goalkeeper, and they are practicing passing.

The practice procedure is as follows:

1. The goalkeeper kicks off first, passing the ball with equal probability to any one of the n players. At this point, all players have a pass count of 0.
2. After receiving the ball, a player passes it according to a specific preference. If player i has the ball, the probability that he passes it to player j is $p_{i,j}$.
3. Each time a player kicks the ball, their cumulative number of passes increases by 1. If, after a pass, **the number of passes for all n players is exactly even**, the goalkeeper will immediately step in to stop the ball, and the practice ends.

Now the goalkeeper wants to know the expected total number of passes made by the players when he stops the ball.

Input

The first line contains a positive integer n ($1 \le n \le 16$), denoting the number of players.

The next n lines each contain n integers $a_{i,j}$ ($0 \le a_{i,j} < 998244353$). Here, after player i gets the ball, the probability that they pass it to player j is $p_{i,j} = \frac{a_{i,j}}{\sum_{k=1}^{n} a_{i,k}}$.It is guaranteed that for every $1 \le i \le n$, we have $(\sum_{k=1}^{n} a_{i,k}) \not\equiv 0 \pmod{998244353}$.

Output

If the expectation converges, output the answer modulo 998244353; otherwise, output `infinity`.

It can be proven that if the expectation converges, then it can always be written as a rational number $\frac{p}{q}$, where p, q are integers, $p \ge 0$, and $q \ge 1$. You should output the value of $p \cdot q^{-1} \bmod 998244353$, where q^{-1} denotes the modular inverse of q modulo 998244353, which satisfies $q \cdot q^{-1} \equiv 1 \pmod{998244353}$.

Examples

standard input	standard output
5 1 0 0 0 0 2 1 0 0 0 3 0 1 0 0 4 0 0 1 0 5 0 0 0 1	infinity
5 0 1 0 0 0 0 0 1 0 0 0 0 0 1 0 0 0 0 0 1 1 0 0 0 0	10
6 5 6 499122175 10 10 998244352 7 998244351 7 4 499122176 3 4 10 2 9 8 9 6 6 6 3 10 10 998244351 3 1 499122175 10 9 3 7 6 499122176 499122176 3	53368493
1 8	2

Problem H. Treasure Island

Sail is on Treasure Island. He foresees that n events will occur on the island, and he will leave after all events have finished. Specifically, there are the following three types of events:

1. Some gold coins appear. For each gold coin, he can either convert it into k yuan, or feed it to the God of Wealth to obtain a bullet with damage k;
2. Some silver coins appear. For each silver coin, he can either convert it into 1 yuan, or feed it to the God of Wealth to obtain a bullet with damage 1;
3. A monster appears. Whenever a monster with x HP appears, Sail must immediately attack it using some bullets, provided that the total damage of the selected bullets is **greater than or equal to** x. If this condition cannot be met, Sail will be killed by the monster. Once used, bullets are consumed and cannot be reused.

He wants to know whether he can leave the island safely, and if so, what is the maximum amount of money he can have left when leaving.

Input

The first line contains a positive integer T ($1 \le T \le 10^4$), indicating the number of test cases.

For each test case, the first line contains two positive integers n ($1 \le n \le 2 \times 10^5$) and k ($1 \le k \le 10^6$), where n is the total number of events, and k is the amount of money obtained from exchanging coins or the damage value of a bullet.

The next n lines each contain two positive integers o ($o \in \{1, 2, 3\}$) and x ($1 \le x \le 10^6$), representing an event.

- If $o = 1$ or $o = 2$, x represents the number of gold or silver coins that appear;
- If $o = 3$, x represents the HP of the monster.

It is guaranteed that the sum of n over all test cases does not exceed 2×10^5.

Output

For each test case, if Sail cannot leave safely, output -1; otherwise, output the maximum amount of money that can remain.

Example

standard input	standard output
5 4 3 1 2 3 5 2 3 3 3 4 3 1 2 2 2 3 5 3 1 4 3 1 2 2 2 3 5 3 3 4 3 1 2 3 5 2 2 3 3 4 3 1 2 1 3 3 4 3 9	0 1 0 -1 0

Note

For the second test case, Sail does the following in order:

1. Two gold coins appear. Sail exchanges them for two bullets with damage 3.
2. Two silver coins appear. Sail exchanges one of them for a bullet with damage 1, and the other for 1 yuan.
3. A monster with HP 5 appears. Sail uses two bullets with damage 3 to kill it.
4. A monster with HP 1 appears. Sail uses one bullet with damage 1 to kill it.

In the end, Sail leaves with 1 yuan.

