Problem Set

Please check that you have 12 problems that are spanned across 26 pages in total (including Korean translation and this cover page).

Problem A
Best Student
Time Limit: 1.2 Seconds

The SY School selects a best student every day. Given a list of the best students for n days, the school wants to know a student who is most often selected as the best student during a period $[S, E]$ of days from S to E. The school plans to award for the student with a gift.

Given a list of best students for n consecutive days and q queries $\{(S_1, E_1), \ldots, (S_q, E_q)\}$, write a program to find the best student to be selected most often during the period $[S_i, E_i]$ for each query (S_i, E_i).

Input
Your program is to read from standard input. The input starts with a line containing two integers, n and q, representing the number of days and the number of queries, respectively, where $1 \le n \le 100{,}000$ and $1 \le q \le 100{,}000$. Students have unique id numbers between 1 and 10^9. The next line consists of n positive integers representing n id numbers for best students, ordered from day 1 to day n. Each of the following q lines consists of two positive integers, S_i and E_i, that represent a query (S_i, E_i), where $[S_i, E_i]$ is the period of days from S_i to E_i. Note that $1 \le S_i \le E_i \le n$ for $i = 1, \ldots, q$.

Output
Your program is to write to standard output. Print exactly q lines. The i-th line should contain the id number of the student selected most often as best student during the i-th period $[S_i, E_i]$. When there are more than one such students, the program should print the largest one among their id numbers.

The following shows sample input and output for two test cases.

Sample Input 1	Output for the Sample Input 1
5 3 2 1 2 1 1 1 2 1 4 1 5	2 2 1

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

Problem B
Carrot Field
Time Limit: 1 Second

In the middle of an infinitely wide carrot field, there is a rectangular stable whose sides are parallel either to x-axis or to y-axis. Each corner of the stable has integer coordinates. A horse is tied to a pole in the lower left corner of the stable as shown in Figure B.1. The length of the leash tied to this horse is L. A carrot is planted for every two-dimensional grid point in the carrot field, and the horse can eat any carrot planted within the reach of the leash.

For example, if the size of the stable is 11×6 and the length of the leash is 9, the carrots that the horse can eat are indicated by the dots in the right of Figure B.1 below. But note that the horse and the leash cannot enter the stable, and there are no carrots planted on the sides and the interior of the stable.

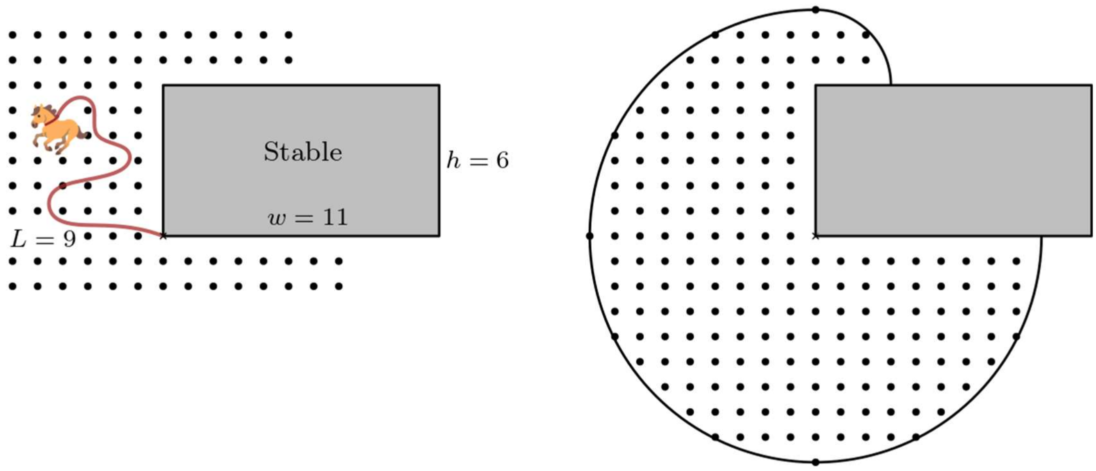


Figure B.1 The horse is tied to the lower left corner of the stable by the leash of length 9 (left). Dots represent the carrots that the horse can eat (right).

You are given three integers w, h, and L where the stable has width w and height h, and the leash has length L. Write a program to compute the maximum number of carrots a horse can eat. Note that if the distance between a grid point and the corner to which the horse is tied is exactly L, the horse can eat the carrot at the grid point.

Input
Your program is to read from standard input. The first line contains three positive integers w, h, and L ($1 \le w, h, L \le 100{,}000$) where w and h represent the width and the height of the stable, respectively, and L is the length of the leash.

Output
Your program is to write to standard output. Print exactly one line which contains the maximum number of carrots a horse can eat.

The following shows sample input and output for three test cases.

Sample Input 1	Output for the Sample Input 1
`11 6 3`	`18`

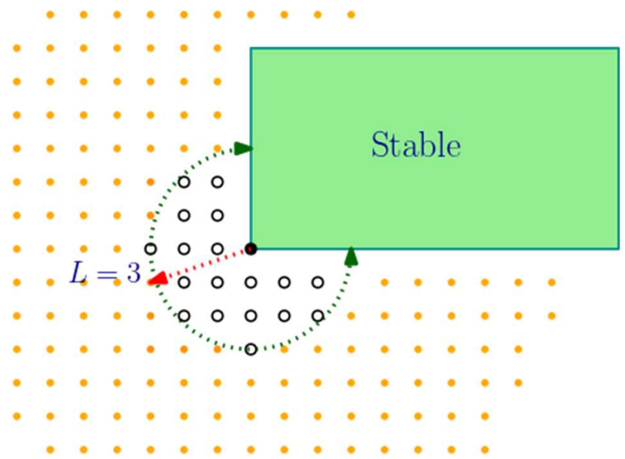


Figure B.2 The illustration for the first sample test case.

