The 2024 CCPC Shandong Invitational Contest and Provincial Collegiate Programming Contest

Contest Session

May 26, 2024

Problem List

A	Printer
B	Triangle
C	Colorful Segments 2
D	Hero of the Kingdom
E	Sensors
F	Divide the Sequence
G	Cosmic Travel
H	Stop the Castle
I	Left Shifting
J	Colorful Spanning Tree
K	Matrix
L	Intersection of Paths
M	Palindromic Polygon

This problem set should contain 13 (thirteen) problems on 17 (seventeen) numbered pages. Please inform a runner immediately if something is missing from your problem set.

Prepared by SUA Programming Contest Problem Setter Team.
https://sua.ac/

Hosted by

SHANDONG UNIVERSITY 1901

CIIIC
国家信息通信国际创新园
CHINA INTERNATIONAL ICT INNOVATION CLUSTER

Problem Set Prepared by

It’s against the rules to open non-contest websites during the contest.
If you’re interested (which is our pleasure),
please scan the QR code only after the contest.

Problem A. Printer

Judges from the SUA Programming Contest Problem Setter Team are printing problem sets for the upcoming 2024 CCPC Shandong Invitational Contest and CCPC Shandong Provincial Collegiate Programming Contest.

There are n printers in the printing shop. The i-th printer can produce one copy of the problem set every t_i seconds. However, each time after producing l_i copies from the i-th printer, it must halt for w_i seconds to avoid overheating. That is to say, the i-th printer will repeat the following working schedule: Work continuously for $t_i \times l_i$ seconds, then halt for w_i seconds.

The judges will use all printers at the same time. Calculate the minimum number of seconds needed to produce at least k copies of the problem set.

Input

There are multiple test cases. The first line of the input contains an integer T ($1 \le T \le 100$) indicating the number of test cases. For each test case:

The first line contains two integers n and k ($1 \le n \le 100$, $1 \le k \le 10^9$) indicating the number of printers and the number of copies needed.

For the following n lines, the i-th line contains three integers t_i, l_i, and w_i ($1 \le t_i, l_i, w_i \le 10^9$). Their meanings are described above.

Output

For each test case, output one line containing one integer indicating the minimum number of seconds needed.

Example

standard input	standard output
2 3 15 3 4 5 5 7 2 1 2 20 1 100 1 1 100	25 10000

Note

For the first sample test case, in 25 seconds, the first printer can produce 6 copies, the second printer can produce 5 copies, while the third printer can produce 4 copies. So they can produce $6 + 5 + 4 = 15$ copies in total.

Problem B. Triangle

Given n strings $S_1, S_2, \cdots, S_n$ consisting of lower-cased English letters, we say three strings S_a, S_b and S_c form a triangle, if all the following constraints are satisfied:

- $S_a + S_b > S_c$ or $S_b + S_a > S_c$.
- $S_a + S_c > S_b$ or $S_c + S_a > S_b$.
- $S_b + S_c > S_a$ or $S_c + S_b > S_a$.

Here $+$ is the string concatenation operation and strings are compared by lexicographic order. For example, `ba`, `cb` and `cbaa` forms a triangle, because:

- `cb` + `ba` = `cbba` > `cbaa`.
- `cbaa` + `ba` = `cbaaba` > `cb`.
- `cb` + `cbaa` = `cbcbaa` > `ba`.

Count the number of integer tuples (a, b, c) such that $1 \le a < b < c \le n$ and S_a, S_b, S_c forms a triangle.

Input

There are multiple test cases. The first line of the input contains an integer T indicating the number of test cases. For each test case:

The first line contains an integer n ($1 \le n \le 3 \times 10^5$) indicating the number of strings.

For the following n lines, the i-th line contains a string S_i ($1 \le |S_i| \le 3 \times 10^5$) consisting of lower-cased English letters.

It's guaranteed that the total length of the strings in a single test case does not exceed 3×10^5, and the total length of strings of all test cases does not exceed 10^6.

