Problem A. The Grand Tournament

Input file: `standard input`
Output file: `standard output`

Today, The First Grand Tournament of Automated Driving has officially commenced!

The experiment field of this tournament is a rectangular region on a 2-dimensional plane, with axes parallel to the coordinate axes. The bottom-left corner of the field is at coordinate (x_l, y_l) while the top-right corner is at coordinate (x_r, y_r). There are two segments A and B lying strictly inside the rectangle. The two segments may share common points. There is also a car inside the rectangle, which can be regarded as a point.

A subtask of this tournament requires that the distances between the car and the two segments must be equal all the time during the movement. The distance between a point P and a segment Q is defined as the minimum Euclidean distance from P to any point on Q.

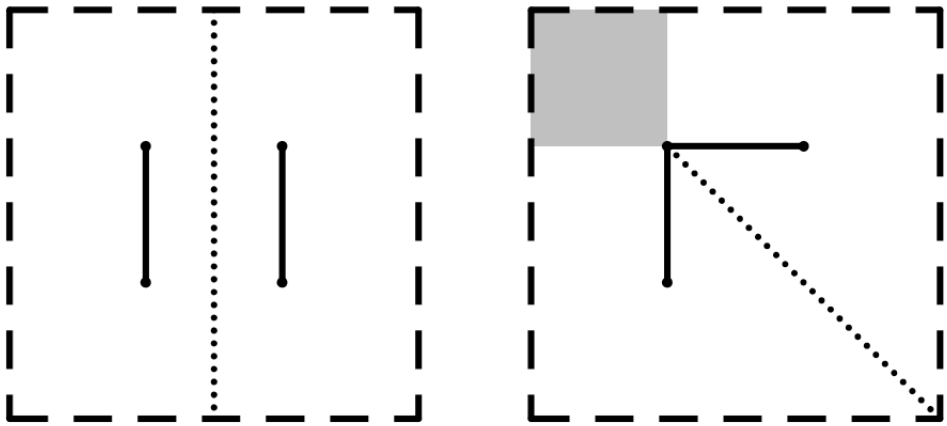

Figure 1: Explanation of the sample data.

Please write a program to find the area of valid positions of the car.

Input

The input contains multiple cases. The first line of the input contains a single integer T ($1 \le T \le 10^5$), indicating the number of test cases.

For each case, the first line of the input contains four integers x_l, y_l, x_r, y_r ($-1\,000 \le x_l < x_r \le 1\,000$, $-1\,000 \le y_l < y_r \le 1\,000$), denoting the coordinates of the bottom-left and the top-right corners of the rectangle. Each of the next two lines contains four integers x_1, y_1, x_2, y_2, denoting a segment that connects (x_1, y_1) and (x_2, y_2), where $x_1, x_2 \in (x_l, x_r)$ and $y_1, y_2 \in (y_l, y_r)$.

For each case, it is guaranteed that the two endpoints of each segment do not coincide.

Output

For each test case, print a single line containing a single real number, the area of valid positions of the car. Your answer will be considered correct if the absolute or relative error does not exceed 10^{-9}.

Formally, if your answer is a and the jury's answer is b, then your answer will be considered correct if and only if $\frac{|a-b|}{\max\{1,|b|\}} \le 10^{-9}$.

Example

standard input	standard output
2 0 0 3 3 1 1 1 2 2 1 2 2 0 0 3 3 1 1 1 2 1 2 2 2	0.000000000000000 1.000000000000000

Problem B. Whispers of the Old Gods

Input file: `standard input`
Output file: `standard output`

Legend has it that the Old Gods' voices are dread whispers that no mere mortal can hope to resist. And now, the time of their awakening draws near. Will you head their whispers at your ear?

