Bergen Open 2018

November 10, 2018

Problems

A	Awkward Party
B	Backpack Buddies
C	Counting Clauses
D	Dice and Ladders
E	Expecting Rain
F	Fishmongers
G	Gameworld Tornado
H	Hidden Words
I	ISP Merger
J	Joint Attack
K	Keyboards in Concert
L	License to Launch

Do not open before the contest has started.

Advice, hints, and general information

- The problems are not sorted by difficulty.
- Your solution programs must read input from *standard input* (e.g. `System.in` in Java or `cin` in C++) and write output to *standard output* (e.g. `System.out` in Java or `cout` in C++). For further details and examples, please refer to the documentation in the help pages for your favorite language on Kattis.
- For information about which compiler flags and versions are used, please refer to the documentation in the help pages for your favorite language on Kattis.
- Your submissions will be run multiple times, on several different inputs. If your submission is incorrect, the error message you get will be the error exhibited on the first input on which you failed. E.g., if your instance is prone to crash but also incorrect, your submission may be judged as either "wrong answer" or "run time error", depending on which is discovered first. The inputs for a problem will always be tested in the same order.
- If you think some problem is ambiguous or underspecified, you may ask the judges for a clarification request through the Kattis system. The most likely response is "No comment, read problem statement", indicating that the answer can be deduced by carefully reading the problem statement or by checking the sample test cases given in the problem, or that the answer to the question is simply irrelevant to solving the problem.
- In general we are lenient with small formatting errors in the output, in particular whitespace errors within reason. But not printing any spaces at all (e.g. missing the space in the string "`1 2`" so that it becomes "`12`") is typically not accepted.

  The safest way to get accepted is to follow the output format exactly.
- For problems with floating point output, we only require that your output is correct up to some error tolerance.

  For example, if the problem requires the output to be within either absolute or relative error of 10^{-4}, this means that

  - If the correct answer is 0.05, any answer between 0.0499 and $.0501$ will be accepted.
  - If the correct answer is 500, any answer between 499.95 and 500.05 will be accepted.

  Any reasonable format for floating point numbers is acceptable. For instance, "17.000000", "0.17e2", and "17" are all acceptable ways of formatting the number 17. For the definition of reasonable, please use your common sense.

Problem A
Awkward Party
Problem ID: awkwardparty

CC0, by strecosa via Pixabay

Martin has invited everyone he knows to celebrate his 535th birthday, and a whopping n people from all over the world have accepted the invitation.

When deciding the seating arrangement, Martin's mother Margarethe have decided that all the guests should be seated with *maximum awkwardness*; this is to ensure that nobody has anything meaningful to discuss during dinner, and everyone would instead silently enjoy her rather tasty coriander soup (as the saying goes; "when the food is good, conversation dies").

Margarethe knows that awkwardness is maximized if the guests are seated in a long row along a single table, in such a way that nobody sits next to someone speaking the same language as themselves. Better yet, she has defined the *awkwardness level* of a seating arrangement to be the minimum number of seats separating any two guests speaking the same language. If no two people speak the same language, the awkwardness level is defined to be n (the number of guests). Two seats next to each other are said to be separated by 1.

Given the languages spoken in a proposed seating arrangement, can you help Margarethe determine the awkwardness level?

Input

The first line contains an integer n ($1 \leq n \leq 100\,000$) denoting the number of guests. On the second line follows n integers, the i'th of which x_i ($0 \leq x_i \leq 10^9$) indicating the language spoken by the guest seated at position i in the proposed arrangement (each guest speaks exactly one language).

Output

A single integer, the awkwardness level of the proposed seating arrangement.

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

Sample Input 2	Sample Output 2
3 1 2 3	3

This page is intentionally left blank.

Problem B
Backpack Buddies
Problem ID: backpackbuddies

The prudent Mr. Day and adventurous Dr. Knight have decided to be backpack buddies and go trekking in the gorgeous Norwegian mountains. Of course they will use the trails marked by the Norwegian Trekking Association (DNT), like any sane trekkers would; such trails all start and end at some cabin where it is possible to lay over. Mr. Day for one likes sleeping comfortably, so he immediately suggested that they sleep in a cabin every night.