Output

For each test case, output one line containing one integer indicating the number of valid tuples.

Example

standard input	standard output
3 6 cbaa cb cb cbaa ba ba 3 sdcpc sd cpc 1 ccpc	16 0 0

Problem C. Colorful Segments 2

Consider n segments on the number axis, where the left endpoint of the i-th segment is l_i and the right endpoint is r_i. You need to paint each segment into one of the k colors, so that for any two segments with the same color they do not overlap.

Calculate the number of ways to color the segments.

We say segment i overlaps with segment j, if there exists a real number x satisfying both $l_i \le x \le r_i$ and $l_j \le x \le r_j$.

We say two ways of coloring the segments are different, if there exists one segment which has different colors in the two ways.

Input

There are multiple test cases. The first line of the input contains an integer T indicating the number of test cases. For each test case:

The first line contains two integers n and k ($1 \le n \le 5 \times 10^5$, $1 \le k \le 10^9$) indicating the number of segments and the number of colors.

For the following n lines, the i-th line contains two integers l_i and r_i ($1 \le l_i \le r_i \le 10^9$) indicating the left and right endpoints of the i-th segment.

It's guaranteed that the sum of n of all test cases will not exceed 5×10^5.

Output

For each test case output one line containing one integer indicating the answer. As the answer might be large, output it modulo 998244353.

Example

standard input	standard output
2 4 3 4 7 3 4 5 8 1 3 2 1000 100 200 300 400	24 1000000

Note

Let c_i be the color of the i-th segment.

For the first sample test case, one valid way to color the segments is $c_1 = 1$, $c_2 = 3$, $c_3 = 3$, and $c_4 = 1$. Because the 1-st and the 4-th segments do not overlap, also the 2-nd and the 3-rd segments do not overlap.

However, $c_1 = 1$, $c_2 = 2$, $c_3 = 1$, and $c_4 = 3$ is not a valid way. Because the 1-st and the 3-rd segments overlap with each other and they can't have the same color.

Problem D. Hero of the Kingdom

Hero of the Kingdom is a point-and-click adventure game, in which the protagonist sets off on a dangerous journey to save his/her father and becomes the hero of the kingdom.

Money in the game is called *gold* and can be used to buy various supplies or even complete certain quests. As the saying goes, one is never too rich to earn, our talented player BaoBao has just found a way to become rich. In the game, there is a mill where the owner sells flour for p gold per bag. There is also a tavern where the barman buys flour at the price of q gold per bag ($q > p$). It's obvious that BaoBao can pocket the difference, but moving between the two places and clicking on the buttons for buying and selling also takes time.

More precisely, if BaoBao buys x bags of flour from the mill in one shot, he will have to spend $(ax + b)$ seconds and px gold; If BaoBao sells x bags of flour to the tavern in one shot, he will have to spend $(cx + d)$ seconds but then earn qx gold. BaoBao currently has m gold, but as it's time for him to go to bed, he can only play the game for at most t seconds. Calculate the maximum number of gold he can have when he finishes playing the game.

Input

There are multiple test cases. The first line of the input contains an integer T ($1 \le T \le 500$) indicating the number of test cases. For each test case:

The first line contains three integers p, a, and b ($1 \le p, a \le 10^9$, $0 \le b \le 10^9$).

The second line contains three integers q, c, and d ($p < q \le 10^9$, $1 \le c \le 10^9$, $0 \le d \le 10^9$).

The third line contains two integers m and t ($1 \le m, t \le 10^9$).

Output

For each test case, output one line containing one integer, indicating the maximum number of gold BaoBao can have after at most t seconds.