We may represent a whisper by a sequence of digits. And surprisingly, the whispers of the Old Gods can be characterized by a *regular expression* (`<regex>`). The syntax of a regular expression is shown below (in Backus-Naur form):

```
<regex> ::= <regex> <regex>
          | <regex> "|" <regex>
          | <regex> "+"
          | <atomic-regex>

<atomic-regex> ::= <digit>
                 | "[" <digit-sequence> "]"
                 | "(" <regex> ")"
```

where `<digit>` is any single digit (`0`, `1`, ..., `9`) and `<digit-sequence>` is any nonempty sequence of digits. In case of any ambiguity, the positive closure (`<regex> "+"`) has the highest precedence, which is followed by the concatenation (`<regex> <regex>`), and the alternation (`<regex> "|" <regex>`) has the lowest precedence. For example, the regular expression `1|23+` should be parsed as `(1|(2(3+)))`.

Every regular expression R *recognizes* a set of digit strings, denoted $L(R)$, which is defined as follows:

- $L(R_1R_2) = \{s \circ t | s \in L(R_1), t \in L(R_2)\}$, where R_1, R_2 are regular expressions and $\circ$ means string concatenation;
- $L(R_1|R_2) = L(R_1) \cup L(R_2)$, where R_1, R_2 are regular expressions;
- $L(R+) = \bigcup_{i=1}^{\infty} \{s^i | s \in L(R)\}$, where $s^i = s \circ s^{i-1}$ $(i > 1)$ and $s^1 = s$;
- the regular expression d (where d is any single digit) recognizes the single digit d;
- the regular expression $[s]$ (where s is any nonempty sequence of digits) recognizes any single digit appearing in s;
- parenthesizing a regular expression doesn't change the set it recognizes, i.e., $L((R)) = L(R)$.

Given a regular expression describing the whispers of the Old Gods and a whisper you heard last night, you wonder how close is the whisper you heard to the Old Gods'. Specifically, you want to know the minimum number of changes to make it recognized by the regular expression. There are three kinds of changes to the whisper:

- inserting a single digit at any position;
- removing any single digit;
- replacing any single digit with any other digit.

For example, let the regular expression be `(5|6+)[12]3+`, and the sequence of digits you've heard be `4334`. One possible way to make minimum number of changes to make the digit sequence recognized by the regular expression is `4334 -> 54334 -> 52334 -> 5233`.

Input

The input contains two lines.

The first line of the input is the regular expression R characterizing the whispers of the Old Gods, which contains at most 5 000 characters. The regular expression is syntactically correct and contains no character other than digits, `|`, `[`, `]`, `(`, `)`, and `+`.

The second line is a nonempty digital sequence with at most 10^4 digits, denoting the whisper you've heard.

Output

Print a single integer, denoting the minimum number of changes you could make to the whisper, such that it can be recognized by the regular expression.

Examples

standard input	standard output
(5|6+)[12]3+ 4334	3
[12](3+|4+)+ 2345	1
(234|25)+ 2442342523535	3

Problem C. Mean Streets of Gadgetzan

Input file: `standard input`
Output file: `standard output`

Despite being a city of opportunity, Gadgetzan is a place where gangs rule the streets.

Recently, a group of robbers broke the exterior wall of the First Bank of Gadgetzan and stole several crates of jade antiques. The bank's owner has just posted an announcement in *Gadgetzan Gazette*, offering a great quantity of arcane dust and many unused decks for the capture of perpetrators.

You, as a famous detective in Gadgetzan, have collected several testimonies from witnesses. Here is how these testimonies may look like.

"Kazakus was not at home on the day of the robbery."

"If the courier was at home, so was the Kazakus."

"If Kazakus and the courier were at home that day, then the Kazakus must not be the robber."

Generically, each testimony is in one of the following four forms:

Unconditional Affirmation This kind of testimony asserts that some proposition is true.

Unconditioanl Denial This kind of testimony asserts that some proposition is false.

Conditional Affirmation This kind of testimony asserts that some proposition is true if several propositions (called the *antecedents*) are true.

Conditional Denial This kind of testimony asserts that some proposition is false if several propositions are true.

Note that every proposition is either true or false.

Having collected these testimonies, you are going to solve the mystery. You want to find out if these testimonies conflict with each other. And if not, you want to figure out the truth of this incidence; that is, you want to identify the truth of each proposition, such that all testimonies are valid.

Input

The first line of input contains a single integer n, m ($1 \leq n, m \leq 10^6$), denoting the number of testimonies, and the number of propositions that may involve in the testimonies. The propositions are numbered 1 through m.