However, Dr. Knight was of a different opinion; she argued that they'd arrive at their destination much quicker if they went for as long as possible each day, and then slept in tents in between two cabins if need be. Even though Mr. Day agreed that reaching the destination as quickly as possible was desirable, he was not willing to sacrifice the comfort of a proper bed.

After a fierce argument, Mr. Day and Dr. Knight decided to split up, each to follow their own strategy. Assuming they both walk at the same pace, how long will Dr. Knight need to wait at the final destination before Mr. Day shows up?

Input

The first line of input contains two integers, n ($1 \leq n \leq 100\,000$) the number of cabins, and m ($1 \leq m \leq 100\,000$) the number of trails. Then follows m lines, each describing a trail. The i'th such line contains three integers, u_i, v_i and d_i ($0 \leq u_i, v_i < n$, $0 \leq d_i \leq 12$) meaning that there is a trail between cabin u_i and cabin v_i which will take exactly d hours to traverse. (There can be more than one trail between two cabins, and some trails could lead back to the same cabin from where it started).

Mr. Day and Dr. Knight start in cabin 0 and their destination is cabin $n-1$. Each day they start walking at 08:00 in the morning, and then walks non-stop for (at most) 12 hours before settling down for the night.

Output

Output a single integer, the number of hours Dr. Knight needs to wait at the destination before Mr. Day shows up.

Sample Input 1	Sample Output 1
5 6 0 1 2 0 3 8 1 2 11 2 3 5 2 4 2 4 3 9	4

Sample Input 2

```
3 2
0 1 2
1 2 12
```

Sample Output 2

```
10
```

Problem C
Counting Clauses
Problem ID: countingclauses

It's time for the annual 3-SAT competition, where the contestants compete to answer as many instances of 3-SAT as possible within the time limit. 3-SAT is a classic NP-complete problem, where you are given a boolean formula in *conjunctive normal form*, in which we have a set of *clauses* each consisting of exactly three *literals*. Each literal refer either positively or negatively to a *variable*, which can be assigned a value of either `True` or `False`. The question is whether there exists an assignment to the variables such that every clause evaluates to `True`. No clause will contain duplicates of a literal (however it is possible that a clause contain both $\neg x_i$ and x_i). An example of a 3-SAT instance is shown below (from sample input 1):

$$(\neg x_1 \vee x_2 \vee x_3) \wedge (\neg x_1 \vee \neg x_2 \vee x_3) \wedge (x_1 \vee \neg x_2 \vee x_3) \wedge (x_1 \vee \neg x_2 \vee \neg x_3) \wedge (x_1 \vee x_2 \vee \neg x_3)$$

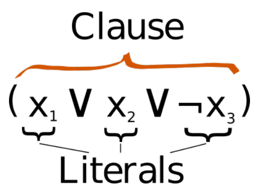


Illustration of the terminology

Øyvind is a judge in the competition, responsible for verifying the quality of problem instances crafted by the other judges before the contest starts. Øyvind hates 3-SAT instances with less than eight clauses – as these are always satisfiable they provide no real challenge for the contestants. Therefore, he will deem such problem instances to be unsatisfactory. Whenever Øyvind encounters an instance with eight or more clauses he knows that it is a real challenge to figure out whether this instance is satisfiable or not – and therefore he will judge these problem instances to be satisfactory. Given an instance of 3-SAT, can you help find Øyvind's judgement?

Input

The input is a single instance of the 3-SAT problem. The first line is two space-separated integers: m ($1 \leq m \leq 20$), the number of clauses and n ($3 \leq n \leq 20$), the number of variables. Then m clauses follow, one clause per line. Each clause consists of 3 distinct space-separated integers in the range $[-n, n] \setminus \{0\}$. For each clause, the three values correspond to the three literals in the clause. If the literal is negative, that means that the clause is satisfied if the corresponding variable is set to `False`, and if it is positive the clause is satisfied if the variable is set to `True`.

Output

Print "`satisfactory`" on a single line if Øyvind finds the 3-SAT instance to be satisfactory, and "`unsatisfactory`" otherwise.