Example

standard input	standard output
3 5 2 3 8 1 5 14 36 5 2 0 8 1 3 17 6 100 1 0 10000 1 0 99 100000	32 20 99

Note

For the first sample test case, one of the optimal strategy is:

- BaoBao first buys 2 bags of flour from the mill. It takes $2 \times 2 + 3 = 7$ seconds and costs $5 \times 2 = 10$ gold. Then he sells all flour to the tavern. It takes another $1 \times 2 + 5 = 7$ seconds but BaoBao earns $8 \times 2 = 16$ gold. BaoBao now has $14 - 10 + 16 = 20$ gold, and there are $36 - 7 - 7 = 22$ seconds remaining.
- BaoBao then buys 4 bags of flour from the mill. It takes $2 \times 4 + 3 = 11$ seconds and costs $5 \times 4 = 20$ gold. Then he sells all flour to the tavern. It takes another $1 \times 4 + 5 = 9$ seconds but BaoBao earns $8 \times 4 = 32$ gold. BaoBao now has $20 - 20 + 32 = 32$ gold, and there are $22 - 11 - 9 = 2$ seconds remaining.
- Now BaoBao doesn't have time to buy or sell flour. So the answer is 32.

For the second sample test case, BaoBao only has time to buy and sell one bag of flour. So the answer is $17 - 5 + 8 = 20$.

For the third sample test case, BaoBao doesn't have enough gold to buy flour. So the answer is 99.

Problem E. Sensors

There are n red balls arranged in a row, numbered from 0 to $(n-1)$ (both inclusive) from left to right. We are going to perform n operations, where the i-th operation will color the a_i-th ball to blue. After all operations all the balls will become blue.

There are m sensors, numbered from 1 to m (both inclusive), monitoring the color of the balls. The i-th sensor will become active, if there is exactly one red ball among all the balls numbered from l_i to r_i (both inclusive); Otherwise the sensor remains inactive.

Determine which sensors are active after each operation.

More precisely, let's say there are k_i active sensors after the i-th operation and their indices are $s_{i,1}, s_{i,2}, \cdots, s_{i,k_i}$. For each $0 \le i \le n$, output $v_i = \sum\limits_{j=1}^{k_i} s_{i,j}^2$. Specifically, define $v_0 = \sum\limits_{j=1}^{k_0} s_{0,j}^2$, where k_0 is the number of active sensors before the first operation, and the indices of the active sensors are $s_{0,1}, s_{0,2}, \cdots, s_{0,k_0}$.

Input

There are multiple test cases. The first line of the input contains an integer T indicating the number of test cases. For each test case:

The first line contains two integers n and m $(1 \le n, m \le 5 \times 10^5)$ indicating the number of balls and the number of sensors.

For the following m lines, the i-th line contains two integers l_i and r_i $(0 \le l_i \le r_i < n)$ indicating the detection range of the i-th sensor.

The next line contains n integers $a'_1, a'_2, \cdots, a'_n$ $(0 \le a'_i < n)$ where a'_i indicates the **encoded** i-th operation. The real value of a_i is equal to $(a'_i + v_{i-1}) \bmod n$, where v_{i-1} is the answer after the $(i-1)$-th operation, defined in the description above. With the encoded operations, you're forced to calculate the answer to each operation before processing the next one. It's guaranteed that each a_i is distinct after decoding.

It's guaranteed that neither the sum of n nor the sum of m of all test cases will exceed 5×10^5.

Output

For each test case, output one line containing $(n+1)$ integers $v_0, v_1, \cdots, v_n$ separated by a space. The meaning of v_i is defined in the description above.

Example

standard input	standard output
3 5 4 2 4 2 3 3 3 0 2 3 2 4 2 0 2 1 1 1 1 0 2 1 0 1 0 0	9 13 29 17 16 0 1 1 0 0 1 0

Note

For the first sample test case:

- Before the first operation, only sensor 3 is active, so $v_0 = 3^2 = 9$.
- For the 1-st operation, the real $a_1 = (3 + 9) \bmod 5 = 2$. After this operation, sensors 2 and 3 are active, so $v_1 = 2^2 + 3^2 = 13$.
- For the 2-nd operation, the real $a_2 = (2 + 13) \bmod 5 = 0$. After this operation, sensors 2, 3 and 4 are active, so $v_2 = 2^2 + 3^2 + 4^2 = 29$.
- For the 3-rd operation, the real $a_3 = (4 + 29) \bmod 5 = 3$. After this operation, sensors 1 and 4 are active, so $v_3 = 1^2 + 4^2 = 17$.
- For the 4-th operation, the real $a_4 = (2 + 17) \bmod 5 = 4$. After this operation, only sensor 4 is active, so $v_4 = 4^2 = 16$.
- For the 5-th operation, the real $a_5 = (0 + 16) \bmod 5 = 1$. After this operation, no sensor is active, so $v_5 = 0$.