Sample Input 2	Output for the Sample Input 2
`11 6 15`	`591`

Sample Input 3	Output for the Sample Input 3
`11 6 20`	`1134`

Problem B

당근 밭

제한 시간: 1 초

무한히 넓은 당근 밭 가운데 x, y축으로 수평인 직사각형 마구간이 있다. 그림 B.1 의 왼쪽 그림처럼 마구간의 왼쪽 아래 모서리 기둥에 말이 묶여 있고, 마구간의 네 모서리는 모두 격자점에 있다. 상하좌우로 인접한 두 격자점 사이의 간격은 1이다. 말을 묶은 줄의 길이는 L로 유한하다. 그리고 당근 밭의 모든 격자점마다 하나씩 당근이 심어져 있다. 그리고 말을 묶은 줄이 닿을 수 있는 거리 안에 심어진 당근은 말이 모두 먹을 수 있다고 가정한다.

만일 마구간의 크기가 11×6이고 묶은 줄의 길이가 9일때, 말이 먹을 수 있는 당근이 그림 B.1 의 오른쪽 그림에서 점으로 표시되어 있다. 단, 말과 말을 묶은 줄은 마굿간안으로는 들어갈 수 없으며 마굿간의 경계와 내부에는 당근이 심어져 있지 않다.

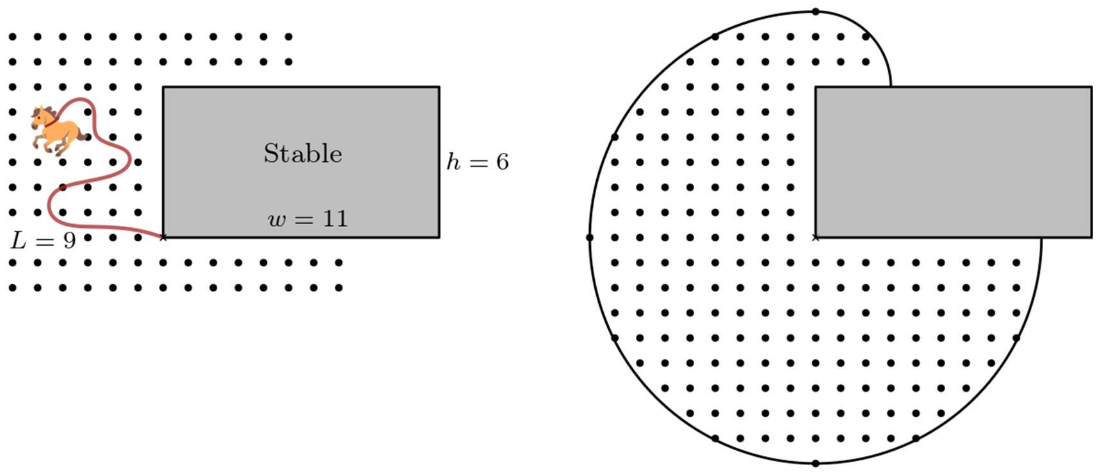


그림 B.1 (왼쪽) 마구간과 길이 9 인 줄에 묶인 말. (오른쪽) 말이 먹을 수 있는 모든 당근 (점).

마구간의 크기 $w \times h$와 말을 묶은 줄의 길이 L, 이렇게 3 개의 정수 w, h, L 이 주어졌을 때, 말이 먹을 수 있는 당근의 최대 개수를 구하는 프로그램을 작성하시오. 격자점과 말을 묶은 기둥과의 거리가 정확하게 L인 경우, 이 격자점의 당근은 말이 먹을 수 있음에 유의하자.

Input

입력은 표준입력을 사용한다. 첫 번째 줄에 마구간의 크기와 줄의 길이를 나타내는 3 개의 양의 정수 w, h, L ($1 \leq w,\ h,\ L \leq 100{,}000$)가 주어진다.

Output

출력은 표준출력을 사용한다. 주어진 조건에서 말이 먹을 수 있는 당근의 최대 개수를 첫 번째 줄에 정수로 출력한다.

다음은 세 테스트 경우에 대한 입출력 예이다.

Sample Input 1	Output for the Sample Input 1
11 6 3	18

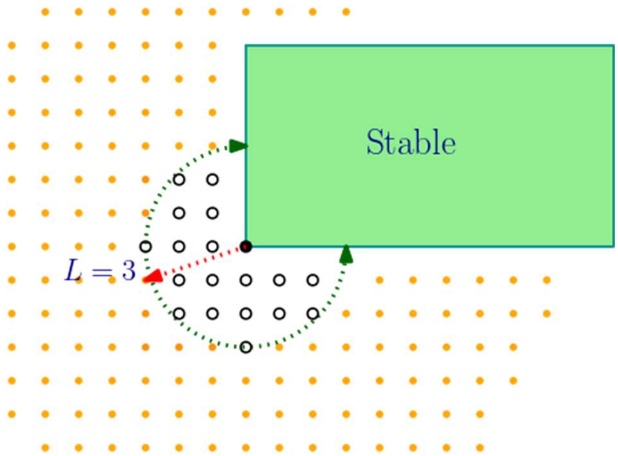


그림 B.2 첫 번째 테스트 케이스에 대한 설명

Sample Input 2	Output for the Sample Input 2
11 6 15	591

Sample Input 3	Output for the Sample Input 3
11 6 20	1134

Problem C
Colorful Tower of Hanoi
Time Limit: 1 Second

This problem is a variant of the well-known Tower of Hanoi problem.

The Tower of Hanoi problem can be described as follows. There are three rods, say, rod-1, rod-2, and rod-3. On rod-1, n disks of different sizes are stacked in decreasing order. That is, the smallest disk is at the top and the largest is at the bottom. The goal of the problem is to move all the disks stacked on rod-1 to rod-3. However, during the transfer process, only one disk can be moved from on rod to another at a time, and a disk must not be placed on top of a smaller one.

The original problem is changed as follows.