Sample Input 1

```
5 3
-1 2 3
-1 -2 3
1 -2 3
1 -2 -3
1 2 -3
```

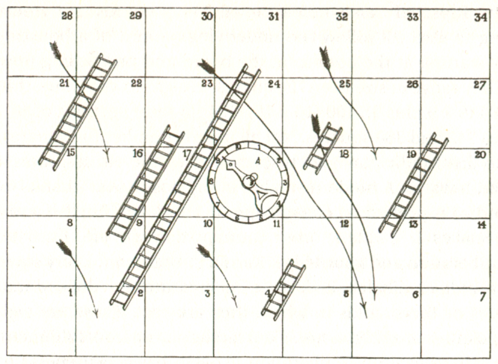
Sample Output 1

```
unsatisfactory
```

Sample Input 2	Sample Output 2

`8 3`
`1 2 3`
`1 2 -3`
`1 -2 3`
`1 -2 -3`
`-1 2 3`
`-1 2 -3`
`-1 -2 3`
`-1 -2 -3`
	`satisfactory`

Problem D
Dice and Ladders
Problem ID: diceandladders



Public Domain, via Wikimedia Commons

The ladder game is a fun children's game, the rules are as follows: You start at cell number 1 and each round you roll a dice and move the number specified by the dice. If you end on a cell with a ladder starting from this cell then you have to follow the ladder in its direction a single time, that is if the ladder ends at a cell where a new ladder starts you will not follow the new ladder. The game ends when you move to or past the last cell.

Your task is to find the minimum number of dice-rolls required to finish the game with a probability of at least p.

Input

The first line contains three integers r ($3 \leq r \leq 8$), c ($3 \leq c \leq 8$) and k ($0 \leq k \leq 50$) on one line, the number of rows, columns and ladders respectively. The second line contains a single floating point number p ($0 < p < 1$) as described above (with at most 6 digits after the decimal point).

Then follows k lines, each describing a ladder. The i'th of these lines contains two integers s_i ($2 \leq s_i < r \cdot c$) and e_i ($1 \leq e_i \leq r \cdot c$), the starting cell and ending cell of the ladder i, respectively. Two ladders will never start at the same cell, but multiple ladders may end at the same cell. The cells are numbered like in the illustration, meaning cell 1 is in the bottom left corner and there are c more cells in the same row. Cell $c + 1$ starts to the left in the second row, and so on.

It is guaranteed that it is possible to finish the game with a probability of p in less than 10^8 dice-rolls. The input is also constructed in such a way that the expected number of dice-rolls such that you finish the game with a probability of p is the same as the expected number of dice-rolls such that you finish the game with a probability of $p \pm 10^{-9}$

Output

A single integer, the minimum number of dice-rolls required such that you finish the game with a probability at least p.

Sample Input 1

```
3 3 2
0.4
2 3
4 7
```

Sample Output 1

```
2
```

This page is intentionally left blank.

Problem E
Expecting Rain
Problem ID: expectingrain


CC0, misterfarmer via Pixabay

The Bluewater Geocloud Organization (BGO) has recently developed brand new software able to predict with pinpoint precision at which second any particular cloud will start and stop raining, and with what intensity. There is, however, some uncertainty about how a cloud will move around; for each zip code, each cloud will be over that zip code with some probability.

You have scraped some information about your zip code from the BGO website, and want to use it to plan your walk to the bus stop. You wish to minimize the expected amount of rain that would fall on you. To reach the bus you must get to the bus stop within t seconds from now. You have timed your walking speed to be exactly $1\frac{m}{s}$.

To complicate matters, some parts of the walk to the bus are covered by roofs where it might be beneficial to make shorts breaks whilst waiting for the worst rain to pass. Your front door (at d meters from the bus stop) is always under a roof – but the bus stop need not be.

Input

The first line of input is four space-separated integers: d ($1 \leq d \leq 1\,000$), the distance to the bus stop in meters, t ($d \leq t \leq 10\,000$) the time until the bus leaves, c ($0 \leq c \leq 1\,000\,000$), the number of clouds tracked by BGO, and finally r ($0 \leq r \leq d$), the number of roofs. The next c lines describe the clouds; the i'th such line contains four numbers s_i, e_i, p_i and a_i describing the i'th cloud:

- s_i ($0 \leq s_i < t$) is an integer giving the number of seconds until the cloud starts its raining period,
- e_i ($s_i < e_i \leq t$) is an integer giving the number of seconds until the cloud ends its raining period,
- p_i ($0 \leq p_i \leq 1$) is a real number (with at most 6 digits after the decimal point) giving the probability that the cloud is in your zip code during its raining period, and
- a_i ($0 \leq a_i \leq 100$) is an integer indicating the amount of rain the cloud will release during its raining period, given as nm per second.