Problem F. Divide the Sequence

Given an integer sequence $a_1, a_2, \cdots, a_n$ of length n, divide the sequence into k continuous non-empty subarrays such that each element belongs to exactly one subarray. Let s_i be the sum of elements in the i-th subarray from left to right, for each $1 \le k \le n$, calculate the maximum value of

$$\sum_{i=1}^{k} i \times s_i$$

More formally, for each $1 \le k \le n$, let $r_0 = 0$ and $r_k = n$, you need to find $(k-1)$ integers $r_1, r_2, \cdots, r_{k-1}$ such that $r_0 < r_1 < r_2 < \cdots < r_{k-1} < r_k$ and maximize

$$\sum_{i=1}^{k} i \times (\sum_{j=r_{i-1}+1}^{r_i} a_j)$$

Input

There are multiple test cases. The first line of the input contains an integer T indicating the number of test cases. For each test case:

The first line contains one integer n ($1 \le n \le 5 \times 10^5$) indicating the length of the sequence.

The second line contains n integers $a_1, a_2, \cdots, a_n$ ($-10^6 \le a_i \le 10^6$) indicating the sequence.

It's guaranteed that the sum of n of all test cases will not exceed 5×10^5.

Output

For each test case, output one line containing n integers $v_1, v_2, \cdots, v_n$ separated by a space, where v_i is the answer for $k = i$.

Example

standard input	standard output
2 6 1 3 -4 5 -1 -2 1 100	2 4 5 3 1 -2 100

Note

For the first sample test case, consider $k = 3$, we can divide the sequence into $\{\{1\}, \{3, -4\}, \{5, -1, -2\}\}$. The answer is $1 \times 1 + 2 \times (3 - 4) + 3 \times (5 - 1 - 2) = 5$.

Problem G. Cosmic Travel

BaoBao is a cosmic traveler, shuttling between an infinite number of parallel universes. Each universe is numbered with an integer starting from 0.

There are n magic apples in each universe. Although these universes have many similarities, there are still slight differences between them. The magic power of the i-th apple in the j-th universe is $a_i \oplus j$. Here $\oplus$ denotes bitwise exclusive or operation.

BaoBao is very indecisive, so he prepared q traveling plans. Each traveling plan can be described by three integers l, r and k, which means BaoBao will travel to each universe numbered from l to r (both inclusive), and collect the apple with the k-th smallest magic power among the n apples in each universe.

For each traveling plan, please help BaoBao calculate the sum of the magic power of the apples he collects. Note that the traveling plan does not really remove the apple from each universe. That is, each query is independent.

Input

There is only one test case in each test file.

The first line contains two integers n and q ($1 \le n, q \le 10^5$) indicating the number of apples in each universe and the number of plans.

The second line contains n integers $a_1, a_2, \cdots, a_n$ ($0 \le a_i < 2^{60}$).

For the following q lines, the i-th line contains three integers l_i, r_i and k_i ($0 \le l_i \le r_i < 2^{60}$, $1 \le k_i \le n$) indicating the i-th traveling plan.

Output

For each traveling plan output one line containing one integer indicating the answer for that plan. As the answer might be large, output it modulo 998244353.

