21-st International Olympiad In Informatics

TASKS AND SOLUTIONS

August 8 - 15, 2009

Plovdiv, Bulgaria

Day 0

Task 0.1 AREA

A large agricultural corporation is undertaking an audit of its N land properties. As a first step, they would like to know the total area of their properties. Each property has a rectangular shape, and no two properties overlap. You realize that this is a trivial calculation, but you understand that you need to solve this problem in order to test your ability to properly handle large amounts of data and large numbers under the IOI 2009 contest system.

TASK

Write a program that, given the dimensions of the N land properties, determines the sum of their areas.

CONSTRAINTS

$1 \leq N \leq 500\ 000$ – The number of land properties
$1 \leq A_k, B_k \leq 20\ 000$ – The lengths of the sides of property k

INPUT

Your program should read from the standard input the following data:
• The first line contains a single integer: the number of properties N.
• The next N lines describe the properties, one property per line. The k^{th} of these lines describes property number k and it contains two integers separated by a single space: the lengths of the sides of the property A_k and B_k measured in meters.

OUTPUT

Your program should write to the standard output a single line containing a single integer: the total area of all properties, measured in square meters.

IMPORTANT NOTE

The final answer may not fit in 32 bits. You have to use a 64-bit data type, such as `long long` in C/C++ or `int64` in Pascal, in order to compute and store the answer in a single variable. Please see the technical info sheet for details.

EXAMPLE

Sample Input	Sample Output
3 15 25 5 6 3 1	408

Task 0.2 HILL (proposed by Iskren Chernev)

An extraterrestrial team of contestants, known as the aliens, is coming to Plovdiv for the IOI. They would like to land their flying saucer on one of Plovdivs hills, but they have no map of the city, so they need to use their laser scanner to find a hill. The aliens have divided the city into a grid of N by M cells, and each time they use the scanner they can determine the altitude of a single cell. Their scanner is so precise that no two cells would ever have the same altitude.

The aliens define a hill as a cell that has higher altitude than all of its adjacent cells. Two cells are considered adjacent if they share a side. Thus typically each cell has four adjacent ones, with the cells on the border of the grid being the exception (they have less than four adjacent cells).

The aliens have time for only 3 050 scans before their saucer hits the ground. Help the aliens find a hill before they run out of time.

TASK

Write a program that, given the dimensions of the grid, interactively uses the scanner to determine the location of a hill, while doing no more than 3 050 scans in total.

CONSTRAINTS

$1 \leq N, M \leq 1\ 000$ – The dimensions of the grid
$1 \leq A_{x,y} \leq 2\ 000\ 000\ 000$ – The altitude of cell $< x, y >$

INPUT

Your program should read from the standard input a single line with two integers N and M, separated by a single space.

INTERACTION

After reading the input data, your program should start using the laser scanner. Each time you want to use the scanner for a cell with coordinates x and y ($1 \leq x \leq N$, $1 \leq y \leq M$), you should print a single line on the standard output, containing three integers separated by single spaces: 0 (a literal zero), x and y, in this order.

After submitting your request to the scanner, you should read a single line on the standard input. It will contain a single integer: the altitude of the cell with coordinates x and y in some alien units.

OUTPUT

When your program has determined the location of a hill $< a, b >$ ($1 \leq a \leq N$, $1 \leq b \leq M$), you should report your answer by printing a single line on the standard output, containing three integers separated by single spaces: 1 (a literal one), a and b, in this order.

IMPORTANT NOTE: In order to interact properly with the scanner, your program needs to flush the standard output after every line that you print on the standard output. Please see the technical info sheet for instructions on how to do this properly.

EXAMPLE

Sample Input	Sample Output	
3 2		
	0 1 1	[flush stdout/output]
3		
	0 2 1	[flush stdout/output]
4		
	0 3 1	[flush stdout/output]
1		
	0 1 2	[flush stdout/output]
7		
	0 2 2	[flush stdout/output]
6		
	1 1 2	[flush stdout/output]

0.3 MUSEUM (proposed by Boyko Bantchev)

The Plovdiv Museum of Modern Art has an exhibition of ancient Thracian vases. There are N vases total. The first one is a miniature of height 1 centimeter. The second one is of height 2 centimeters; the third one is 3 centimeters tall and so on until the N^{th} vase, which is N centimeters tall.

Since this a modern art museum and the vases are ancient, the organizers of the exhibition would like to add a modern, chaotic twist to the presentation of the vases. They have decided to arrange the vases in a line that satisfies the following condition: For any three vases A, B and C, such that Bs height is exactly the average of the heights of A and C, either B must be positioned to the left of both A and C, or B must be positioned to the right of both A and C (in other words, B may not be positioned between A and C on the line).

TASK

Write a program that, given the number of vases, determines a linear arrangement of the vases that satisfies the condition of the exhibition organizers.

CONSTRAINTS

$1 \leq N \leq 2\ 000$ – The number of vases

INPUT

You are given five problem instances in the files `museum.1.in` to `museum.5.in`. Each file contains a single line, which in turn contains a single integer: the number of vases N.

OUTPUT

You are to submit five output files, named `museum.1.out` to `museum.5.out`, each corresponding to one of the input files. The files should be in the following format:

There should be N lines, each representing the N positions in the arrangement, in order from left to right. Line k should contain a single integer H_k, the height of the vase you decided to place on position k. All N heights should be distinct integers between 1 and N inclusive.

EXAMPLE

Sample Input `museum.0.in`	Sample Output `museum.0.out`
5	3 1 2 5 4

In the above arrangement, 3 is neither between 2 and 4, nor is it between 1 and 5. Also, 2 is not between 1 and 3, and 4 is not between 3 and 5. Thus, it satisfies the condition of the exhibition organizers.