Finally r roof segments follow, each on its own line; the j'th such line contains two integers x_j and y_j ($0 \leq x_j < y_j \leq d + 1$), indicating that there is a roof segment starting at distance x_j away from home, ending at distance y_j away from home along the route to the bus stop. Both your home, the bus stop an the entire route between them are in the same zip code. No two roofs overlap, however one roof may start at the same exact location as another ends.

Output

The output consists of single a real value, the minimum amount of rain in nm you can expect on your route if you reach the bus stop in time. Answers with absolute or relative precision 10^{-5} of the actual value will be accepted.

Sample Input 1	Sample Output 1
20 60 2 1 5 15 0.33333 30 22 60 0.66666 70 0 10	466.662

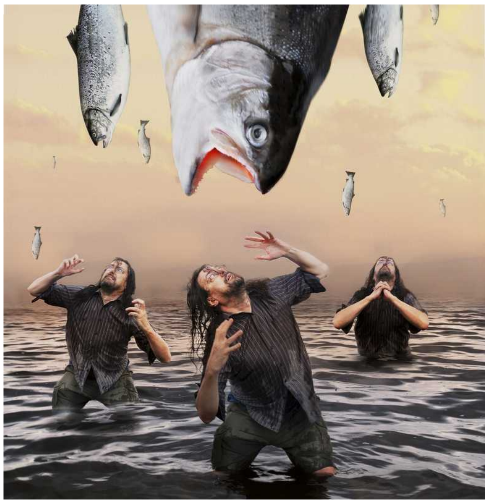
Sample Input 2	Sample Output 2
3 4 2 1 1 3 0.25 8 2 4 0.66667 15 1 2	10.00005

Sample Input 3	Sample Output 3
3 4 1 0 0 2 0.25 8	2.0

Sample Input 4	Sample Output 4
3 5 2 1 0 3 0.125 32 2 5 0.5 32 3 4	28.0

Problem F
Fishmongers
Problem ID: fishmongers

You fish fish.
You hate fish.
You love monies.
Therefore sell fish.
To fishmongers.
For maximum profit.



CC0, by Geir Fløde via Pixabay

Input

The first line of input contains two integers n ($1 \leq n \leq 100\,000$), the number of fish you have to sell, and m ($1 \leq m \leq 100\,000$), the number of fishmongers. On the second line follows n space-separated integers $w_1, w_2, \ldots, w_n$, the weight of each of your fish in kilograms ($1 \leq w_i \leq 100\,000$). Finally, there are m lines, the j'th of which consisting of two integers x_j ($1 \leq x_j \leq 100\,000$) and p_j ($1 \leq p_j \leq 100\,000$), respectively indicating how many fish the j'th fishmonger wants to buy and how many monies he will pay per kilogram.

Output

A single integer, the maximum number of monies you can obtain by selling your fish to the fishmongers.

Sample Input 1	Sample Output 1
4 3 1 2 7 5 2 4 1 5 3 3	66

This page is intentionally left blank.

Problem G
Gameworld Tornado
Problem ID: gameworld

CC-BY 4.0, LootHunter by DragonDePlatino, via Wikimedia Commons

The Brilliant Game Overseers (BGO) is building a brand new game. The investors have noted that large gameworlds is a selling point these days, so in order to appease the them, a large gameworld with lots of interesting activities has been planned. In addition to a large map showing the entire world in all its glory, a number of smaller maps in various dimensions have been designed showing the layout of smaller features in the Game. Because features should be spread out a little, these maps can overlap.

Woe upon woe, the day after all maps had finally been finished, a tornado blew through the offices of BGO, and now the master map has been lost, as well as a number of the smaller maps. Desperate to salvage as much as possible of the Game, you have been tasked with trying to piece together as much as possible of the original map, using the feature maps that were left behind by the tornado. Luckily all the maps have their coordinates marked and are axis aligned, so you figure a good start will be to find out how large is the area (in pixels) covered by the remaining maps.