1. Disks of the same size are allowed. Therefore, in the process of moving disks, a disk can be placed on top of a disk of the same size or larger.
2. Each disk is painted in one of three colors: red, green, or blue. However, disks of the same size are painted in the same color.
3. If there are two or more disks of the same size, the relative order between them may be maintained or reversed after moving the disks, which is determined by the disk color. That is, if the disk color is red, the relative order of disks of the same size should be reversed after all disks are finally moved. If the color of the disk is blue, the relative order must be the same as the initial order. And if the color is green, the relative order after movement is not important. However, it is not necessary to satisfy the condition for the relative order in the intermediate process of moving the disks.
4. The total number of disk moves should be minimized.

Figure C.1 shows an example which satisfies the conditions of relative order depending on the colors after moving all the disks from rod-1 to rod-3.

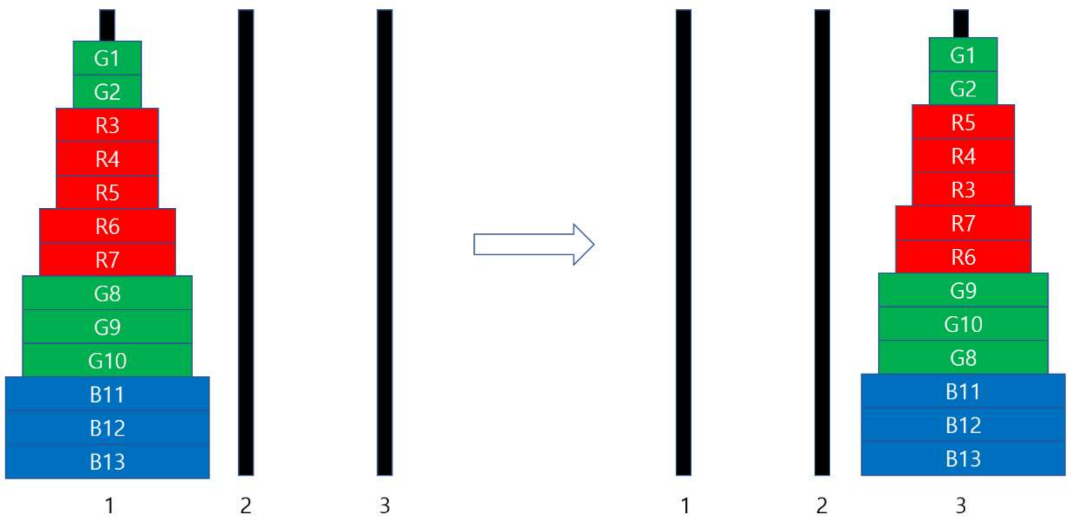


Figure C.1

Input

Your program is to read from standard input. The input starts with a line containing an integer $m (1 \le m \le 25)$, where m indicates the largest diameter of disks. For each disk of size 1 to m, the disk information is given in turn in the following m lines. More specifically, the i-th $(1 \le i \le m)$ line contains an upper-case character and an integer $k (\ge 1)$, which indicate the color and the number of the disks with diameter i, respectively. The character to specify the color is either '`R`' for red, '`G`' for green, or '`B`' for blue. Note that for each size of 1 to m, there is at least one disk with that size.

Note that the total number of disks is no more than 50.

Output

Your program is to write to standard output. Print exactly one line. The line should contain the minimum number of moves to move all the disks initially stacked on rod-1 to rod-3 satisfying the conditions on relative order depending on colors.

The following shows sample input and output for three test cases.

Sample Input 1	Output for the Sample Input 1
2 R 1 B 3	9

Sample Input 2	Output for the Sample Input 2
3 G 1 B 2 R 3	11

Sample Input 3	Output for the Sample Input 3
5 G 2 R 3 R 2 G 3 B 3	120

Problem C
Colorful Tower of Hanoi

제한 시간: 1 초

이 문제는 잘 알려진 하노이 탑 문제의 변형이다.

하노이 탑 문제는 다음과 같다. 세 개의 막대기, 즉 막대기-1, 막대기-2, 막대기-3 이 있고, 막대기-1 에 크기가 다른 n 개의 디스크가 내림차순으로 쌓여 있다. 즉, 가장 작은 디스크가 가장 위에, 가장 큰 디스크가 가장 아래에 놓여 있다. 문제의 목표는 막대기-1 에 쌓여 있는 모든 디스크를 막대기-3 으로 옮기는 것이다. 단, 옮기는 과정에서, 한 번에 하나의 디스크만 옮길 수 있으며, 어떤 경우에도 큰 디스크가 작은 디스크 위에 놓여서는 안 된다.

원래의 하노이 탑 문제가 아래와 같이 변형된다.

1. 동일한 크기의 디스크가 허용된다. 따라서 디스크를 이동하는 과정에서 어떤 디스크는 크기가 동일하거나 또는 더 큰 디스크 위에 놓을 수 있다.
2. 각 디스크는 빨간색(R), 녹색(G) 또는 파란색(B)의 세 가지 색상 중 하나로 칠해져 있다. 하지만, 동일한 크기의 디스크는 동일한 색상을 가진다.
3. 동일한 크기의 디스크가 둘 이상일 경우, 디스크를 옮긴 후 이들 간의 상대적 순서가 유지될 수도 있고, 반대로 될 수 있는데, 이를 디스크 색이 결정한다. 즉, 디스크 색이 빨간색이면 모든 디스크를 최종적으로 이동한 후 동일한 크기의 디스크의 상대적인 순서가 반대로 되어야 한다. 디스크의 색이 파란색이면 모든 디스크의 이동이 완료된 후의 상대적인 순서가 처음의 순서와 동일해야 한다. 그리고 색이 녹색이면 이동 후의 상대적인 순서가 중요하지 않다. 다만, 디스크를 이동시키는 중간 과정에서는 상대적인 순서에 대한 조건을 만족할 필요는 없다.
4. 디스크 이동의 총 횟수를 최소화해야 한다.