TECHNICAL INFO SHEET (day 0)

These pages contain helpful information on how to avoid slow input/output performance with C++ streams (cin / cout), how to use 64-bit data types (variables) and how to flush the output for interactive tasks.

Slow Input / Output with C++ Streams

When solving tasks with very large amounts of input / output data, you may notice that C++ programs using the cin and cout streams are much slower than equivalent programs that use the scanf and printf functions for input and output processing. Thus, if you are using the cin / cout streams we strongly recommend that you switch to using scanf / printf instead. However, if you still want to use cin / cout, we recommend adding the following line at the beginning of your program:

```
ios::sync_with_stdio(false);
```

and also making sure that you never use `endl`, but use `\n` instead.

Please note, however, that including `ios::sync_with_stdio(false)` breaks the synchrony between cin / cout and scanf / printf, so if you are using this, you should never mix usage of cin and scanf, nor mix cout and printf.

64-bit Data Types

For some tasks you may need to deal with numbers too large to fit in 32 bits. In these cases, you would have to use a 64-bit integer data type, such as long long

in C/C++ or int64 in Pascal. Here is some sample code that illustrates the usage of these data types:

C/C++

```
int main(void) {
    long long varname;
    scanf("%lld", &varname);
    // Do something with the varname variable
    printf("%lld\n", varname);
    return 0;
}
```

Pascal

```
var
    varname: Int64;
begin
    read(varname);
    { Do something with the varname variable }
    writeln(varname);
end.
```

Flushing the Output

Whenever you solve an interactive task, you always need to flush the buffer of your output after every new line printed on the output. Here is some code to illustrate how to do this under C, C++ and Pascal:

C or C++ with scanf / printf

```
fflush(stdout);
```

C++ with cin / cout

```
cout << flush;
```

Pascal

```
flush(output);
```

Task Overview Sheet (Day 0)

	Area	Hill	Museum
Type	Batch	Interactive	Output-only
Detailed Feedback	Full	Partial	None
Time Limit (per test case)	2 seconds	1 second	Not Applicable
Memory Limit (per test case)	32 MB	64 MB	Not Applicable
Points	100	100	100

NOTE: On the actual competition there will be four problems on each day. We have only three problems here because the practice contest is shorter in duration and because there are only three possible task types.

Day 1

1.1 ARCHERY (proposed by Velin Tzanov)

An archery tournament is held according to the following rules. There are N targets arranged in a line and numbered from 1 to N inclusive according to their place on the line (the leftmost target being target 1, and the rightmost target being target N). There are also $2*N$ archers. At any time during the tournament, there are two archers on each target. Every round of the tournament goes according to the following procedure:

The two archers on each target compete with each other and determine a winner and a loser between them. Then all archers are rearranged as follows:

The winners on targets 2 to N inclusive move to the target on their left (i.e., targets 1 to $N-1$ respectively).

The losers on targets 2 to N inclusive, as well as the winner on target 1, remain on the same target.

The loser on target 1 moves to target N.

The tournament continues for R rounds, with the number of rounds being at least as many as the number of archers (i.e., $R \geq 2*N$).

You are the only archer to arrive for the tournament exactly on time. All other $2*N-1$ archers have arrived early and are already standing in a line. What you have to do now is to insert yourself somewhere into the line amongst them. You know that after you take your position, the two leftmost archers in the line will start the tournament on target 1, the next two will start on target 2 and so on, with the two rightmost archers starting on target N.

All the $2*N$ archers in the tournament (including yourself) are ranked by skill, where a smaller rank corresponds to better skill. No two archers have the same rank. Also, whenever two archers compete, the one with the smaller rank will always win.

Knowing how skilled each of your competitors is, you want to insert yourself in such a way as to ensure that you will finish the tournament on a target with as small a number as possible. If there are multiple ways to do this, you prefer the one that starts at a target with as large a number as possible.

TASK

Write a program that, given the ranks of all archers, including yourself, as well as your competitors arrangement on the line, determines on which target you

should start the tournament, so that you can achieve your goals as defined above.

CONSTRAINTS

$1 \leq N \leq 200\ 000$ – The number of targets; also equal to half the number of archers

$2 * N \leq R \leq 1\ 000\ 000\ 000$ – The number of tournament rounds

$1 \leq S_k \leq 2 * N$ – The rank of archer k

INPUT

Your program must read from standard input the following data:

The first line contains the integers N and R, separated by a space.

The next $2 * N$ lines list the ranks of the archers. The first of these lines contains your rank. The rest of these lines contain the ranks of the other archers, one archer per line, in the order in which they have arranged themselves (from left to right). Each of these $2 * N$ lines contains a single integer between 1 and $2 * N$ inclusive. A rank of 1 is the best and a rank of $2 * N$ is the worst. No two archers have the same rank.

OUTPUT

Your program must write to standard output a single line containing a single integer between 1 and N inclusive: the number of the target on which you will start the tournament.

GRADING

For a number of tests, worth a total of 60 points, N will not exceed 5 000. Also, for some of these tests, worth a total of 20 points, N will not exceed 200.

EXAMPLES

Sample Input	Smaple Output
4 8 7 4 2 6 5 8 1 3	3

You are the second worst archer. If you start on target 1, you will then go to target 4 and stay there until the end. If you start on target 2 or 4, you will just stay there for the whole tournament. If you start on target 3, you will beat the worst archer and then move to target 2 and stay there.

Sample Input	Smaple Output
4 9 2 1 5 8 3 4 7 6	2

You are the second best archer. The best one is already on target 1 and will stay there for the whole duration of the tournament. Thus, no matter where you start, you will always move from your target, going through all targets from 4 to 1 over and over again. In order for you to end on target 1 after 9 transitions, you have to start on target 2.