For the fourth test case, Sail does the following in order:

1. Two gold coins appear. Sail exchanges them for two bullets with damage 3.
2. A monster with HP 5 appears. Sail uses two bullets with damage 3 to kill it.
3. Two silver coins appear. Sail exchanges them for two bullets with damage 1.
4. A monster with HP 3 appears. The total damage of the bullets Sail currently has is less than 3, so he is killed.

No matter how he performs the exchanges, Sail cannot leave safely, so you should output -1.

Problem I. The Test of the Clever Pig

Little H is a clever pig who can use artificial intelligence. Today, he is participating in the Little Pig IQ Test Contest, but he got stuck on the last problem:

- Given two strings s and t consisting of lowercase letters, find the lexicographically smallest common subsequence of length 2 between them.

A string a is a subsequence of a string b if and only if a can be obtained from b by deleting some characters (or none) without changing the relative order of the remaining characters.

Little H does not know how to solve this problem. As Little H's agent, please help him.

Input

The first line contains an integer T ($1 \le T \le 2 \times 10^4$), representing the number of test cases.

For each test case, two strings s and t are given on two separate lines ($1 \le |s|, |t| \le 10^5$). It is guaranteed that all strings **consist only of lowercase letters**.

It is guaranteed that over all test cases, the sum of $|s| + |t|$ does not exceed 2×10^5. $|s|$ and $|t|$ denote the lengths of strings s and t, respectively.

Output

For each test case, output one line containing a string representing the answer you found.

If the two strings do not have a common subsequence of length 2, output one line containing `HENG!`.

Example

standard input	standard output
3 azzza zazaz azzza zbzbz you ak	aa zz HENG!

Problem J. Splendor

Little S really likes playing board games, especially *Splendor*. Now he has simplified and modified the rules.

The game includes the following components:

- Chips: There are 5 colors, numbered $1, 2, 3, 4, 5$, and the number of chips of each color can be considered infinite.
- Cards: There are n cards, stacked in order to form a **deck**, with the cards numbered $1, 2, \ldots, n$. Each card i has a color c_i ($1 \le c_i \le 5$) and the quantities of the five types of chips required for exchange: $p_{i,1}$, $p_{i,2}$, $p_{i,3}$, $p_{i,4}$, and $p_{i,5}$.

At the beginning of the game, the top 4 cards (cards 1 to 4) of the deck are revealed on the table to serve as the initial **exchangeable cards**. The remaining cards in the deck are now cards 5 through n.

In each turn, you may perform exactly one of the following three operations:

1. Choose one color and take 2 chips of that color;
2. Choose three different colors and take 1 chip of each chosen color;
3. Choose a card i from the **exchangeable cards** on the table, pay the corresponding chips, and exchange it. When you exchange card i, the number of chips of each color k required is subject to a discount based on the **previously exchanged cards**: If you have previously exchanged d_k cards of color k, then when exchanging the current card, the actual number of chips of color k required is $\max(0, p_{i,k} - d_k)$.

Whenever you successfully exchange an **exchangeable card** on the table:

1. Remove the card from the table;
2. If there are still face-down cards in the deck, draw 1 card from the top of the deck in order and place it on the table as an **exchangeable card**;
3. **Clear all remaining chips from your current hand (note that this rule differs from the original rules)**.

Your goal is to empty the **deck**. Note that there may still be some **exchangeable cards** left on the table at this point; this situation is also considered a successful completion of the goal. Please calculate the minimum number of turns required to achieve this goal.

Input

The first line contains an integer n ($4 \le n \le 100$).

For the next n lines, the i-th line contains 6 integers, respectively c_i, $p_{i,1}$, $p_{i,2}$, $p_{i,3}$, $p_{i,4}$, $p_{i,5}$ ($1 \le c_i \le 5$, $0 \le p_{i,j} \le 10^9$), indicating the color of the i-th card and the number of chips required.

Output

Output an integer representing the minimum number of turns.

Examples

standard input	standard output
5 1 4 1 1 7 1 1 2 2 8 4 8 1 2 3 6 7 2 1 6 5 7 3 1 1 9 9 9 9 9	7
7 2 15 34 0 15 24 4 11 20 0 18 0 1 4 0 2 48 20 3 17 0 8 43 70 1 0 58 7 1 52 1 64 0 81 40 0 2 4 0 0 0 10	86

Note

For the first example: For the first four cards, it takes 6, 8, 7, and 8 operations, respectively, to accumulate the required number of chips through chip-taking operations.

The optimal strategy is: First, spend 6 moves to collect chips to meet the requirement of the first card, then use the 7-th move to "exchange cards". After the exchange is complete, the 5-th card (the last one) in the deck is placed on the table, at which point the deck is empty. This takes a total of 7 moves.