그림 C.1 은 디스크를 모두 옮긴 후, 크기가 같은 디스크의 색상에 따른 상대적 순서의 조건을 만족시키는 예를 보여준다.

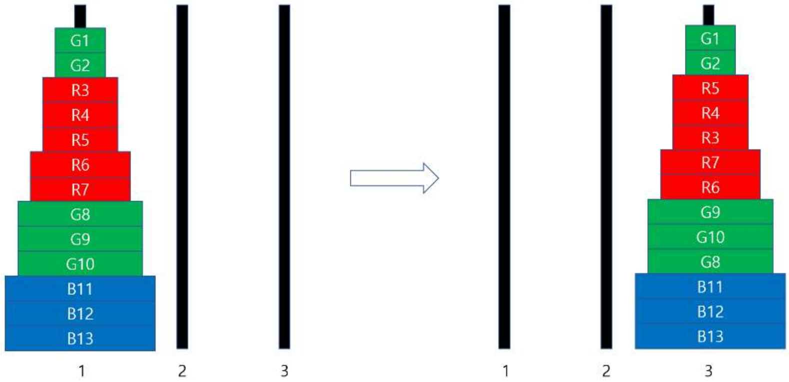


그림 C.1

Input

입력은 표준입력을 사용한다. 첫 번째 줄에는 정수 $m(1 \leq m \leq 25)$이 주어진다. 여기서, m은 가장 큰 디스크의 지름을 나타낸다. 이어지는 m 줄에서, i $(1 \leq i \leq m)$ 번째 줄에는 하나의 영어 대문자와 정수 $k(\geq 1)$가 주어지는데, 이는 지름의 크기가 i 인 디스크에 관한 정보를 나타낸다. 즉, 영어 대문자는 디스크의 색을, 정수 k는 동일한 크기의 디스크 개수를 나타낸다. 문자 'R'은 빨간색, 'G'는 녹색, 'B'는 파란색을 각각 의미한다. 1부터 m까지의 디스크 지름 각각에 대해서, 그 지름을 가진 디스크가 적어도 하나 이상 존재한다.

참고로, 쌓여 있는 디스크의 총 개수는 50 을 넘지 않는다.

Output

출력은 표준출력을 사용한다. 결과를 한 줄에 출력하되, 디스크의 색에 따른 상대적인 순서의 조건을 만족시키면서 막대기-1 에 쌓여 있는 모든 디스크를 막대기-3 으로 이동하기 위한 최소 이동 횟수를 출력한다.

다음은 세 테스트 경우에 대한 입출력 예이다.

Sample Input 1	Output for the Sample Input 1
2 R 1 B 3	9

Sample Input 2	Output for the Sample Input 2
3 G 1 B 2 R 3	11

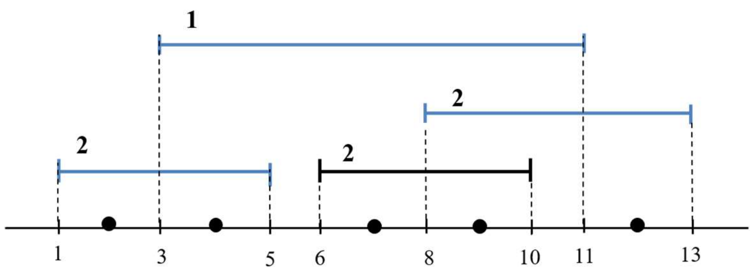
Sample Input 3	Output for the Sample Input 3
5 G 2 R 3 R 2 G 3 B 3	120

Problem D
Drones
Time Limit: 2 Seconds

A farmer grows rare flowers, which are located on a line. The total number of flowers is n, and each flower f_i has a location x_i on the line. Each flower can be considered as a point on the line. In order to nourish the flowers, the farmer fertilizes them regularly. So the farmer will rent several fertilizer drones.

The drones spray fertilizer on the ground as they fly. The flight trajectory of each drone is a line segment, which can be considered as a closed interval. There are m given drones D_i with intervals $[a_i, b_i]$, each of which has a cost c_i. We should choose a set of drones such that each of the given points is contained in at least one of the chosen intervals, that is, the union of the chosen intervals contains all the points. The set of drones satisfying this condition is said to be a candidate set of drones.

We assume that all the points and the endpoints of all the intervals have distinct coordinates. We say that a point is covered by an interval, if it is contained in the interval. For a candidate set C of drones, the cost paid to each point is defined to be the sum of the costs of intervals in C which cover the point. Our goal is to find an optimal set of drones, minimizing the maximum cost of points among all the candidate sets.



For example, there are five points on a line and four given intervals as shown in the above figure. In this figure, the numbers on the intervals represent their costs. All possible candidate sets of intervals are $\{[1,5], [6,10], [8,13]\}$, $\{[1,5], [3,11], [8,13]\}$, and the set with all of four intervals. For the first set, the point located at 9 has the cost 4, which is the maximum cost of points. For the second and the third set, the same point has the costs 3 and 5, respectively, which are the maximum. Thus the cost 3 is the minimum of the maximum cost of points among all the candidate sets.

Given the locations of n points and m intervals with their costs, write a program to output the minimum of the maximum cost of points among all the candidate sets of drones.

Input

Your program is to read from standard input. The input starts with a line containing two integers, n and m ($n \geq 1$, $2 \leq n + m \leq 100{,}000$, $1 \leq m \leq 2{,}000$), where n is the number of points and m is the number of intervals. The second line contains n integers, representing the locations of points, where the integers are between 0 and 10^9. In the following m lines, each line contains three integers a, b, and c, representing an interval $[a, b]$ with a cost c ($0 \leq a < b \leq 10^9$ and $1 \leq c \leq 10^9$). Note that all the points and the endpoints of all the intervals have distinct coordinates.

Output

Your program is to write to standard output. Print exactly one line. The line should contain the minimum of the maximum cost of points among all the candidate sets of drones. If there is no candidate set of drones, then the line should contain -1.

The following shows sample input and output for three test cases.

