A – Letter Factory

Note: This is the first of the Letter Factory problems. For the harder version, see "Letter Factory (Harder)".

Sachin works at the Letter Factory and has questionable work ethics. The foreman at the letter factory has given Sachin a string s that needs to be mutated using a sequence of specific instructions. In particular, in each instruction, the foreman will specify pairs of integers a and b, and Sachin needs to transform s by either "incrementing" or "decrementing" all the letters of s between the ath letter and the bth letter (inclusive) by a single character. "Increment"ing a letter means turning the letters into the following letter (i.e., lexicographically next character); for example, 'a' would become 'b', 'b' would become 'c', …, and 'z' would wrap back around and become 'a'. Similarly, "decrement"ing 'z' would become 'y', 'y' would become 'x', …, and 'a' would wrap around to become 'z'. Once *all* the instructions in the sequence are performed in sequential order, Sachin needs to output the entire modified string. As was mentioned earlier, Sachin's work ethic is (at the very least) questionable, and so he would like you to do the heavy lifting for him as he would rather spend his time playing *Dress to Impress* on Roblox.

Input

The first line of input contains a single integer, q, $1 \leq q \leq 10^5$, which is the number of queries the foreman makes on the given string.

The second line of input contains a single string, s, $s, 1 \leq |s| \leq 10^5$, which is the string the foreman has given Sachin. It will contain *only* lowercase letters of the English alphabet.

The next q lines of input each contain three space-separated integers, a, b, d, where $0 \leq a \leq b < |s|$ and $0 \leq d \leq 1$, each triple representing a single instruction from the sequence of instructions given by the foreman. Here a and b represent the left and right endpoints of the portion of the string s that need to be modified. If d is 0, a decrement is to be performed. Otherwise, an increment is to be performed.

Output

The first and only line of output should contain the string s once it has been modified after performing the entire sequence of instructions.

Sample Input 1

```
abc
0 1 0
1 2 1
0 2 1
```

Sample Output 1

```
ace
```

Sample Input 2

```
2
dztz
0 0 0
1 1 1
```

Sample Output 2

```
catz
```

B – Painstaking Paths

Gian just *adores* traversing trees! Gian would like to remind you that a tree is a special type of acyclic graph in which there exists *exactly* one path between any pair of nodes. As part of a challenge, Gian will give you a tree with n nodes where every node is assigned a unique positive integer between 1 and n. He will also give you a positive integer, k, and would like you to give him the k-th lexicographically smallest path in this tree. Since every path in the tree is simply an ordered list of nodes on the path, the rules for "lexicographical comparison" (similar to comparing words in a dictionary) are given as follows:

1. A path P is lexicographically smaller than Q if the first node of P is numbered smaller than the first node of Q. If the first nodes of P and Q are the same, then P is lexicographically smaller than Q if the remainder of path P after removing the first node is lexicographically smaller than the remainder of Q.

2. Note that an empty path (with no nodes) is lexicographically smaller than any non-empty path.

Input

The first line of input contains two integers $1 \leq n \leq 10^5$, $1 \leq k \leq n^2$ the number of nodes in the tree, and the integer k as described in the problem statement. The ith line of the next n lines of input contains one integer $1 \leq v \leq 10^9$: the value of the ith node. All nodes will possess unique values.
Finally, $n-1$ lines follow, containing a single pair of space-separated integers $1 \leq a, b \leq n$, denoting the edges in the tree.

Output

The first and only line of output should contain two integers i and j, separated by a single space, denoting the kth smallest path in this tree, given as a sequence of nodes each separated by a single space.

Sample Input 1

```
5 11
4
3
5
2
1
3 4
4 1
2 5
3 2
```

Sample Output 1

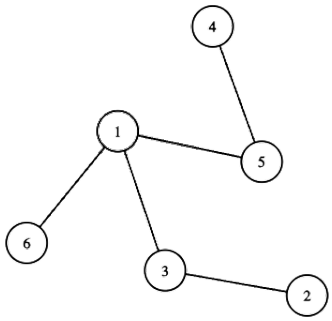
```
2
```

Sample Input 2

```
6 8
1
2
3
4
6
5
5 1
3 2
1 3
1 6
5 4
```

Sample Output 2

```
2 3
```



Graph for the second sample input. Note that the nodes' values are not necessarily equal to the nodes themselves.