1.2 HIRING (proposed by Velin Tzanov)

You have to hire workers for a construction project. There are N candidates applying for the job, numbered from 1 to N inclusive. Each candidate k requires that if he is hired, he must be paid at least S_k dollars. Also, each candidate k has a qualification level Q_k. The regulations of the construction industry require that you pay your workers in proportion to their qualification level, relative to each other. For example, if you hire two workers A and B, and $Q_A = 3 * Q_B$, then you have to pay worker A exactly three times as much as you pay worker B. You are allowed to pay your workers non-integer amounts of money. This even includes quantities that cannot be written with a finite number of digits in decimal form, such as a third or a sixth of a dollar.

You have W dollars at hand and you want to hire as many workers as possible. You decide whom to hire and how much to pay them, but you have to meet the minimum salary requirements of those you choose to hire, and you have to obey the industry regulations. You also have to fit within your budget of W dollars.

The nature of your project is such that the qualification level is completely irrelevant, so you are only interested in maximizing the number of workers without regard to their qualification level. However, if there is more than one way to achieve this, then you want to select the one where the total amount of money you have to pay your workers is as small as possible. In case there is more than one way to achieve this, then you are indifferent among these ways and you would be satisfied with any one of them.

TASK

Write a program that, given the different salary requirements and qualification levels of the candidates, as well as the amount of money you have, determines which candidates you should hire. You must hire as many of them as possible and you must do so with as little money as possible, while complying with the industry regulations specified above.

CONSTRAINTS

$1 \leq N \leq 500\ 000$ – The number of candidates
$1 \leq S_k \leq 20\ 000$ – The minimum salary requirement of candidate k
$1 \leq Q_k \leq 20\ 000$ – The qualification level of candidate k
$1 \leq W \leq 10\ 000\ 000\ 000$ – The amount of money available to you

IMPORTANT NOTE

The maximum value of W does not fit in 32 bits. You have to use a 64-bit data type, such as long long in C/C++ or `int64` in Pascal, in order to store the value of W in a single variable. Please see the technical info sheet for details.

INPUT

Your program must read from standard input the following data:

• The first line contains the integers N and W, separated by a space.
• The next N lines describe the candidates, one candidate per line. The k^{th} of these lines describes candidate number k and it contains the integers S_k and Q_k, separated by a space.

OUTPUT

Your program must write to standard output the following data:
• The first line must contain a single integer H, the number of workers that you hire.
• The next H lines must list the identifying numbers of the candidates you choose to hire (each of them a different number between 1 and N), one per line, in any order.

GRADING

For any given test case, you will receive full points if your choice of candidates enables you to achieve all of your goals, while satisfying all constraints. If you produce an output file with a correct first line (i.e., a correct value of H), but which does not meet the above description, you will receive 50% of the points for that test case. The latter will be the case even if the output file is not properly formatted, as long as the first line is correct.

For a number of tests, worth a total of 50 points, N will not exceed 5 000.

EXAMPLES

Sample Input	Sample Output
4 100 5 1000 10 100 8 10 20 1	2 2 3

The only combination for which you can afford to hire two workers and still meet all the constraints is if you select workers 2 and 3. You can pay them 80 and 8 dollars respectively and thus fit in your budget of 100.

Sample Input	Sample Output
3 4 1 2 1 3 1 3	3 1 2 3

Here you can afford to hire all three workers. You pay 1 dollar to worker 1 and 1.50 dollars each to workers 2 and 3, and you manage to hire everyone with the 4 dollars that you have.

Sample Input	Sample Output
3 40 10 1 10 2 10 3	2 2 3

Here you cannot afford to hire all three workers, as it would cost you 60 dollars, but you can afford to hire any two of them. You choose to hire workers 2 and 3

because they would cost you the smallest sum of money, compared to the other two-worker combinations. You can pay 10 dollars to worker 2 and 15 dollars to worker 3 for a total of 25 dollars. If you were to hire workers 1 and 2 you would have to pay them at least 10 and 20 dollars respectively. If you were to hire 1 and 3, then you would have to pay them at least 10 and 30 dollars respectively.

Task 1.3 POI (proposed by Carl Hultquist)

The local Plovdiv Olympiad in Informatics (POI) was held according to the following unusual rules. There were N contestants and T tasks. Each task was graded with only one test case, therefore for every task and every contestant there were only two possibilities: either the contestant solved the task, or the contestant did not solve the task. There was no partial scoring on any task.

The number of points assigned to each task was determined after the contest and was equal to the number of contestants that did not solve the task. The score of each contestant was equal to the sum of points assigned to the tasks solved by that contestant.

Philip participated in the contest, but he is confused by the complicated scoring rules, and now he is staring at the results, unable to determine his place in the final standings. Help Philip by writing a program that calculates his score and his ranking.

Before the contest, the contestants were assigned unique IDs from 1 to N inclusive. Philips ID was P. The final standings of the competition list the contestants in descending order of their scores. In case of a tie, among the tied contestants, those who have solved more tasks will be listed ahead of those who have solved fewer tasks. In case of a tie by this criterion as well, the contestants with equal results will be listed in ascending order of their IDs.

TASK

Write a program that, given which problems were solved by which contestant, determines Philips score and his rank in the final standings.

CONSTRAINTS

$1 \leq N \leq 2\ 000$ – The number of contestants
$1 \leq T \leq 2\ 000$ – The number of tasks
$1 \leq P \leq N$ – Philips ID

INPUT

Your program must read from standard input the following data:

• The first line contains the integers N, T and P, separated by individual spaces.

• The next N lines describe which tasks were solved by which contestant. The k^{th} of these lines describes which tasks were solved by the contestant with ID k. Each such line contains T integers, separated by spaces. The first of these numbers denotes whether or not contestant k solved the first task. The second number denotes the same for the second task and so on. These T numbers are all either 0 or 1, where 1 means that contestant k solved the corresponding task, and 0 means that he or she did not solve it.