Sample Input 1	Output for the Sample Input 1
5 4 4 9 2 7 12 1 5 2 6 10 2 3 11 1 8 13 2	3

Sample Input 2	Output for the Sample Input 2
5 4 4 9 2 7 12 1 5 2 8 11 2 3 6 1 10 13 2	-1

Sample Input 3	Output for the Sample Input 3
18 7 3 4 6 13 14 8 9 11 22 23 16 17 18 29 30 31 26 27 5 21 2 19 24 6 7 15 3 2 12 4 20 28 5 1 10 5 25 32 3	6

Problem E
Histogram
Time Limit: 3 Seconds

For a range of $[1, n]$, the natural numbers in the interval are called the data values and let f_i be the frequency count of the data value i in the range. The frequency of a data value i is the number of occurrences of the data value i in the list of numbers. For example, the frequency of the data value 2 in the list $[3, 2, 3, 2, 4, 2]$ is 3. The following frequency table shows the frequency counts of the values in the range $[1, 8]$.

value i	1	2	3	4	5	6	7	8
frequency f_i	4	2	3	6	5	6	12	16

A bucket b_i is represented by a range $[s_i, e_i]$ and its representative value r_i. For each bucket, we use the average frequency as the representative value. We want to represent the frequency table by a histogram that consists of a small number of buckets such that the intervals of the buckets do not overlap and cover the range of all data values in the frequency table. For example, a possible histogram of the above frequency table with two buckets is shown below where the first and second buckets cover the value range $[1, 4]$ and $[5, 8]$, respectively. In the histogram, the average frequencies of the first and second buckets are 3.75 and 9.75, respectively.

bucket interval	$[1, 4]$	$[5, 8]$
average frequency	3.75	9.75

After a set of buckets from a frequency table is constructed, when the frequency of a value is asked, the average frequency of the bucket to which the value belongs should be answered. For instance, if the frequency of the value 2 is asked, since the value 2 belongs to the first bucket whose average frequency is 3.75, 3.75 will be answered instead of the true frequency 2 of the value 2. We define the error of a bucket in the histogram as the sum of the squared errors of the frequencies of all values in the bucket. In the above histogram, the error of the first bucket for the range $[1, 4]$ is $(4 - 3.75)^2 + (2 - 3.75)^2 + (3 - 3.75)^2 + (6 - 3.75)^2 = 8.75$ while that of the second bucket for the range $[5, 8]$ is $(5 - 9.75)^2 + (6 - 9.75)^2 + (12 - 9.75)^2 + (16 - 9.75)^2 = 80.75$. The error of a histogram is then defined as the sum of the errors of all buckets in the histogram. Thus, the error of the histogram becomes $8.75 + 80.75 = 89.5$. On the other hand, the following histogram has the error of 21.333333, that is the minimum among the errors of the histograms with two buckets.

bucket interval	$[1, 6]$	$[7, 8]$
average frequency	4.333333	14

Given a frequency table and the number B of buckets of the histogram, write a program that finds a minimum error histogram with at most B buckets.

Input
Your program is to read from standard input. The input starts with a line containing an integer B ($1 \le B \le 30$), where B is the limit on the number of buckets. The second line contains an integer n ($1 \le n \le 4{,}000$) that is the number of data values, indicating the range $[1, n]$. The next n lines contain the frequencies of the

data values. The i-th line of them contains a positive integer that is the frequency f_i of the data value i, where $1 \le f_i \le 100$.

Output

Your program is to write to standard output. Print exactly one line. The line should contain a real number z that represents the error value OPT of the histogram with a minimum error. The output z should be in the format that consists of its integer part, a decimal point, and its fractional part, and it should satisfy the condition that $OPT - 10^{-4} < z < OPT + 10^{-4}$.

The following shows sample input and output for two test cases.

Sample Input 1	Output for the Sample Input 1
2 8 4 2 3 6 5 6 12 16	21.333333

Sample Input 2	Output for the Sample Input 2
3 8 1 2 3 4 5 6 7 8	4.500000

Problem F
Logistical Warehouse
Time Limit: 4 Seconds

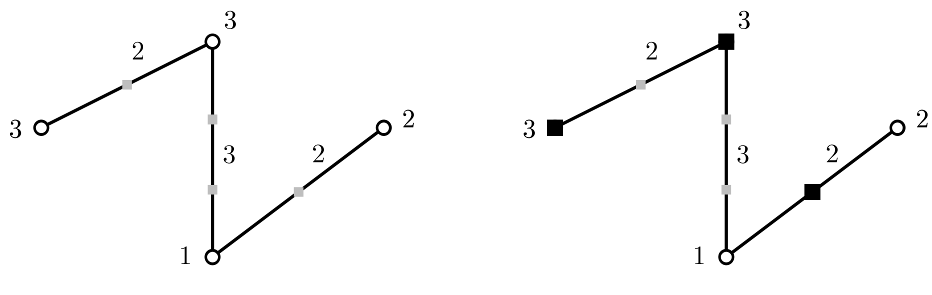
ICP(International Carrier Products) Company plans to build new k logistical warehouses in order to facilitate the efficient delivery of products. Before being delivered to the target location, the products are stored at a logistical warehouse. Then these products are delivered to the final location for distribution. The locations of logistical warehouses are very important in optimizing delivery time and space of warehouse logistics.

Imagine that the chain for supply and management is given as a tree $T = (V, E)$. Each node v_i of T has weight w_i, and each edge e_j of T has an integer length l_j. For a point p in an edge, the distance from a node v_i to p is defined to be $w_i \times |\pi(v_i, p)|$, where $\pi(v_i, p)$ is the path connecting v_i and p in T and $|\pi(v_i, p)|$ is the sum of the lengths of segments (edges) in $\pi(v_i, p)$.

We want to choose k points on the edges of T as centers under the following restriction: each center lying on an edge e must be a point at an integer distance from each endpoint of e. Observe that centers are allowed to lie on nodes. For instance, if the length of e is 3, we can choose a center among the following four points: the two endpoints of e and the two points on e at distance 1 from the endpoints of e.