Sample Input 3

```
1 1
2
```

Sample Output 3

```
1
```

C – Subsequence Scoring

Gustavo has an array of integers he cares for *very* dearly. Gustavo has undertaken the arduous (and perhaps vain) task of measuring the love he feels for his array. He spent many hours carefully crafting the perfect way to measure this love for his array, and finally, he found the perfect solution to this cumbersome problem! The method of quantifying his love is as follows: he takes this array and considers each and every one of its *subsequences*, and in each of these subsequences, he takes the maximum of all the elements in that subsequence. The sum of all these maximum values is the measure of his love for his array. We define a *subsequence* of an array to be any sequence of elements that can be derived by deleting some (possibly none) of the elements in that array.

Unfortunately, even though Gustavo is the one who devised this method of quantifying his love for his array, he is actually unable to figure out how to compute it himself due to his extreme inability to perform even the most basic arithmetic (a plight which befalls anyone unlucky enough to participate in as many math competitions as he has). He has instead asked you to write the code for this task.

Input

The first line of input contains an integer n, $1 \leq n \leq 10^5$, which represents the number of integers in Gustavo's array.

The second line of input contains the array of integers A, given as a list of n space-separated integers $1 \leq A_i \leq 10^9$.

Output

Print one integer representing the measure of Gustavo's love for his array. As this measure may be very large, print it modulo $10^9 + 7$.

Sample Input 1

```
3
3 2 2
```

Sample Output 1

```
18
```

Sample Input 2

```
4
3 3 3 3
```

Sample Output 2

```
45
```

Sample Input 3

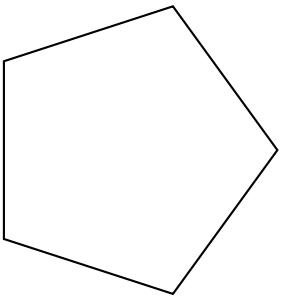
```
5
3 6 5 1 4
```

Sample Output 3

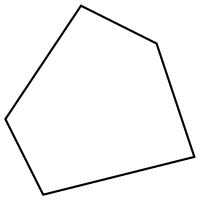
```
159
```

D – Delectable Delights

Ricky has a considerably strong sweet tooth, and as such, is an avid enjoyer of sweet treats and delights. One such delight Ricky enjoys eating is cake (given in the input as vertices of a polygon). Despite the immense joy that eating anything with sugar brings him, Ricky is also very picky, and cakes are no exception. Ricky will only eat cakes that are in the shape of a *regular polygon* when seen from above. Since mere mortals like you and Ricky cannot take perfect measurements, two values are said to be the same if they differ by no more than 10^{-4}, as differences smaller than this are not noticeable to the human eye.



A cake Ricky would eat



A cake Ricky would not eat

Input

The first line of input contains the integer n, $3 \leq n \leq 10^5$, representing the number of vertices in the cake being described.

The next n lines of input each contain two space-separated real numbers, x and y, $-10^9 \leq x, y \leq 10^9$, representing the x and y coordinates of the cake corners in Cartesian space.

Note: The list of vertices A is given in order, meaning that there is an edge between points A_i and A_{i+1} for all $i < n$. There is also an edge between A_n and A_1.

Output

The first and only line of output should contain "`Yay!`" if the polygon described in the input is a *regular* polygon, else it should contain "`:(`" (each without quotes).

Sample Input 1

```
5
-3 1
0 0
1.8781075 2.5441526
0.0388418 5.1165253
-2.9759945 4.1621865
```

Sample Output 1

```
Yay!
```

Sample Input 2

```
5
0 0
-5 5
-3 9
0 8
5 1
```

Sample Output 2

```
:(
```

E – Pleasant Permutations

Anne was raised by a very caring family, and we know that nothing says "caring family" like growing up playing with all sorts of permutations! A permutation of length n is a sequence of n numbers containing all of the integers from 1 to n *exactly once* in some order. As Anne grew up and got wiser, she began to look at her permutations in a different way: she began to wonder if some subarrays (i.e., any consecutive subsequence) of her permutation *were themselves* permutations. Unfortunately, even though she has had time to come up with this very fascinating question, she is not quite ready to figure out the answer to this on her own. She would like you to devise an approach to answer whether a permutation of k appears as a subarray of her permutation, for any value of k such that $1 \leq k \leq n$.

Input

The first line of input contains a single integer n, $1 \leq n \leq 4 \cdot 10^5$, representing the number of integers in Anne's permutation.

The next line of input contains a list, P, of n space-separated integers representing Anne's permutation. The i-th number in the permutation is the i-th integer given on this line.