Example

standard input	standard output
8 3 2 0 2 4 0 5 2 6 1 1 6 2 7 5 0 1048575 4	4 23 720895450

Problem H. Stop the Castle

There are n castles and m obstacles on a chessboard with 10^9 rows and 10^9 columns. Each castle or obstacle occupies exactly one cell and all occupied cells are distinct. Two castles can attack each other, if they're on the same row or the same column, and there are no obstacles or other castles between them. More formally, let (i, j) be the cell on the i-th row and the j-th column. Two castles positioned at (i_1, j_1) and (i_2, j_2) can attack each other, if one of the following conditions is true:

- $i_1 = i_2$ and for all $\min(j_1, j_2) < j < \max(j_1, j_2)$, there is no obstacle or castle positioned at (i_1, j).
- $j_1 = j_2$ and for all $\min(i_1, i_2) < i < \max(i_1, i_2)$, there is no obstacle or castle positioned at (i, j_1).

Find a way to place the minimum number of additional obstacles onto the chessboard, so that no two castles can attack each other. Note that the additional obstacles cannot be placed in an already occupied cell.

Input

There are multiple test cases. The first line of the input contains an integer T indicating the number of test cases. For each test case:

The first line contains an integer n ($2 \le n \le 200$) indicating the number of castles.

For the following n lines, the i-th line contains two integers r_i and c_i ($1 \le r_i, c_i \le 10^9$) indicating that the i-th castle is located on the r_i-th row and the c_i-th column.

The next line contains an integer m ($0 \le m \le 200$) indicating the number of obstacles.

For the following m lines, the i-th line contains two integers r'_i and c'_i ($1 \le r'_i, c'_i \le 10^9$) indicating that the i-th obstacle is located on the r'_i-th row and the c'_i-th column.

It's guaranteed that the occupied cells are distinct. It's also guaranteed that neither the sum of n nor the sum of m of all test cases will exceed 400.

Output

For each test case:

If it is possible to stop the castles from attacking each other, first output one line containing one integer k, indicating the minimum number of additional obstacles needed. Then output k lines where the i-th line contains two integers x_i and y_i ($1 \le x_i, y_i \le 10^9$) separated by a space, indicating that you're going to place the i-th additional obstacle in cell (x_i, y_i). If there are multiple valid answers, you can output any of them.

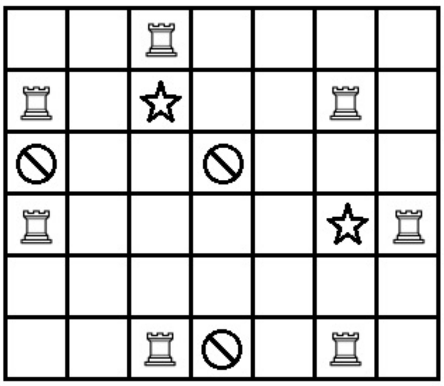
If it is impossible to stop the castles from attacking each other, just output `-1` in one line.

Example

standard input	standard output
4 7 1 3 6 6 4 7 2 1 6 3 4 1 2 6 3 3 4 6 4 3 1 2 1 1 2 2 0 3 1 1 1 3 3 3 1 1 2 3 1 1 1 3 2 3 0	2 2 3 4 6 0 1 2 3 -1

Note

The first sample test case is shown below. We only need to add 2 additional obstacles (marked as stars), one located at $(2, 3)$ while the other located at $(4, 6)$.



For the second sample test case, the only two castles do not lie on the same row or the same column, so no obstacle is needed.

Problem I. Left Shifting

We say a string is beautiful, if its first character is the same as its last character.

Given a string $S = s_0 s_1 \cdots s_{n-1}$ of length n, let $f(S, d)$ be the string obtained by shifting S to the left d times. That is $f(S, d) = s_{(d+0) \bmod n} s_{(d+1) \bmod n} \cdots s_{(d+n-1) \bmod n}$. Find the smallest non-negative integer d such that $f(S, d)$ is beautiful.

Input

There are multiple test cases. The first line of the input contains an integer T indicating the number of test cases. For each test case:

The first and only line contains a string $s_0 s_1 \cdots s_{n-1}$ ($1 \le n \le 5 \times 10^5$) consisting only of lower-cased English letters.