The goal is to choose k centers on the edges of T such that the maximum distance from a node to its closest center is minimized. We call a set of such k centers an *optimal set* of k centers for this problem.

For example, figure (a) shows a tree with four nodes of weights $3, 3, 1, 2$ labelled next to the vertices, and three edges with lengths $2, 3, 2$ labelled next to the edges. Centers are allowed to lie on the nodes and the small gray squares. If we choose three centers among them, an optimal solution is shown in the black squares of figure (b) with the maximum distance 2.



(a) (b)

Given a tree $T = (V, E)$ of n nodes with weights and $n - 1$ edges with integer lengths, write a program to output the maximum distance from a node to its closest center in an optimal set of k centers.

Input

Your program is to read from standard input. The input starts with a line containing two integers n and k ($1 \le k \le n \le 200{,}000$), where n is the number of nodes in a tree and k is the number of centers to choose. Then there are $n - 1$ edges in the tree. The nodes are numbered from 1 to n, and the edges are numbered from 1 to $n - 1$. The next line contains n positive integers such that the i-th integer represents the weight of the i-th node. The weights are no more than 10^6. In the following $n - 1$ lines, the i-th line contains three positive integers.

The first two integers represent the two indices of the nodes at the endpoints of the i-th edge. The third integer represents the length of the i-th edge. The lengths are no more than 10^6.

Output

Your program is to write to standard output. Print exactly one line. The line should contain the maximum distance from a node to its closest center in an optimal set of k centers.

The following shows sample input and output for four test cases.

Sample Input 1	Output for the Sample Input 1
2 1 2 3 1 2 3	4

Sample Input 2	Output for the Sample Input 2
3 2 12 2 1 1 2 1 2 3 3	2

Sample Input 3	Output for the Sample Input 3
4 3 3 3 1 2 1 2 2 2 3 3 3 4 2	2

Sample Input 4	Output for the Sample Input 4
6 2 7 4 3 4 12 3 6 5 4 2 3 3 5 1 12 4 5 3 1 2 4	15

Problem G
Moving Logs
Time Limit: 0.5 Seconds

There are n logs placed in a rectangular warehouse. The logs do not intersect or overlap each other. The right wall of the warehouse is open, through which logs can be dragged out to the right. A log is only moved parallel in the positive x-axis direction. A log can only be pulled out if there are no other logs in the space through which it will move. In Figure G.1, the movement space of log 3 is grayed out. Log 3 cannot be pulled out until logs 1 and 5 have been removed.

Multiple logs can be pulled out simultaneously if there are no other logs in the space through which they will move. Suppose that it takes 1 unit of time to pull out a log. Your task is to pull out all the logs of the warehouse as quickly as possible.

In Figure G.1, in order to pull out all five logs, you have to pull out the logs one by one in the order of 1-5-3-2-4. Therefore, it takes 5 units of time to complete the task. Note that since the end point of log 1 is located in the movement space of log 5, it is not possible to pull out log 5 first.

Consider an example shown in Figure G.2. Logs 2 and 4 can be pulled out at the same time. After that, you can pull out logs 1 and 3 at the same time. Finally, you can pull out log 5. Therefore, it takes 3 units time.

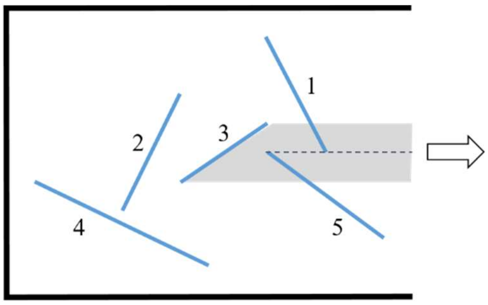


Figure G.1

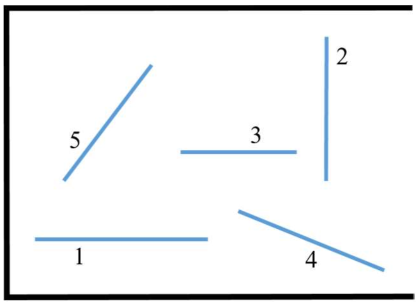


Figure G.2

Given the locations of n logs, write a program to find the minimum time required to pull out all the logs.

Input
Your program is to read from standard input. The input starts with a line containing an integer n ($1 \le n \le 20{,}000$), where n is the number of logs. The logs are numbered from 1 to n. In the following n lines, the i-th line contains four integers, x_1, y_1, x_2, and y_2, where (x_1, y_1) and (x_2, y_2) are the coordinates of both end points of the i-th log and all the integers are between 1 and 10^9. The length of a log is more than 0 and no two logs intersect each other at any point.

Output

Your program is to write to standard output. Print exactly one line. The line should contain an integer representing the minimum units of time to pull out all the logs.

The following shows sample input and output for two test cases.

Sample Input 1	Output for the Sample Input 1
5 9 9 11 5 4 3 6 7 6 4 9 6 1 4 7 1 13 2 9 5	5

Sample Input 2	Output for the Sample Input 2
5 1 2 7 2 11 4 11 9 10 5 6 5 8 3 13 1 2 4 5 8	3

Problem H
Similarity
Time Limit: 1 Second

In modern application systems, a recommendation system is very widely used to recommend books, music, ads, items, etc. The recommendation system needs to attract other users by providing the most interested items to each user. One way of recommendation is to find the most similar user to the current user, then recommend the items that the most similar user purchased to the current user. To help the recommendation system, we design a similarity measure. A user is represented by a sequence $p = p_1, p_2, \dots, p_n$ where n denotes the number of items. It denotes a list of the preference magnitudes of the user for items. When we are given two sequences $p = p_1, p_2, \dots, p_n$ and $q = q_1, q_2, \dots, q_n$, a *similar tuple* is defined as a tuple (i, j, k) such that $p_i < p_j < p_k$ and $q_i < q_j < q_k$. For given two sequences $p = p_1, p_2, \dots, p_n$ and $q = q_1, q_2, \dots, q_n$, the *similarity* is defined as the number of similar tuples.