Then follow n lines, specifying these testimonies. Each line represents a testimony and is in the following format.

Unconditional Affirmation The line contains a single integer, the number of the proposition to confirm.

Unconditioanl Denial The line contains a single integer preceded with an exclamation mark (`!`).

Conditional Affirmation The line starts with a list of integers denoting the numbers of antecedents, followed by the `->` symbol, and ends with an integer denoting the proposition to confirm.

Conditional Denial The line starts with a list of integers denoting the numbers of antecedents, followed by the `->` symbol, and ends with an integer preceded with an exclamation mark denoting the consequent proposition to deny.

Note that there is exactly one space between the numbers and before and after the `->` symbol, but there is no space after the exclamation mark.

It is guaranteed that the antecedents of a conditional testimony are distinct and do not contain the consequent. Also, the total number of antecedents in all conditional testimonies does not exceed 10^6.

Output

If these testimonies conflict with each other, print `conflict` in a single line. Otherwise, print m characters without spacing, each being either `T` (representing true) or `F` (representing false), denoting the truth assignment to these propositions respecting the testimonies. If there are multiple possible assignments, any assignment is acceptable.

Examples

standard input	standard output
3 3 !1 2 -> 1 1 2 -> !3	FFT
6 4 1 -> 2 1 3 -> !2 1 -> 3 4 -> 2 1 2 3 4 -> !1	conflict

Problem D. Journey to Un'Goro

Input file: `standard input`
Output file: `standard output`

Recently, you've taken a trip to Un'Goro.

A small road in Un'Goro has attracted your eyes. The road consists of n steps, each colored either red or blue.

When you go from the ith step to the jth, you count the number of red steps you've stepped. You will be satisfied if the number is odd.

"What is the maximum number of pairs (i, j) $(1 \leq i \leq j \leq n)$, such that I'll be satisfied if I walk from the ith step to the jth?" you wonder. Also, how to construct all colorings such that the number of pairs is maximized?

Input

The only line contains an integer n $(1 \leq n \leq 10^5)$, indicating the number of steps of the road.

Output

Output an integer in the first line, denoting the maximum number of pairs that make you satisfied.

Each of the following several lines contains a string with length n that represents a coloring scheme, in lexicographically ascending order. The ith character is the color of the ith step, where `r` is for red and `b` for blue.

If there are more than 100 different colorings, just find the lexicographically smallest 100 colorings.

Note that in lexicographical order, `b` is ordered before `r`.

Examples

standard input	standard output
1	1 r
2	2 br rb rr
3	4 brb rbr rrr

Problem E. Knights of the Frozen Throne

Input file: `standard input`
Output file: `standard output`

Knights of the Frozen Throne is a famous online game developed and published by ICPC Entertainment, Inc.

A single server called *Zeus Super Computer* (ZSC) handles all requests from all players of the game. Originally, all players' data is stored in ZSC's memory so that the server can respond to a request without querying the database. However, this design hardly scales as the game becomes more and more popular.

As a great engineer in ICPC Entertainment, Tom wants to help the company save costs by using an awesome database with a very large capacity. After some investigation, Tom discovers the behavior patterns of all players, and he can accurately predict when a player will send a request. Now it's time to deploy his optimization design: time-based caching.

To introduce the time-based caching policy, instead of holding all players' data in memory all the time, a player's data is retained in memory only for a while, and ZSC will drop this player's data if he doesn't send any request for some time. Specifically, when the request from one player comes,

- if the player's data is not in memory, ZSC loads his data into memory and records the *last request time* of the player;
- if the player's data is already in memory, ZSC responds to this request immediately, then updates the *last request time* of the player.

Also, ZSC would drop a player's data from memory if X seconds (X is a positive integer parameter to be determined) have elapsed since his last request. In particular, if a request is sent exactly X seconds after the player's *last request time*, ZSC does have to load the player's data into memory. Initially, ZSC's memory is empty.

Tom wants you to help him determine the best positive integer X such that the total cost is minimized.