OUTPUT

Your program must write to standard output a single line with two integers separated by a single space. First, the score that Philip got on the POI competition. Second, Philips rank in the final standings. The rank is an integer between 1 and N inclusive, with 1 denoting the contestant listed at the top (i.e., a contestant who has the highest score) and N to the one listed at the bottom (i.e., a contestant with the lowest score).

GRADING

For a number of tests, worth a total of 35 points, no other contestant will have the same score as Philip.

EXAMPLE

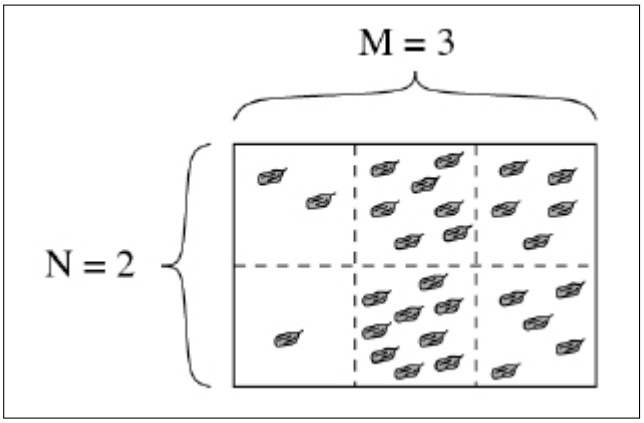
Sample Input	Sample Output
5 3 2 0 0 1 1 1 0 1 0 0 1 1 0 1 1 0	3 2

The first problem was unsolved by only one contestant, so it is worth 1 point. The second problem was unsolved by two contestants, so it is worth 2 points. The third problem was unsolved by four contestants, so it is worth 4 points. Thus the first contestant has a score of 4; the second contestant (Philip), the fourth and the

fifth contestants all have a score of 3; and the third contestant has a score of 1. Contestants 2, 4 and 5 are all tied according to the first tie-break rule (number of problems solved), and according to the second tie-break rule (smaller ID) Philip ranks before the others. Thus Philips rank in the final standings is 2. He is only behind the contestant with ID 1.

Task 1.4 RAISINS (proposed by Emil Kelevedjiev)

Plovdiv's famous master chocolatier Bonny needs to cut a slab of chocolate with raisins. The chocolate is a rectangular block of identical square pieces. The pieces are aligned with the edges of the chocolate, and they are arranged in N rows and M columns, for a total of $N * M$ pieces. Each piece has one or more raisins on it, and no raisins lie between or across pieces.



Initially, the chocolate is one single, monolithic block. Bonny needs to cut it into smaller and smaller blocks until finally she has cut the chocolate down to its $N*M$ individual pieces. As Bonny is very busy, she needs the help of her assistant, Sly Peter, to do the cutting. Peter only makes straight line, end-to-end cuts and he wants to be paid for every single cut he makes. Bonny has no money at hand, but she has plenty of raisins left over, so she offers to pay Peter in raisins. Sly Peter agrees to this arrangement, but under the following condition: every time he cuts a given block of chocolate into two smaller blocks, he has to be paid as many raisins as there are on the block he was given.

Bonny wants to pay Peter as little as possible. She knows how many raisins there are on each of the $N * M$ pieces. She can choose the order in which she gives Peter any remaining blocks, and she can also tell Peter what cuts to make (horizontal or vertical) and where exactly to make them. Help Bonny decide how to cut the chocolate into individual pieces, so that she pays Sly Peter as few raisins as possible.

TASK

Write a program that, given the number of raisins on each of the individual pieces, determines the minimum number of raisins that Bonny will have to pay Sly Peter.

CONSTRAINTS

$1 \leq N, M \leq 50$ – The number of pieces on each side of the chocolate
$1 \leq R_{k,p} \leq 1000$ – The number of raisins on the piece in the k^{th} row and the p^{th} column

INPUT

Your program must read from standard input the following data:

• The first line contains the integers N and M, separated by a single space.

• The next N lines describe how many raisins there are on each piece of the chocolate. The k^{th} of these N lines describes the k^{th} row of the chocolate. Each such line contains M integers separated by single spaces. The integers describe the pieces on the corresponding row in order from left to right. The p^{th} integer on the k^{th} line (among these N lines) tells you how many raisins are on the piece in the k^{th} row and the p^{th} column.

OUTPUT

Your program must write to standard output a single line containing a single integer: the minimum possible number of raisins that Bonny would have to pay Sly Peter

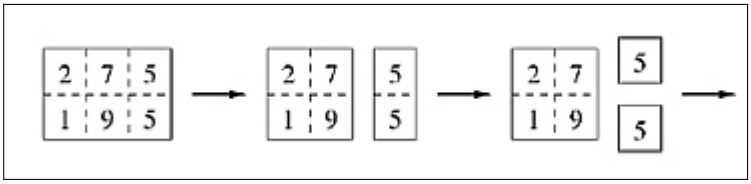
GRADING

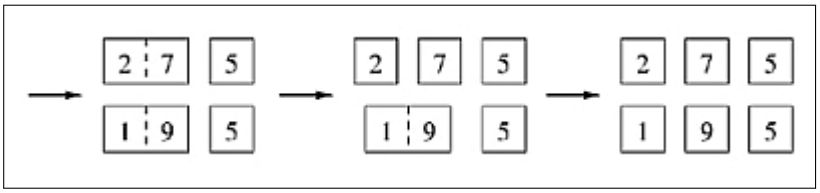
For a number of tests, worth a total of 25 points, N and M will not exceed 7.

EXAMPLE

Sample Input	Sample Output
2 3 2 7 5 1 9 5	77

One possible way (out of many) to achieve a cost of 77 is as follows:





The first cut that Bonny asks Peter to make separates the third column from the rest of the chocolate. Bonny needs to pay Peter 29 raisins for this.