Input

The input starts with an integer n ($1 \leq n \leq 100\,000$) denoting the number of axis aligned rectangular maps you have left. Then follows n lines representing the maps, each with four integers x_1, y_1, x_2, y_2 where $0 \leq x_1, y_1, x_2, y_2 \leq 10^9$. x_1 and y_1 are the coordinates of the lower left corner of the rectangle, and x_2 and y_2 are the upper right corner of the rectangle, i.e. $x_1 < x_2$ and $y_1 < y_2$.

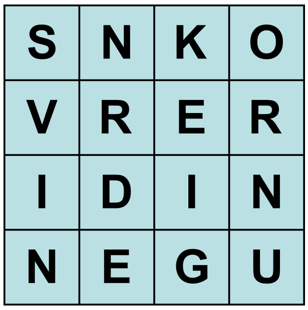
Output

The total area of the game world covered by the remaining maps.

Sample Input 1	Sample Output 1
2 2 2 4 4 3 3 5 5	7

This page is intentionally left blank.

Problem H
Hidden Words
Problem ID: hiddenwords



Ingrid is solving the Saturday newspaper *Hidden Words in a Grid* -puzzle, but is finding it a bit tedious to do by hand. Luckily Ingrid knows how to program, and has written a neat image recognition routine that converts a picture of the puzzle into a nice text-based format. However, she is struggling with writing the program that actually solves the puzzle – can you help her out?

A word is contained within a h by w grid if the word can be constructed by starting in a cell in the grid and walking from there to neighboring unvisited cells. A cell neighbors another cell if it is adjacent, not including diagonal movement. Given such a grid and a list of words, decide how many of the words in the list are contained in the grid.

Input

The first line consists of two integers h and w ($1 \leq h, w \leq 10$), the height and width of the grid. Then h lines follow, each containing a string of length w consisting exclusively of uppercase letters describing one row of the grid. Then follows a line with a single integer n ($1 \leq n \leq 100\,000$), indicating the number of words Ingrid is looking for. Finally the n words follow, each on a separate line. None of these words are longer than 10 characters.

Output

The output consists of a single number, the number of words underneath the grid that are contained in the grid.

Sample Input 1	Sample Output 1
4 4 SNKO VRER IDIN NEGU 3 KORN NEDI DER	2

This page is intentionally left blank.

Problem I
ISP Merger
Problem ID: ispmerger

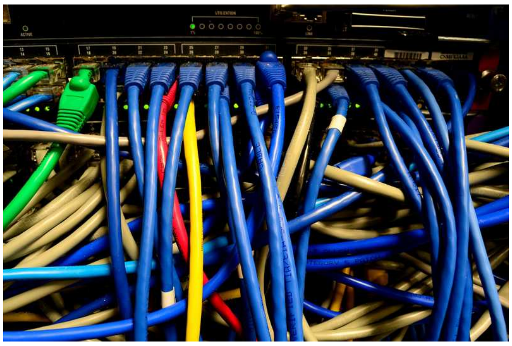


Illustration of secret input. CC0, via pxhere

A number of regional Internet Service Providers (ISPs), both big and small have recently been forced into a merger by the government, in an effort to improve service for all. Of course this has been decided without consulting you, the chief network infrastructure officer, and a deadline for when the ISPs should be merged have already been set.

You have a set of n servers, each with a limited number of network sockets that can be used for connecting physically to other servers. Some servers are already linked up in the existing network architecture, and if server 0 is linked to server 2, then 2 is also linked to server 0 (as you use full duplex ethernet Cat6 cables). No server is directly connected to itself, and no two servers are directly linked with more than one connection.

You want to connect the servers to form a single network, such that all servers can reach each other through some sequence of connections. To make the set deadline, you have estimated that you only have time to make k edits to the existing network infrastructure. An *edit* is either to remove an existing connection between two servers, or to add a new connection between two servers.

Can you connect all the servers to the same network using at most k edits, within the limitations on the number of network sockets in each server?