As for the cost of CPU and memory, it costs a dollars for ZSC to hold one player's data in memory for one second. ZSC's CPU is pretty powerful, and it costs $i \cdot b_i$ if i players' data is loaded in one second.

Input

The input starts with a line of a single integer m ($1 \le m \le 10^5$), indicating the number of game players.

Then follow m lines, specifying the predicted requests of the players. Each of them begins with a single integer k ($1 \le k \le 5 \times 10^5$), denoting the number of requests player i will send. The remaining k integers $p_1, p_2, \cdots, p_k$ ($1 \le p_1 \le p_2 \le \cdots \le p_k \le 10^9$) are the times(in second) predicted by Tom that the player will send request to ZSC. It is guaranteed that the total number of requests sent by all players does not exceed 5×10^5.

The next line contains an integer a ($1 \le a \le 10^4$), the cost for ZSC to hold one player's data in memory for one second.

The last line contains m integers $b_1, b_2, \cdots, b_m$ ($1 \le b_i \le 10^9$), the cost of loading one player's data if i players' data is loaded in one second.

Output

Print two integers in the first line, denoting the minimum total cost and the number of different positive integers X that can achieve this minimum cost. Then list all these values of X in increasing order in the second line.

It can be proved that there are finitely many different positive integers X, such that the total cost is minimized.

Examples

standard input	standard output
4 3 1 4 5 2 2 4 4 2 4 7 8 6 1 3 5 7 8 10 1 2 4 6 5	53 2 3 4
3 3 1 1 4 5 1 4 4 5 7 3 3 4 5 2 6 4 3	55 1 1

Note

The total cost of different values of X is shown in Figure 2.

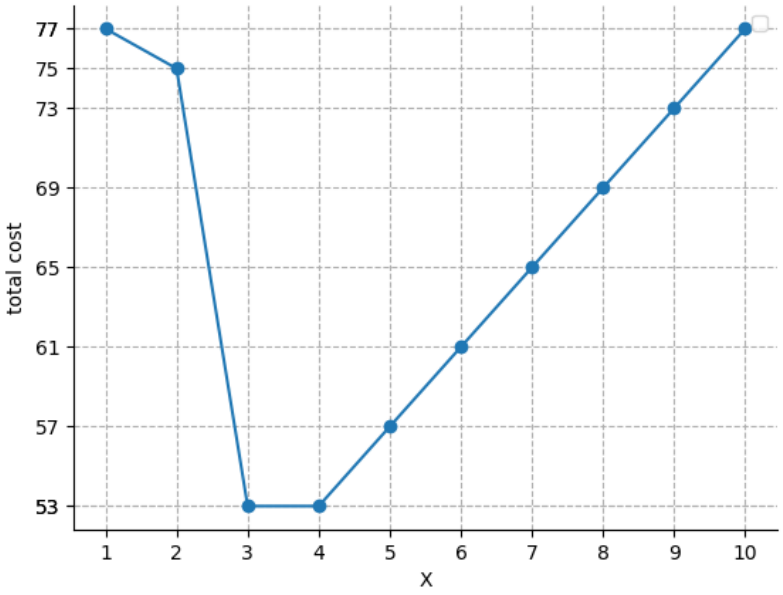

Figure 2: The first sample data.

Problem F. Kobolds and Catacombs

Input file: `standard input`
Output file: `standard output`

Kobolds are rat-like, candle-loving cave folk, digging deep beneath the surface for millennia. Today, they gather together in a queue to explore yet another tunnel in their catacombs!

But just before the glorious movement initiates, they have to arrange themselves in non-descending heights. The shortest is always the leader digging small holes, and the followers swelling it.

The kobolds are hyperactive; they like to move here and there. To make the arrangement easier, they decide to group themselves into consecutive groups first, then reorder in each group.

What's the maximum number of consecutive groups they can be partitioned into, such that after reordering the kobolds in each group in non-descending order, the entire queue is non-descending?

For example, given a queue of kobolds of heights `[1, 3, 2, 7, 4]`, we can group them into three consecutive groups (`[1] [3, 2] [7, 4]`), such that after reordering each group, the entire queue can be non-descending.

Input

The first line of the input contains a single integer n ($1 \le n \le 10^6$), denoting the number of kobolds.