For example, if the given two sequences are $p = 2, 5, 9, 5, 1$ and $q = 1, 4, 5, 3, 2$, the similar tuples are $(1, 2, 3)$, $(1, 4, 3)$, $(5, 2, 3)$, and $(5, 4, 3)$ The similarity of the two sequences is 4.

Given two sequences $p = p_1, p_2, \dots, p_n$ and $q = q_1, q_2, \dots, q_n$, write a program to output the similarity of them.

Input
Your program is to read from standard input. The input starts with a line containing one integer n $(1 \leq n \leq 100{,}000)$, where n is the length of a sequence. In the following two lines, each line contains n integers in range $[0, 10^6]$ that represent a sequence.

Output
Your program is to write to standard output. Print exactly one line. The line should contain the similarity of the two sequences.

The following shows sample input and output for two test cases.

Sample Input 1	Output for the Sample Input 1
5 2 5 9 5 1 1 4 5 3 2	4

Sample Input 2	Output for the Sample Input 2
4 3 2 1 1 3 2 1 1	2

Problem I
Sport Climbing Combined
Time Limit: 1 Second

Sport Climbing is an indoor rock climbing sport that started in 1986. Athletes originally competed only in **lead climbing** events, with **speed climbing** being added in 1989, and the **bouldering** discipline added a decade later in 1999. At the Olympic Games, athletes compete in the three disciplines that are combined in a single ranking to decide gold, silver and bronze medals. The ranking is determined by a multiplied score of climbers' placements in the three disciplines. For example, if a climber places first in lead, 5th in speed, second in bouldering, then the score would be 10. The climber with the lower score has precedence in the ranking.

Given the bib numbers of n climbers and their placements in the three disciplines, write a program to determine which climbers will be awarded gold, silver, and bronze medals. It may happen that two climbers have the same multiplied score. In this case, the climber with the lower added score of placements in the three disciplines wins. If two climbers have the same added score as well as the multiplied score, the climber with the lower bib number wins.

Input
Your program is to read from standard input. The first line contains a positive integer n ($3 \le n \le 100$) indicating the number of climbers. In the following n lines, each line contains four integers b_i, p_i, q_i, r_i, where b_i is the bib number of i-th climber, and p_i, q_i, r_i respectively are his/her placements in the three disciplines of lead, speed and bouldering. You may assume that the bib numbers are distinct and positive integers no more than 999; also, the placements of climbers in each discipline are integers between 1 and n. Two climbers may have the same placement in each discipline.

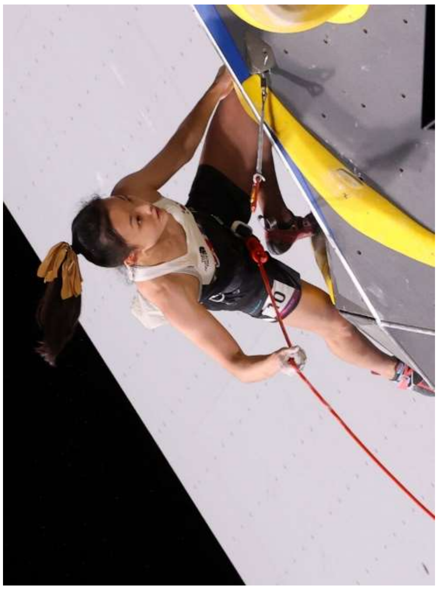
Output
Your program is to write to standard output. Print exactly one line that contains three integers representing the bib numbers of the climbers who will be awarded gold, silver, and bronze medals in order.

The following shows sample input and output for one test case.

Sample Input	Output for the Sample Input
4 301 4 3 2 815 2 2 1 717 1 1 4 505 3 4 2	815 717 301

Problem I
Sport Climbing Combined

제한 시간: 1 초



스포츠 클라이밍은 1986 년에 시작된 실내 암벽 등반 스포츠이다. 선수들은 원래 **리드 클라이밍** 종목에서만 겨루었는데, 1989 년에 **스피드 클라이밍**이 추가되었고, 10 년 후인 1999 년에 **볼더링** 종목이 추가되었다. 올림픽 게임에서는 금, 은, 동메달을 결정하기 위하여 선수들은 세 종목에서 겨루어 종합 순위를 매긴다. 종합 순위는 세 종목에서 거둔 순위를 곱한 점수로 결정된다. 예를 들어, 어떤 선수가 리드에서 1 위, 스피드에서 5 위, 볼더링에서 2 위를 했다면 점수는 10 점이 된다. 곱한 점수가 낮은 선수가 종합 순위에서 앞선다.

선수 n명의 등번호와 이들이 세 종목에서 거둔 순위가 주어질 때, 금, 은, 동메달을 받을 선수를 결정하는 프로그램을 작성하시오. 두 선수의 곱한 점수가 같을 수도 있다. 이 경우, 세 종목 순위의 합산 점수가 낮은 선수가 이긴다. 두 선수의 곱한 점수와 합산 점수가 모두 같으면 등번호가 낮은 선수가 이긴다.

Input

입력은 표준입력을 사용한다. 첫째 줄에 선수의 명수를 나타내는 양의 정수 n $(3 \le n \le 100)$이 주어진다. 이어 n개의 줄 각각에 네 정수 b_i, p_i, q_i, r_i가 주어지는데, b_i는 i번째 선수의 등번호이고, p_i, q_i, r_i는 각각 그 선수가 리드, 스피드, 볼더링 종목에서 거둔 순위를 나타낸다. 선수들의 등번호는 서로 다르고 999 이하인 양의 정수이다. 또한, 각 종목에서 선수의 순위는 1과 n 사이의 정수이다. 각 종목에서 두 선수의 순위가 같을 수도 있다.