It's guaranteed that the sum of n of all test cases will not exceed 5×10^5.

Output

For each test case, output one line containing one integer, indicating the smallest non-negative integer d such that $f(S, d)$ is beautiful. If it's impossible to find such d, output `-1` instead.

Example

standard input	standard output
4 helloccpc abcdcba x abc	3 0 0 -1

Note

For the first sample test case, $f(S, 3) =$ `loccpchel`. As its first and last characters are both `l`, it is a beautiful string. Although $f(S, 6) =$ `cpchelloc` is also beautiful, we need to answer the smallest non-negative d. So the answer is 3.

Problem J. Colorful Spanning Tree

BaoBao has many colored vertices. The colors are numbered from 1 to n (both inclusive), and there are a_i vertices of color i. As BaoBao has just learnt the minimum spanning tree problem in his algorithm class, he decides to practice it with the vertices.

Each pair of vertices is connected by an weighted edge. The weight of each edge is only related to the colors of its two endpoints. More precisely, let c_u be the color of vertex u, if an edge connects vertices u and v, its weight will be b_{c_u,c_v}.

Help BaoBao calculate the total weight of the minimum spanning tree of the graph.

Recall that a minimum spanning tree is a subset of the edges in a connected, weighted graph that connects all the vertices without any cycles and with the minimum possible total weight.

Input

There are multiple test cases. The first line of the input contains an integer T indicating the number of test cases. For each test case:

The first line contains an integer n ($1 \le n \le 10^3$) indicating the number of different colors.

The second line contains n integers $a_1, a_2, \cdots, a_n$ ($1 \le a_i \le 10^6$), where a_i is the number of vertices with color i.

For the following n lines, the i-th line contains n integers $b_{i,1}, b_{i,2}, \cdots, b_{i,n}$ ($1 \le b_{i,j} \le 10^6$) where $b_{i,j}$ is the weight of an edge connecting two vertices with color i and j. It's guaranteed that $b_{i,j} = b_{j,i}$ for all $1 \le i, j \le n$.

It's guaranteed that the sum of n over all the test cases doesn't exceed 10^3.

Output

For each test case, output one line containing one integer, indicating the total weight of the minimum spanning tree.

Example

standard input	standard output
3 3 100 1 1 1 100 2 100 100 1 2 1 100 2 3 3 100 1 1 100 1 1 5	102 5 0

Problem K. Matrix

Construct a matrix of n rows and n columns, satisfying all the following constraints:

- The elements of the matrix are integers ranges from 1 to $2n$ (both inclusive).
- Each integer from 1 to $2n$ (both inclusive) should appear at least once in the matrix.
- Let $a_{i,j}$ be the element on the i-th row and the j-th column, there exists exactly one integer quadruple (x, y, z, w) such that:
  - $1 \le x < z \le n$.
  - $1 \le y < w \le n$.
  - $a_{x,y}$, $a_{x,w}$, $a_{z,y}$, $a_{z,w}$ are pairwise different.

Input

There is only one test case in each test file.

The first and only line of the input contains one integer n ($2 \le n \le 50$) indicating the size of matrix.

Output

If it is possible to construct such a matrix, first output `Yes` in one line. Then output n lines where the i-th line contains n integers $a_{i,1}, a_{i,2}, \cdots, a_{i,n}$ ($1 \le a_{i,j} \le 2n$) separated by a space, where $a_{i,j}$ is the element on the i-th row and the j-th column. If there are multiple valid answers, you can output any of them.

If it is impossible to construct such a matrix, just output `No` in one line.

Examples

standard input	standard output
2	Yes 1 2 3 4
3	Yes 3 2 6 4 3 3 3 1 5

Problem L. Intersection of Paths

There is a tree with n vertices and $(n-1)$ edges, where the i-th edge connects vertices u_i and v_i, and has a weight of w_i.