The second line contains n integers $a_1, a_2, \ldots, a_n$ ($1 \le a_i \le 10^9$), representing the heights of the kobolds in the queue.

Output

Print a single integer, denoting the maximum number of groups.

Example

standard input	standard output
5 1 3 2 7 4	3

Problem G. The Witchwood

Input file: `standard input`
Output file: `standard output`

Shenyang's night fair culture is developed very well. Every time Bob comes to Shenyang, he will definitely go to a night fair called *The Witchwood*. There are n snack stalls in *The Witchwood*, the ith of which gives him a_i pleasure.

Bob's stomach allows him to eat k snack stalls at most. So Bob wants to know the maximum pleasure he can get after visiting the night market.

Input

The first line of input contains two integers n ($1 \le n \le 1000$) and k ($1 \le k \le n$), indicating the number of snack stalls and the capacity of Bob's stomach.

The second line of input contains n integers $a_1, a_2, \ldots, a_n$ ($1 \le a_i \le 10^9$), the ith of which indicates the pleasure of the ith snack stall.

Output

Print one integer denoting the maximum pleasure Bob can get.

Example

standard input	standard output
5 2 9 8 10 2 4	19

Problem H. The Boomsday Project

Input file: standard input
Output file: standard output

Aloha is a poor guy who likes riding a bike. But he is too poor to afford a bicycle.

Recently, the Boom's bike-sharing service entered his school. It only costs r yuan for every single rent of a bicycle. From then on, Aloha could rent a bike and enjoy the speed and passion of riding.

The Boom's company has launched a promotion program named the *Boomsday Project*. In this program, users can buy discount cards with several free rents and a valid period. For example, after purchasing a discount card with k free rents and d days of valid time, one doesn't need to pay extra fees for the following k rents in the next d days. Note that, no matter when the card is bought on day t, it will always expire at the end of day $t+d-1$. Also, any new discount card overrides the previous one even if it is still valid, i.e., the remaining free rents of the previous discount card are voided immediately when the new discount card is purchased.

There are n different types of discount cards in the *Boomsday Project*. A card of ith type with k_i free rents and a valid period of d_i days costs c_i yuan. One can buy any type of discount card unlimited times.

Given the rental records, Aloha wants to know the minimum money he has to pay, provided that he takes the best strategy.

Input

The first line of input contains three integers n, m, r ($1 \le n \le 500, 1 \le m \le 10^5, 1 \le r \le 10^9$), denoting the number of discount card types, the number of rental records, and the price of a single rent.

Then follow n lines, specifying the n types of discount cards. The ith line contains three integers d_i, k_i, c_i ($1 \le d_i, k_i, c_i \le 10^9$), denoting the valid period, the number of free rents, and the price of a discount card of type i.

The last m lines describe Aloha's rental records. The ith of them contains two integers p_i, q_i ($0 \le p_i \le 10^9, 0 \le q_i \le 3 \times 10^5, \sum_{i=1}^{m} q_i \le 3 \times 10^5$), representing that Aloha would rent bikes q_i times on Day p_i. It is guaranteed that $p_1, p_2, \cdots, p_m$ are distinct.

Output

Print one integer in a line, denoting the minimum money Aloha has to pay if he adopts the best strategy.

Examples

standard input	standard output
2 1 10 1 3 12 1 2 9 1 10	42
2 4 10 1 3 12 1 2 9 1 3 2 3 3 3 4 1	45

Problem I. Rise of Shadows

Input file: `standard input`
Output file: `standard output`

Azeroth is a world full of fantasy. In Azeroth, there are H hours in a day and M minutes in an hour.

You found a clock made from Azeroth. The clock has two hands — the hour hand and the minute hand. The two hands point to the same direction at the start of a day. Either hand rotates at a constant speed. The hour hand goes around a full circle in H hours and the minute hand goes around a full circle in M minutes. Surprisingly, it is night in Azeroth if and only if the angle between the two hands is less than or equal to α.

Now you're wondering, given $\alpha = \frac{2\pi A}{HM}$, how many integral moments (i.e., integer minutes since the start of the day) are there, such that the angle between the two hands is less than or equal to α.