Then Bonny gives Peter the smaller of the two blocks: the one that has two pieces with 5 raisins each, and asks Peter to cut the block in two in exchange for 10 raisins.

After this, Bonny gives Peter the largest remaining block: the one having pieces with 2, 7, 1 and 9 raisins respectively. Bonny asks Peter to cut it horizontally, separating the first and the second row and pays him 19 raisins.

Following this, Bonny gives Peter the top-left block, paying 9 raisins. Finally, Bonny asks Peter to split the bottom-left block, paying 10 raisins.

The total cost to Bonny is 29 + 10 + 19 + 9 + 10 = 77 raisins. No other cutting arrangement can get the chocolate cut into its 6 pieces at a smaller cost.

TECHNICAL INFO SHEET (day 1)

These pages contain helpful information on how to avoid slow input/output performance with C++ streams (cin / cout), how to use 64-bit data types (variables) and how to flush the output for interactive tasks. They also include reference for what options are given to the compilers and what stack limitations are in place.

Slow Input / Output with C++ Streams

When solving tasks with very large amounts of input / output data, you may notice that C++ programs using the cin and cout streams are much slower than equivalent programs that use the scanf and printf functions for input and output processing. Thus, if you are using the cin / cout streams we strongly recommend that you switch to using scanf / printf instead. However, if you still want to use cin / cout, we recommend adding the following line at the beginning of your program:

```
ios::sync_with_stdio(false);
```

and also making sure that you never use `endl`, but use `\n` instead.

Please note, however, that including `ios::sync_with_stdio(false)` breaks the synchrony between cin / cout and scanf / printf, so if you are using this, you should never mix usage of cin and scanf, nor mix cout and printf.

64-bit Data Types

For some tasks you may need to deal with numbers too large to fit in 32 bits. In these cases, you would have to use a 64-bit integer data type, such as `long long` in C/C++ or `int64` in Pascal. Here is some sample code that illustrates the usage of these data types:

C/C++

```
int main(void) {
    long long varname;
    scanf("%lld", &varname);
    // Do something with the varname variable
    printf("%lld\n", varname);
    return 0;
}
```

Pascal

```
var
    varname: Int64;
begin
    read(varname);
    { Do something with the varname variable }
    writeln(varname);
end.
```

Flushing the Output

Whenever you solve an interactive task, you always need to flush the buffer of your output after every new line printed on the output. Here is some code to illustrate how to do this under C, C++ and Pascal:

C or C++ with scanf / printf

```
fflush(stdout);
```

C++ with cin / cout

```
cout << flush;
```

Pascal

```
flush(output);
```

Compiler Options

The following commands will be used to compile solutions of batch and interactive tasks (say the task name is abc):

C

```
gcc -o abc abc.c -std=gnu99 -O2 -s -static -lm -x c
```

C++

```
g++ -o abc abc.cpp -O2 -s -static -lm -x c++
```

Pascal

```
fpc -O2 -XS -Sg abc.pas
```

Stack Limitations

Whenever your program is executed through the contest system, the stack size will only be limited by the memory limit for the corresponding task.

Task Overview Sheet (Day 1)

	Archery	Hiring	POI	Raisins
Type	Batch	Batch	Batch	Batch
Detailed Feedback	Partial	None	Full	Partial
Time Limit (per test case)	2 seconds	1.5 seconds	2 seconds	5 seconds
Memory Limit (per test case)	64 MB	64 MB	64 MB	128 MB
Points	100	100	100	100

NOTE: C++ programmers should be aware that using C++ streams (cin / cout) may lead to I/O bottlenecks and substantially slower performance. Please see the technical info sheet for ways to avoid this.

Day 2

Task 2.1 GARAGE (proposed by Carl Hultquist)

A parking garage has N parking spaces, numbered from 1 to N inclusive. The garage opens empty each morning and operates in the following way throughout the day. Whenever a car arrives at the garage, the attendants check whether there are any parking spaces available. If there are none, then the car waits at the entrance until a parking space is released. If a parking space is available, or as soon as one becomes available, the car is parked in the available parking space. If there is more than one available parking space, the car will be parked at the space with the smallest number. If more cars arrive while some car is waiting, they all line up in a queue at the entrance, in the order in which they arrived. Then, when a parking space becomes available, the first car in the queue (i.e., the one that arrived the earliest) is parked there.

The cost of parking in dollars is the weight of the car in kilograms multiplied by the specific rate of its parking space. The cost does not depend on how long a car stays in the garage.

The garage operator knows that today there will be M cars coming and he knows the order of their arrivals and departures. Help him calculate how many dollars his revenue is going to be today.

TASK

Write a program that, given the specific rates of the parking spaces, the weights of the cars and the order in which the cars arrive and depart, determines the total revenue of the garage in dollars.

CONSTRAINTS

$1 \leq N \leq 100$ – The number of parking spaces
$1 \leq M \leq 2\ 000$ – The number of cars
$1 \leq R_s \leq 100$ – The rate of parking space s in dollars per kilogram
$1 \leq W_k \leq 10\ 000$ – The weight of car k in kilograms

INPUT

Your program must read from standard input the following data:
• The first line contains the integers N and M, separated by a space.

• The next N lines describe the rates of the parking spaces. The s^{th} of these lines contains a single integer R_s, the rate of parking space number s in dollars per kilogram.

• The next M lines describe the weights of the cars. The cars are numbered from 1 to M inclusive in no particular order. The k^{th} of these M lines contains a single integer W_k, the weight of car k in kilograms.

• The next $2 * M$ lines describe the arrivals and departures of all cars in chronological order. A positive integer i indicates that car number i arrives at the garage. A negative integer $-i$ indicates that car number i departs from the garage. No car will depart from the garage before it has arrived, and all cars from 1 to M inclusive will appear exactly twice in this sequence, once arriving and once departing. Moreover, no car will depart from the garage before it has parked (i.e., no car will leave while waiting on the queue).