Output

표준출력을 사용한다. 금, 은, 동메달을 받을 선수의 등번호를 나타내는 세 정수를 순서대로 한 줄에 출력한다.

다음은 한 테스트 경우에 대한 입출력 예이다.

Sample Input	Output for the Sample Input
4 301 4 3 2 815 2 2 1 717 1 1 4 505 3 4 2	815 717 301

Problem J
Ten
Time Limit: 0.5 Seconds

A real estate company IC is managing a rectangular section of land. The section is divided into mn segments in $m \times n$ matrix shape, where the number of rows and that of columns are m and n, respectively. Each segment has its own price as a positive integer. IC wants to sell a rectangular subsection of the land, but the price of the subsection should be ten. The price of a subsection is simply the sum of the prices of the segments in the subsection. Since there can be several such subsections, IC wants to identify the number of candidate subsections to sell. Write a program to help IC, counting the number of candidate subsections of the land.

For example, the prices of segments of the land having 5×7 segments are given as follows:

3	1	2	1	4	6	7
4	5	2	2	2	7	9
4	7	1	1	2	9	2
4	5	3	8	7	2	5
1	7	1	7	6	2	3

We can find four candidate subsections to sell marked by rectangles: the first one consists of four segments in the first and the second rows spanning over from the second to the third columns, the second, six segments in the second and the third rows spanning over from the third to the fifth columns, the third, two segments in the first row spanning over from the fifth to the sixth columns, and the fourth, three segments in the seventh column spanning over from the third to the fifth rows. Therefore, your program should report four for the above input.

Input

Your program is to read from standard input. The input starts with two positive integers m and n ($1 \leq m, n \leq 300$), denoting the dimensions of the land, which are given separated by a space. Each of the following m lines contains n positive integers p_{ij} representing the prices of the segments of the i-th row of the land ($1 \leq i \leq m$, $1 \leq j \leq n$, and $1 \leq p_{ij} \leq 10$). The prices are also separated by a space.

Output

Your program is to write to standard output. Print exactly one line containing an integer representing the number of rectangular subsections with the price of ten.

The following shows sample input and output for two test cases.

Sample Input 1

```
3 5
3 1 2 1 4
4 5 2 2 2
4 7 1 1 2
```

Output for the Sample Input 1

```
2
```

Sample Input 2

```
4 6
3 1 2 1 4 6
4 5 2 2 2 7
4 7 1 1 1 9
4 3 3 3 7 2
```

Output for the Sample Input 2

```
8
```

Problem K
Treasure Hunter
Time Limit: 1 Second

Chulsu has a treasure map that marks the locations of k hidden treasures, where each location is presented as a position in the cell of an $m \times n$ grid. Since all the locations in the map except the entrance and exit are hard to access for humans, Chulsu wants to use treasure hunt robot (THR)s to access and collect the treasures. THR always starts from an entrance cell $(1,\ 1)$ and exits through an exit cell (m, n) (here, $(i,\ j)$ denotes a cell at the i-th row and the j-th column). Also, THR can only go immediately right or below from the current cell and cannot be reused after arriving at the exit cell. The following figure shows an example when $m = 3$, $n = 5$, and $k = 4$ (in the figure, X denotes a cell where the treasure exists).

$(1,1)$				
		X		
X	X		X	$(3,5)$

In the above example, if THR accesses the cells $(1,\ 1)$, $(2,\ 1)$, $(3,\ 1)$, $(3,\ 2)$, $(3,\ 3)$, $(3,\ 4)$, and $(3,\ 5)$ in order, it collects three treasures (at $(3,\ 1)$, $(3,\ 2)$, and $(3,\ 4)$) after arriving at the exit cell $(3,\ 5)$. In this case, to collect the treasure at the position $(2,\ 3)$, Chulsu needs to use an additional THR. When m, n, k, and k distinct positions of treasures are given, write a program to compute the minimum number of THR's to collect all the treasures on the map.

Input
Your program is to read from standard input. The input starts with a line containing three integers, m, n, and k $(1 \le m, n, k \le 100{,}000)$, where m is the number of rows, n is the number of columns of the map, and k is the number of treasures on the map. In the following k lines, k distinct positions of treasures are given as two integers where the two integers represent the row and column positions on the map, respectively.

Output
Your program is to write to standard output. Print exactly one line. The line should contain the minimum number of THR's to collect all the treasures on the map.

The following shows sample input and output for two test cases.

Sample Input 1	Output for the Sample Input 1
3 5 4 3 1 3 2 2 3 3 4	2

Sample Input 2	**Output for the Sample Input 2**
7 8 10 3 1 3 2 3 4 2 6 3 7 3 8 4 8 5 3 5 4 5 5	3

Problem L
Triangles
Time Limit: 1.2 Seconds

A triangle is a simple polygon having exactly three distinct corners that are not collinear. For any set S of points in the plane, any triangle is said to have a conflict to S if there is at least one point contained both in S and in the interior of the triangle, excluding its boundary, simultaneously.

Now, the set S is given to be a set of n points in the plane, no three of which are collinear, and let T be the set of all triangles whose corners are chosen from the set S. How many of those in T do have a conflict to S?

Write a program that outputs the number of triangles in T that have a conflict to S.

Input
Your program is to read from standard input. The input starts with a line containing an integer n ($3 \le n \le 500$), where n is the number of points in the set S. Each of the following n lines consists of two integers, each between -10^6 and 10^6, representing the coordinates of each point in the set S. It is guaranteed that no three points in the set S are collinear.

Output
Your program is to write to standard output. Print exactly one line. The line should contain an integer that represents the number of triangles whose corners are chosen from the set S and that have a conflict to the set S.

The following shows sample input and output for two test cases.

Sample Input 1	Output for the Sample Input 1
4 1 1 0 0 2 3 6 1	1

Sample Input 2	Output for the Sample Input 2
6 0 1 2 0 3 -2 3 2 7 1 2 4	6