Input

The only line of the input contains three integers H, M ($2 \leq H, M \leq 10^9$) and A ($0 \leq A \leq \frac{HM}{2}$), representing the number of hours in a day and the number of minutes in an hour, and the limit of the angle in radians, respectively.

Output

Print an integer representing the answer.

Examples

standard input	standard output
5 5 4	9
3 5 1	3

Problem J. Descent of Dragons

Input file: `standard input`
Output file: `standard output`

Hey, look, what's that? It's dragons in dragons' descent!

There are n dragons in a row, indexed 1 through n. Every dragon has a level, and initially, every dragon is of level 0.

You are a hero of the league of explorers. And your task is to counter the attack of the league of evil.

You are training the dragons in peacetime. When the villains launched a battle, you have to select the best dragon to defend.

Specifically, you have to process q events in order. Each event has one of the following types.

Training Given l, r, x, for all dragons of level x with indices between l to r inclusive, increase their levels to $x + 1$;

Defense Given l, r, find the maximum level of dragons with indices between l to r inclusive.

Input

The first line contains two integers n, q ($1 \le n, q \le 5 \times 10^5$), representing the length of the sequence and the number of queries.

The next q lines contain the queries in order, one per line. Each line may either contain four integers $1, l, r, x$ ($1 \le l \le r \le n, 0 \le x \le 5 \times 10^5$), denoting a training event; or three integers $2, l, r$ ($1 \le l \le r \le n$), denoting a defense event. It is guaranteed that there is at least one defense event in the input.

Output

For each defense event, print the result in a line.

Example

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

Problem K. Scholomance Academy

Input file: `standard input`
Output file: `standard output`

As a student of the Scholomance Academy, you are studying a course called *Machine Learning*. You are currently working on your course project: training a binary classifier.

A binary classifier is an algorithm that predicts the classes of instances, which may be positive (+) or negative (-). A typical binary classifier consists of a scoring function S that gives a score for every instance and a threshold θ that determines the category. Specifically, if the score of an instance $S(x) \geq \theta$, then the instance x is classified as positive; otherwise, it is classified as negative. Clearly, choosing different thresholds may yield different classifiers.

Of course, a binary classifier may have misclassification: it could either classify a positive instance as negative (false negative) or classify a negative instance as positive (false positive).

	Predicted class	
Actual class	Positive	Negative
Positive	True positive (TP)	False negative (FN)
Negative	False positive (FP)	True negative (TN)

Table 1: Predicted classes and actual classes.

Given a dataset and a classifier, we may define the true positive rate (TPR) and the false positive rate (FPR) as follows:

$$\mathrm{TPR} = \frac{\#TP}{\#TP + \#FN}, \quad \mathrm{FPR} = \frac{\#FP}{\#TN + \#FP}$$

where $\#TP$ is the number of true positives in the dataset; $\#FP, \#TN, \#FN$ are defined likewise.

Now you have trained a scoring function, and you want to evaluate the performance of your classifier. The classifier may exhibit different TPR and FPR if we change the threshold θ. Let $\mathrm{TPR}(\theta), \mathrm{FPR}(\theta)$ be the TPR, FPR when the threshold is θ, define the *area under curve* (AUC) as

$$\mathrm{AUC} = \int_0^1 \max_{\theta \in \mathbb{R}} \{\mathrm{TPR}(\theta) | \mathrm{FPR}(\theta) \leq r\} dr$$

where the integrand, called *receiver operating characteristic* (ROC), means the maximum possible of TPR given that $\mathrm{FPR} \leq r$.

Given the actual classes and predicted scores of the instances in a dataset, can you compute the AUC of your classifier?

For example, consider the third test data. If we set threshold $\theta = 30$, there are 3 true positives, 2 false positives, 2 true negatives, and 1 false negative; hence, $\mathrm{TPR}(30) = 0.75$ and $\mathrm{FPR}(30) = 0.5$. Also, as θ varies, we may plot the ROC curve and compute the AUC accordingly, as shown in Figure 3.