OUTPUT

Your program must write to standard output a single line containing a single integer: the total number of dollars that will be earned by the garage operator today.

GRADING

For a number of tests worth 40 points there will always be at least one available parking space for every arriving car. In these cases no car will ever have to wait for a space.

EXAMPLES

Sample Input	Sample Output
3 4 2 3 5 200 100 300 800 3 2 −3 1 4 −4 −2 −1	5300

Car number 3 goes to space number 1 and pays $300 * 2 = 600$ dollars.

Car number 2 goes to space number 2 and pays $100 * 3 = 300$ dollars.

Car number 1 goes to space number 1 (which was released by car number 3) and pays $200 * 2 = 400$ dollars.

Car number 4 goes to space number 3 (the last remaining) and pays $800 * 5 = 4\ 000$ dollars.

Sample Input	Sample Output
2 4 5 2 100 500 1000 2000 3 1 2 4 −1 −3 −2 −4	16200

Car number 3 goes to space number 1 and pays $1\ 000 * 5 = 5\ 000$ dollars.

Car number 1 goes to space number 2 and pays $100 * 2 = 200$ dollars. Car number 2 arrives and has to wait at the entrance.

Car number 4 arrives and has to wait at the entrance behind car number 2.

When car number 1 releases its parking space, car number 2 parks there and pays $500 * 2 = 1\ 000$ dollars.

When car number 3 releases its parking space, car number 4 parks there and pays $2\ 000 * 5 = 10\ 000$ dollars.

Task 2.2 MECHO (proposed by Carl Hultquist)

Mecho the bear has found a little treasure the bees secret honeypot, which is full of honey! He was happily eating his newfound treasure until suddenly one bee saw him and sounded the bee alarm. He knows that at this very moment hordes of bees will emerge from their hives and start spreading around trying to catch him. He knows he has to leave the honeypot and go home quickly, but the honey is so sweet that Mecho doesnt want to leave too soon. Help Mecho determine the latest possible moment when he can leave.

Mechos forest is represented by a square grid of N by N unit cells, whose sides are parallel to the north-south and east-west directions. Each cell is occupied by

a tree, by a patch of grass, by a hive or by Mechos home. Two cells are considered adjacent if one of them is immediately to the north, south, east or west of the other (but not on a diagonal). Mecho is a clumsy bear, so every time he makes a step, it has to be to an adjacent cell. Mecho can only walk on grass and cannot go through trees or hives, and he can make at most S steps per minute.

At the moment when the bee alarm is sounded, Mecho is in the grassy cell containing the honeypot, and the bees are in every cell containing a hive (there may be more than one hive in the forest). During each minute from this time onwards, the following events happen in the following order:

• If Mecho is still eating honey, he decides whether to keep eating or to leave. If he continues eating, he does not move for the whole minute.

• Otherwise, he leaves immediately and takes up to S steps through the forest as described above. Mecho cannot take any of the honey with him, so once he has moved he cannot eat honey again.

After Mecho is done eating or moving for the whole minute, the bees spread one unit further across the grid, moving only into the grassy cells. Specifically, the swarm of bees spreads into every grassy cell that is adjacent to any cell already containing bees. Furthermore, once a cell contains bees it will always contain bees (that is, the swarm does not move, but it grows).

In other words, the bees spread as follows: When the bee alarm is sounded, the bees only occupy the cells where the hives are located. At the end of the first minute, they occupy all grassy cells adjacent to hives (and still the hives themselves). At the end of the second minute, they additionally occupy all grassy cells adjacent to grassy cells adjacent to hives, and so on. Given enough time, the bees will end up simultaneously occupying all grassy cells in the forest that are within their reach.

Neither Mecho nor the bees can go outside the forest. Also, note that according to the rules above, Mecho will always eat honey for an integer number of minutes.

The bees catch Mecho if at any point in time Mecho finds himself in a cell occupied by bees.

TASK

Write a program that, given a map of the forest, determines the largest number of minutes that Mecho can continue eating honey at his initial location, while still being able to get to his home before any of the bees catch him.

CONSTRAINTS

$1 \leq N \leq 800$ – The size (side length) of the map

$1 \leq S \leq 1\,000$ – The maximum number of steps Mecho can take in each minute

INPUT

Your program must read from standard input the following data:

• The first line contains the integers N and S, separated by a space.

• The next N lines represent the map of the forest. Each of these lines contains N characters with each character representing one unit cell of the grid. The possible characters and their associated meanings are as follows:

T denotes a tree

G denotes a grassy cell

M denotes the initial location of Mecho and the honeypot, which is also a grassy cell

D denotes the location of Mechos home, which Mecho can enter, but the bees cannot.

H denotes the location of a hive

NOTE: It is guaranteed that the map will contain exactly one letter M, exactly one letter D and at least one letter H. It is also guaranteed that there is a sequence of adjacent letters G that connects Mecho to his home, as well as a sequence of adjacent letters G that connects at least one hive to the honeypot (i.e., to Mechos initial location). These sequences might be as short as length zero, in case Mechos home or a hive is adjacent to Mechos initial location. Also, note that the bees cannot pass through or fly over Mechos home. To them, it is just like a tree.

OUTPUT

Your program must write to standard output a single line containing a single integer: the maximum possible number of minutes that Mecho can continue eating honey at his initial location, while still being able to get home safely.

If Mecho cannot possibly reach his home before the bees catch him, the number your program writes to standard output must be -1 instead.

GRADING

For a number of tests, worth a total of 40 points, N will not exceed 60.

EXAMPLES

Sample Input	Sample Output
7 3 TTTTTTT TGGGGGT TGGGGGT MGGGGGD TGGGGGT TGGGGGT THHHHHT	1

After eating honey for one minute, Mecho can take the shortest path directly to the right and he will be home in another two minutes, safe from the bees.