Your task is to process q queries. The i-th query can be described as three integers a_i, b_i and k_i. This query will temporarily change the weight of the a_i-th edge to b_i. After that you should choose $2k_i$ distinct vertices $s_1, s_2, \cdots, s_{k_i}, e_1, e_2, \cdots, e_{k_i}$ and consider the k_i simple paths on the tree, where the p-th path starts from vertex s_p and ends at vertex e_p. We say an edge is good, if it is contained in all k_i paths. Maximize the total weights of good edges.

Note again that the change in the weight of each query is temporary. After each query you should change back the weight.

Input

There is only one test case in each test file.

The first line contains two integers n and q ($2 \le n \le 5 \times 10^5$, $1 \le q \le 5 \times 10^5$) indicating the number of vertices and the number of queries.

For the following $(n-1)$ lines, the i-th line contains three integers u_i, v_i and w_i ($1 \le u_i, v_i \le n$, $1 \le w_i \le 10^9$) indicating that the i-th edge connects vertices u_i and v_i, and has a weight of w_i.

For the following q lines, the i-th line contains three integers a_i, b_i and k_i ($1 \le a_i \le n-1$, $1 \le b_i \le 10^9$, $1 \le k_i \le \lfloor \frac{n}{2} \rfloor$) indicating the i-th query.

Output

For each query output one line containing one integer indicating the answer.

Example

standard input	standard output
7 3 1 2 20 2 3 10 2 4 40 4 6 10 1 5 30 5 7 10 2 100 1 5 50 2 2 100 3	160 110 20

Note

For the first query, choose $s_1 = 3$ and $e_1 = 7$.

For the second query, choose $s_1 = 4$, $s_2 = 6$, $e_1 = 7$ and $e_2 = 5$.

For the third query, choose $s_1 = 3$, $s_2 = 4$, $s_3 = 6$, $e_1 = 5$, $e_2 = 1$ and $e_3 = 7$.

Problem M. Palindromic Polygon

There is a convex polygon with n vertices. Vertices are numbered from 1 to n (both inclusive) in counter-clockwise order, and vertex i has a value of $f(i)$.

We say a subset of the vertices is palindromic, if their values constitute a palindrome in counter-clockwise order. More formally, let's say the subset contains k vertices $v_0, v_1, \cdots, v_{k-1}$ in counter-clockwise order. There should exist an integer d such that $0 \le d < k$, and for all $0 \le i < k$ we have $f(v_{(d+i) \bmod k}) = f(v_{(d-1-i) \bmod k})$.

Among all palindromic subsets, find the one whose convex hull has the largest size.

Input

There are multiple test cases. The first line of the input contains an integer T indicating the number of test cases. For each test case:

The first line contains an integer n ($3 \le n \le 500$) indicating the number of vertices of the convex polygon.

The second line contains n integers $f(1), f(2), \cdots, f(n)$ ($1 \le f(i) \le 10^9$) where $f(i)$ is the value of the i-th vertex.

For the following n lines, the i-th line contains two integers x_i and y_i ($-10^9 \le x_i, y_i \le 10^9$) indicating the coordinates of the i-th vertex. The vertices are listed in counter-clockwise order. The convex polygon is guaranteed to have positive size and no two vertices coincide. However there might be three vertices lying on the same line.

It's guaranteed that the sum of n of all test cases does not exceed 10^3.

Output

For each test case output one line containing one integer, indicating the size of the largest convex hull of a palindromic subset, multiplied by 2. It can be proven that this value is always an integer.

Example

standard input	standard output
3 8 2 4 2 4 3 4 5 3 2 3 0 6 -3 3 -3 0 -2 -3 1 -5 3 -3 4 0 3 1 2 3 0 0 1 0 0 1 3 1 1 1 0 0 1 0 0 1	84 0 1

Note

The first sample test case is illustrated below. Choose vertices 2, 4, 5, 6, 8 and consider $d = 1$, then the value sequence $\{4, 3, 4, 3, 4\}$ is a palindrome.