Input

The first line contains a single integer n ($2 \leq n \leq 10^6$), the number of instances in the dataset. Then follow n lines, each line containing a character $c \in \{+, -\}$ and an integer s ($1 \leq s \leq 10^9$), denoting the actual class and the predicted score of an instance.

It is guaranteed that there is at least one instance of either class.

Output

Print the AUC of your classifier within an absolute error of no more than 10^{-9}.

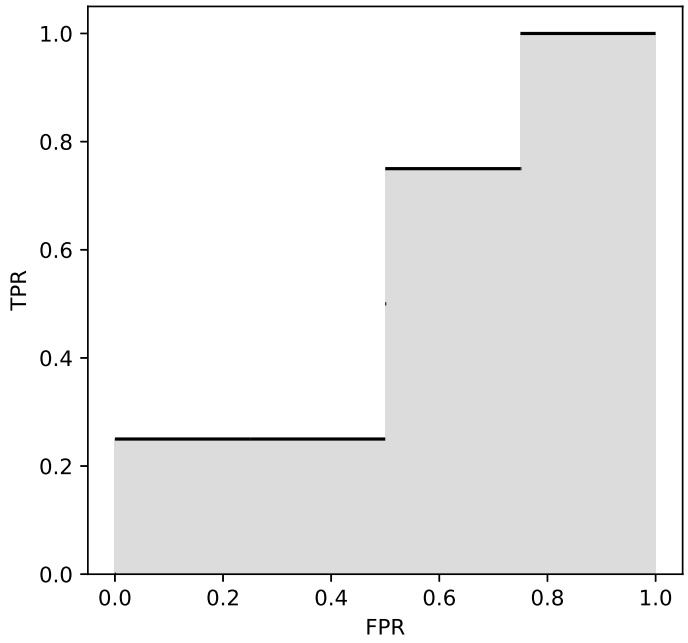

Figure 3: ROC and AUC of the third sample data.

Examples

standard input	standard output
3 + 2 - 3 - 1	0.5
6 + 7 - 2 - 5 + 4 - 2 + 6	0.888888888888889
8 + 34 + 33 + 26 - 34 - 38 + 39 - 7 - 27	0.5625

Problem L. Forged in the Barrens

Input file: standard input
Output file: standard output

Beacon towers are built throughout the Barrens. There was once a time when there were n beacon towers built from west to east for defending against the invaders. The altitude of the i-th beacon tower, based on historical records, is a_i.

The defenders divide strategically all beacon towers into k parts where each part contains several, but at least one, consecutive beacon towers. The scale of an individual part is given by the difference between the highest and the lowest altitudes of beacon towers, and the most sensible partition maximizes the sum of scales of all parts.

As a historian, you are dying to know the maximum sums of scales for every $k = 1, 2, \ldots, n$.

Input

The first line contains an integer n ($1 \le n \le 2 \times 10^5$), denoting the number of beacon towers throughout the Barrens.

The second line contains n integers $a_1, a_2, \ldots, a_n$ ($1 \le a_i \le 10^9$) is the altitudes of the beacon towers in order.

Output

Output n lines, the ith of which contains an integer indicating the maximum sum for $k = i$.

Examples

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

Problem M. United in Stormwind

Input file: standard input
Output file: standard output

The adventurers gather together in the majestic capital city, Stormwind, for a referendum about the future of their alliance. The referendum contains m questions, and each question has two options, `A` and `B`.

After collecting the results of the referendum, the alliance received n survey results. When sorting out the results, Alice comes up with a special idea. She thinks that a nonempty subset of questions is *discriminative* if there are at least k pairs of results different in at least one of the questions in the set.

She wants to know the number of different discriminative subsets of questions. Can you help Alice solve this problem?

Input

The first line contains three integers n, m, k ($1 \le n \le 2 \times 10^5, 1 \le m \le 20, 1 \le k \le \frac{n(n-1)}{2}$), as specified in the problem statement.

Each of the following n lines contains a string of length m, which contains only `A` and `B`, indicating the answer to each question in the referendum results.

Output

Output a single integer that represents the number of different discriminative subsets of questions.

Examples

standard input	standard output
2 2 1 AA BB	3
2 2 1 AA AB	2