Sample Input	Sample Output
7 3 TTTTTTT TGGGGGT TGGGGGT MGGGGGD TGGGGGT TGGGGGT TGHHGGT	2

After eating honey for two minutes, Mecho can take steps $\rightarrow\uparrow\rightarrow$ during the third minute, then steps $\rightarrow\rightarrow\rightarrow$ during the fourth minute and steps $\downarrow\rightarrow$ during the fifth minute.

Task 2.3 REGIONS (proposed by Long Fan and Richard Peng)

The United Nations Regional Development Agency (UNRDA) has a very well defined organizational structure. It employs a total of N people, each of them coming from one of R geographically distinct regions of the world. The employees are numbered from 1 to N inclusive in order of seniority, with employee number 1, the Chair, being the most senior. The regions are numbered from 1 to R inclusive in no particular order. Every employee except for the Chair has a single supervisor. A supervisor is always more senior than the employees he or she supervises.

We say that an employee A is a manager of employee B if and only if A is Bs supervisor or A is a manager of Bs supervisor. Thus, for example, the Chair is a manager of every other employee. Also, clearly no two employees can be each others managers.

Unfortunately, the United Nations Bureau of Investigations (UNBI) recently received a number of complaints that the UNRDA has an imbalanced organizational structure that favors some regions of the world more than others. In order to investigate the accusations, the UNBI would like to build a computer system that would be given the supervision structure of the UNRDA and would then be able to answer queries of the form: given two different regions r_1 and r_2, how many pairs of employees e_1 and e_2 exist in the agency, such that employee e_1 comes from region r_1, employee e_2 comes from region r_2, and e_1 is a manager of e_2. Every query has two parameters: the regions r_1 and r_2; and its result is a single integer: the number of different pairs e_1 and e_2 that satisfy the above-mentioned conditions.

TASK

Write a program that, given the home regions of all of the agencys employees, as well as data on who is supervised by whom, interactively answers queries as described above.

CONSTRAINTS

$1 \leq N \leq 200\ 000$ – The number of employees
$1 \leq R \leq 25\ 000$ – The number of regions
$1 \leq Q \leq 200\ 000$ – The number of queries your program will have to answer
$1 \leq H_k \leq R$ – The home region of employee k (for $1 \leq k \leq N$)
$1 \leq S_k < k$ – The supervisor of employee k (for $2 \leq k \leq N$)
$1 \leq r_1, r_2 \leq R$ – The regions inquired about in a given query

INPUT

Your program must read from standard input the following data:

• The first line contains the integers N, R and Q, in order, separated by single spaces.

• The next N lines describe the N employees of the agency in order of seniority. The k^{th} of these N lines describes employee number k. The first of these lines (i.e., the one describing the Chair) contains a single integer: the home region H_1 of the Chair. Each of the other $N-1$ lines contains two integers separated by a single space: employee ks supervisor S_k, and employee ks home region H_k.

INTERACTION

After reading the input data, your program must start alternately reading queries from standard input and writing query results to standard output. The Q queries must be answered one at a time; your program must send the response to the query it has already received before it can receive the next query.

Each query is presented on a single line of standard input and consists of two different integers separated by a single space: the two regions r_1 and r_2.

The response to each query must be a single line on standard output containing a single integer: the number of pairs of UNRDA employees e_1 and e_2, such that e_1s home region is r_1, e_2s home region is r_2 and e_1 is a manager of e_2.

NOTE: The test data will be such that the correct answer to any query given on standard input will always be less than 1 000 000 000.

IMPORTANT NOTE: In order to interact properly with the grader, your program needs to flush standard output after every query response. It also needs to avoid accidentally blocking when reading standard input, as might happen for instance when using `scanf("%d\n")`. Please see the technical info sheet for instructions on how to do this properly.

GRADING

For a number of tests, worth a total of 30 points, R will not exceed 500.

For a number of tests, worth a total of 55 points, no region will have more than 500 employees.

The tests where both of the above conditions hold are worth 15 points.

The tests where at least one of the two conditions holds are worth 70 points.

EXAMPLE

Sample Input	Sample Output
6 3 4 1 1 2 1 3 2 3 2 3 5 1 1 2	
	1 [flush standard output]
1 3	
	3 [flush standard output]
2 3	
	2 [flush standard output]
3 1	
	1 [flush standard output]

TESTING

If you would like to test your solution through the contest systems test interface, the input file you provide should include both the input data and all queries, as illustrated in the sample input above.

Task 2.4 SALESMAN (proposed by Velin Tzanov)

The traveling salesman has decided that optimally scheduling his trips on land is an intractable computational problem, so he is moving his business to the linear world of the Danube River. He has a very fast boat that can get him from anywhere to anywhere along the river in no time, but unfortunately the boat has terrible fuel consumption. It costs the salesman U dollars for every meter traveled upstream (towards the source of the river) and D dollars for every meter traveled downstream (away from the source of the river).

There are N trade fairs that the salesman would like to visit along the river. Each trade fair is held for one day only. For each trade fair X, the traveling salesman knows its date T_X, measured in the number of days since he purchased his boat. He also knows the fairs location L_X, measured as the distance in meters

from the source of the river downstream to the fair, as well as the number of dollars M_X that the salesman is going to gain if he attends this trade fair. He has to start and end his journey at his waterfront home on the river, which is at location S, measured also in meters downstream from the source of the river.

Help the traveling salesman choose which trade fairs to attend (if any) and in what order, so that he may maximize his profit at the end of his travels. The traveling salesmans total profit is defined as the sum of the dollars he gained at the fairs he attended, minus the total sum of dollars he spent traveling up and down the river.