Input

The first line of the input is three space separated integers n ($1 \leq n \leq 100\,000$), the number of servers, m ($0 \leq m \leq 200\,000$), the number of existing connections and k ($0 \leq k \leq 50\,000$), the number of edits you have time to make. Then follows a line with n integers $c_0, c_1, \ldots, c_i, \ldots, c_{n-1}$, with the i'th number giving the number of network sockets for the i'th server (for all i the capacity is bounded by $1 \leq c_i < n$). Then m lines follow, with two integers u_j and v_j each, giving the id's of two servers that are connected in the old network architecture. Servers are 0-indexed, i.e. for every j, it holds that $0 \leq u_j, v_j < n$. A server will never be connected to more servers than it has connection sockets.

Output

Output "`yes`" on a single line if the servers can be connected to one network by making k or less edits, and "`no`" if it is not possible.

Sample Input 1

```
4 5 2
3 3 3 3
0 1
0 3
1 3
1 2
2 3
```

Sample Output 1

```
yes
```

Sample Input 2

```
5 4 4
1 1 2 2 2
0 1
2 3
3 4
4 2
```

Sample Output 2

```
yes
```

Sample Input 3

```
3 0 3
1 1 1
```

Sample Output 3

```
no
```

Problem J
Joint Attack
Problem ID: jointattack

Public Domain, Bill Ingalls via nasa.gov

General Torstein has sent the x-coordinate for the next joint attack and is expecting you to promptly follow his orders in order to avoid impeding doom. Unfortunately Torstein hates numbers with more than 2 digits and loves horizontal line segments, and has therefore sent the coordinate as a continued fraction, i.e.

$$x = x_0 + \cfrac{1}{x_1 + \frac{1}{\cdots}}$$

Your rocket launcher only accepts coordinates as reduced fractions, so you need to quickly compute the correct numbers to feed it in order to commence the attack. Hurry! Failure may have dire consequences!

Input

The first line of output is one integer n ($1 \leq n < 40$), the number of coefficients in the continued fraction, followed by a line with n integers ($1 \leq x_i < 100$) the coefficients of x.

Output

The coordinate x as a reduced fraction. It is guaranteed that the numerator and denominator are both less than 10^{18}.

Sample Input 1	Sample Output 1
2 2 3	7/3

This page is intentionally left blank.

Problem K
Keyboards in Concert
Problem ID: keyboardconcert

Olav has some electronic keyboards and would like to play a tune. Unfortunately all of Olav's keyboards are broken so each of them can only play some of the notes. By switching which instrument he is using he will be able to play the whole tune, but moving keyboards around is annoying so he would like to minimize the amount of times he has to switch. Can you help Olav figure out the minimum number of keyboard switches needed to play the entire song?

Input

The first line of input is two space separated integers; n $(1 \leq n \leq 1\,000)$ the number of instruments, and m $(1 \leq m \leq 1\,000)$, the number of notes in the tune. This is followed by n lines, each starting with an integer k_i $(1 \leq k_i \leq 1\,000)$, the number of notes playable by instrument i, followed by k_i pairwise distinct integers $\ell_1, \ell_2, \ldots, \ell_{k_i}$, the notes that instrument i can play $(1 \leq \ell_j \leq 1\,000)$. Finally, there is a line with m space-separated integers – the notes of the tune in order.

Output

The minimum number of times Olav needs to switch the instrument he is using during the tune.

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

Problem L
License to Launch
Problem ID: licensetolaunch

A Rocket launch (Public Domain, NASA via Wikimedia Commons)

Birk has made a new shiny rocket and just received his licence from the Bluesky Global Order (BGO) to launch anytime within the next n days. He is, however, worried that the rocket will hit space junk on its way. In order minimize the risk of a collision, Birk has made a model of how many pieces of space junk there will be for each of the next n days. He decided that he will launch his rocket on the day with the least space junk, and if there are multiple days with the same amount of space junk he of course wants to send his rocket up as early as possible.

Can you help Birk find out how many days he has to wait until he sends up his rocket?

Input

On the first line there is a single integer n ($1 \leq n \leq 100\,000$) the number of days for which the launch license is valid. On the second line follows n integers between 0 and 10^9 where the i'th integer is the amount of space junk on the i'th day. The first day is day $i = 0$.

Output

Output a single integer, the number of days Birk needs to wait before he launches his rocket.

Sample Input 1	Sample Output 1
5 3 4 1 7 2	2