The specific steps are as follows:

1. Take 1 chip each of colors 1, 4, and 5.
2. Take 1 chip each of colors 1, 3, and 4.
3. Take 1 chip each of colors 1, 4, and 5.
4. Take 1 chip each of colors 2, 4, and 5.
5. Take 1 chip each of colors 1, 2, and 4.
6. Take 2 chips of color 4. After the first 6 rounds, the number of chips in hand is: 4 of color 1, 2 of color 2, 1 of color 3, 7 of color 4, and 3 of color 5, which is sufficient to pay the chip requirements for the first card $(4, 1, 1, 7, 1)$.
7. Select the first card, pay the corresponding chips, and exchange it. After the exchange is complete, the 5-th card (the last one) from the deck is placed on the table, at which point the deck is empty. The goal is achieved in a total of 7 rounds.

Problem K. Stock Trading

Alice is a quantitative trader. She is backtesting a high-frequency trading strategy now. Given a stock's price sequence at n time points, denoted as $v_1, v_2, \ldots, v_n$.

The trading bot Alice has set up is controlled by two parameters: a buy threshold a and a sell threshold b ($a < b$), and strictly follows the following rules:

- If the price of the stock at a given time $v_i \le a$, the bot will buy exactly 1 share;
- If the price of the stock at a given time $v_i \ge b$, and the bot currently holds at least 1 share, the bot will immediately sell 1 share;
- At each time point, the bot can buy or sell at most 1 share;
- After the transaction at the n-th time point is completed, if the robot still holds unsold shares, a forced liquidation mechanism is triggered, selling all remaining shares at the price v_n.

Given a fixed buy threshold a, there are m independent queries, each specifying a sell threshold b. For each query, please help Alice calculate the total profit under this strategy (total proceeds from sales minus total costs from purchases).

Input

The first line contains a positive integer T ($1 \le T \le 10^4$), the number of test cases.

For each test case, the first line contains two positive integers n, a ($1 \le n \le 2 \times 10^5$, $1 \le a < 10^9$), representing the number of time points and the buy threshold.

The second line contains n integers $v_1, v_2, \ldots, v_n$ ($1 \le v_i \le 10^9$), giving the historical stock prices at each time point in chronological order.

The third line contains an integer m ($1 \le m \le 2 \times 10^5$), representing the number of queries.

The next line contains m integers $b_1, b_2, \ldots, b_m$ ($a < b_i \le 10^9$), representing the sell threshold for each query.

It is guaranteed that over all test cases, the sum of n and the sum of m do not exceed 2×10^5.

Output

For each test case, output one line containing m integers separated by spaces, representing the final profit of the trading strategy for each query.

Example

standard input	standard output
2 5 10 5 8 15 12 1 3 11 13 16 6 10 5 5 15 12 20 11 3 11 13 16	14 3 -11 17 25 21

Note

For the first query on the first test case ($a = 10, b = 11$), the trading process is as follows:

1. $v_1 = 5 < a$: Buy 1 share, cumulative profit -5, holding 1 share;
2. $v_2 = 8 < a$: Buy 1 share, cumulative profit -13, holding 2 shares;
3. $v_3 = 15 > b$: Sell 1 share, cumulative profit 2, holding 1 share;
4. $v_4 = 12 > b$: Sell 1 share, cumulative profit 14, holding 0 shares;
5. $v_5 = 1 < a$: Buy 1 share, cumulative profit 13, holding 1 share;
6. Liquidation: Sell the remaining 1 share at $v_5 = 1$, final profit 14.

Problem L. Two-Player Game

Insight and Maya are playing a game. Initially, they have an array $a = [a_1, a_2, \ldots, a_n]$ of length n containing only positive integers. **Insight moves first**, and the two players take turns. In each round, the current player must perform the following steps:

1. Select an index i ($1 \le i \le n$) from the array such that $a_i > 0$;
2. For all indices j such that $j \neq i$, update a_j to $\min(a_j, a_i)$;
3. Subtract 1 from the value of a_i.

When it is a player's turn, if all elements in the array are 0 (in which case no positive number a_i can be selected), that player loses the game, and the other player wins. Assuming both players adopt optimal strategies to ensure their own victory, determine who will ultimately win the game.

Input

The first line contains an integer T ($1 \le T \le 10^4$), the number of test cases.

For each test case, the first line contains an integer n ($1 \le n \le 5 \times 10^5$), representing the length of the array.

The next line contains n positive integers $a_1, a_2, \ldots, a_n$ ($1 \le a_i \le 10^9$), representing the initial array.

It is guaranteed that the sum of n over all test cases does not exceed 2×10^6.

Output

For each test case, output one line: if Maya has a strategy that allows her to win, output `Maya`; otherwise, output `Insight`.

Example

standard input	standard output
3 3 2 1 3 4 2 1 3 1 4 2 1 1 4	Insight Insight Maya