Keep in mind that if trade fair A is held earlier than trade fair B, the salesman can visit them only in this order (i.e., he cannot visit B and then visit A). However, if two fairs are held on the same date, the salesman can visit them both in any order. There is no limit to how many fairs the salesman can visit in a day, but naturally he can't visit the same fair twice and reap the gains twice. He can pass through fairs he has already visited without gaining anything.

TASK

Write a program that, given the date, location and profitability of all fairs, as well as the location of the traveling salesmans home and his costs of traveling, determines the maximum possible profit he can make by the end of his journey.

CONSTRAINTS

$1 \leq N \leq 500\ 000$ – The number of fairs
$1 \leq D \leq U \leq 10$ – The cost of traveling one meter upstream (U) or downstream (D)
$1 \leq S \leq 500\ 001$ – The location of the salesmans home
$1 \leq T_k \leq 500\ 000$ – The day on which fair k is held
$1 \leq L_k \leq 500\ 001$ – The location of fair k
$1 \leq M_k \leq 4\ 000$ – The number of dollars the salesman would earn if he attends fair k

INPUT

Your program must read from standard input the following data:

• The first line contains the integers N, U, D and S, in this order, separated by single spaces.

• The next N lines describe the N fairs in no particular order. The k^{th} of these N lines describes the k^{th} fair and contains three integers separated by single

spaces: the day of the fair T_k, its location L_k, and its profitability for the salesman M_k.

NOTE: All locations given in the input will be different. That is to say, no two fairs will happen at the same location and no fair will happen at the salesmans home.

OUTPUT

Your program must write to standard output a single line containing a single integer: the maximum profit the salesman can possibly make by the end of his journey.

GRADING

For a number of tests, worth a total of 60 points, no two fairs will be held on the same day. For a number of tests, worth a total of 40 points, none of the numbers in the input will exceed 5 000.

The tests where both of the above conditions hold are worth 15 points.

The tests where at least one of the two conditions holds are worth 85 points.

EXAMPLE

Sample Input	Sample Output
4 5 3 100 2 80 100 20 125 130 10 75 150 5 120 110	50

An optimal schedule would visit fairs 1 and 3 (the ones at locations 80 and 75). The sequence of events and their associated profits and costs would be as follows:

• The salesman travels 20 meters upstream at a cost of 100 dollars. Profit so far: -100

• He attends fair number 1 and earns 100. Profit so far: 0

• He travels 5 meters upstream at a cost of 25. Profit so far: -25

• He attends fair number 3 where he earns 150. Profit so far: 125

• He travels 25 meters downstream to return home at a cost of 75. Profit at the end: 50

TECHNICAL INFO SHEET (day 2)

These pages contain helpful information on how to avoid slow input/output performance with C++ streams (cin / cout), how to use 64-bit data types (variables) and how to properly communicate with the grader on interactive tasks. They also include reference for what options are given to the compilers and what stack limitations are in place.

Slow Input / Output with C++ Streams

When solving tasks with very large amounts of input / output data, you may notice that C++ programs using the cin and cout streams are much slower than equivalent programs that use the scanf and printf functions for input and output processing. Thus, if you are using the cin / cout streams we strongly recommend that you switch to using scanf / printf instead. However, if you still want to use cin / cout, we recommend adding the following line at the beginning of your program:

```
ios::sync_with_stdio(false);
```

and also making sure that you never use `endl`, but use `\n` instead.

Please note, however, that including `ios::sync_with_stdio(false)` breaks the synchrony between cin / cout and scanf / printf, so if you are using this, you should never mix usage of cin and scanf, nor mix cout and printf.

64-bit Data Types

For some tasks you may need to deal with numbers too large to fit in 32 bits. In these cases, you would have to use a 64-bit integer data type, such as `long long` in C/C++ or `int64` in Pascal. Here is some sample code that illustrates the usage of these data types:

C/C++

```
int main(void) {
    long long varname;
    scanf("%lld", &varname);
    // Do something with the varname variable
    printf("%lld\n", varname);
    return 0;
}
```

Pascal

```
var
    varname: Int64;
begin
    read(varname);
    { Do something with the varname variable }
    writeln(varname);
end.
```

Communication with Grader on Interactive Tasks

Whenever you solve an interactive task, you always need to flush the buffer of your output after every new line printed on the output. Here is some code to illustrate how to do this under C, C++ and Pascal:

C or C++ with scanf / printf

```
fflush(stdout);
```

In addition, when using `scanf`, you must avoid reading input in a way that blocks the execution of your program while waiting for white space on standard input. Such blocking might happen if you use `scanf` with a first argument that ends with a space or a new line. In particular, you can safely use `%d` as a `scanf` argument, but you should NOT use `%d` (with a trailing space) or `%d\n` (with a trailing new line).

C++ with cin / cout

```
cout << flush;
```

Pascal

```
flush(output);
```

Compiler Options

The following commands will be used to compile solutions of batch and interactive tasks (say the task name is abc):

C

```
gcc -o abc abc.c -std=gnu99 -O2 -s -static -lm -x c
```

C++

```
g++ -o abc abc.cpp -O2 -s -static -lm -x c++
```

Pascal

```
fpc -O2 -XS -Sg abc.pas
```

Stack Limitations

Whenever your program is executed through the contest system, the stack size will only be limited by the memory limit for the corresponding task.

Task Overview Sheet (Day 2)

	Garage	Mecho	Regions	Salesman
Type	Batch	Batch	Reactive	Batch
Detailed Feedback	Full	Partial	None	Partial
Time Limit (per test case)	1 second	1 second	8 seconds	3 seconds
Memory Limit (per test case)	32 MB	64 MB	128 MB	128 MB
Points	100	100	100	100

NOTE: C++ programmers should be aware that using C++ streams (cin / cout) may lead to I/O bottlenecks and substantially slower performance. Please see the technical info sheet for ways to avoid this.