Output

The first and only line of output should contain n space-separated integers, each either `0` or `1`: the k-th of these integers should be `0` if a permutation of k does *not* exist as a subarray of Anne's permutation, and `1` otherwise.

Sample Input 1

```
4
1 3 4 2
```

Sample Output 1

```
1 0 0 1
```

Sample Input 2

```
6
1 3 2 5 4 6
```

Sample Output 2

```
1 0 1 0 1 1
```

Sample Input 3

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

Sample Output 3

```
1 0 0 0 0 0 0 1
```

F – Download Delirium

Not only is Ricky picky with his sweets, he's also *very* meticulous about what type of software makes its way onto his computer. Ricky claims that over 90% of downloadable software available on the internet today is bloatware! You have suggested to Ricky to download an application called XoX that can help with organizing the High School Programming Competition, but Ricky needs rigorous convincing that XoX isn't bloatware. There are many well-known diagnostics that could be run on XoX. However, these diagnostics have dependencies (for example, diagnostic A may need the results of diagnostic B to run). Your job is to determine the order in which to run these diagnostics so that none of these dependencies are violated. If there exists two diagnostics such that A must be executed before B, and B must be executed prior to A, then the dependencies are self-contradicting and no such order can be found, in which case, Ricky will conclude that the App XoX is certainly bloatware.

Input

The first line of input contains two integers, n and m, where $2 \leq n \leq 10^5$ is the number of diagnostics to be run on the App, and $n-1 \leq m \leq \min(\frac{n(n-1)}{2}, 2 \cdot 10^5)$ is the number of pairs of diagnostics, (x, y), such that $1 \leq x, y \leq n$ with known dependencies such that x cannot run unless it has the results of y. The next m lines of input contain these pairs of dependencies as described above. All pairs listed will be distinct.

Output

If it is possible to run the diagnostics in some order that is not self-contradicting, Ricky would like you to output the diagnostics in *any* consistent order, with each diagnostic separated by a single space. Otherwise, he would like you to tell him "`Hmm... definitely bloatware.`" (without the quotes).

Sample Input 1

```
3 3
1 2
2 3
1 3
```

Sample Output 1

```
3 2 1
```

Sample Input 2

```
3 3
1 2
2 3
3 1
```

Sample Output 2

```
Hmm... definitely bloatware.
```

Sample Input 3

```
10 9
8 3
8 2
3 7
3 5
5 9
9 10
2 1
10 4
3 6
```

Sample Output 3

```
1 2 7 4 10 9 5 6 3 8
```

G – Letter Factory (Harder)

Note: This problem is a slightly different (and more difficult) variation of the problem "Letter Factory".

Sachin is back at the letter factory! As before, the foreman at the letter factory will give Sachin a string s and will give him a series of instructions. As before, some of the instructions will involve pairs of integers a and b, and Sachin must transform the specified consecutive regions (from positions a through b) of the string by shifting them all up one letter if d is 1, or down one letter if d is 0, wrapping around to the other end of the alphabet wherever necessary.

However, in this version, the foreman may also ask a new type of query as well, where Sachin is given a position a in the string, and is required to report the letter of the string at that specific position. Thus, the foreman will not ask for the entire contents of the string in bulk after all of his queries are processed! Instead, some of the instructions by the foreman will require Sachin to compute specific letters in the string at that time. As before, Sachin would like you to assist him with responding to the queries.

Input

The first line contains a single integer, q, such that $1 \leq q \leq 10^5$ it represents the number of queries the foreman makes for the given string.

The second line of input contains a single string, s, $1 \leq |s| \leq 10^5$, representing the string given to Sachin containing *only* lowercase English letters.

The next q lines of input will each contain a single query. All integers on these lines will be separated by a single space.

The first number on each of these lines will be an integer, T, which is either 0 or 1. If T is 0, then three integers $0 \leq a \leq b < |s|$ and $0 \leq d \leq 1$ will follow, denoting that the subarray of s from a to b (inclusive) is to be shifted up (if d is 1) or down (if d is 0), as described above.

If T is 1, then it will be followed by integer, k, such that $0 \leq k < |s|$ denoting that the k-th letter of s is to be reported. Note that position numbers in the string start from 0, not 1.

Output

For each query of type 1, print a line containing a single lowercase letter, reporting the contents of the string s at the specified position.

Sample Input 1

```
4
abc
0 0 1 0
1 0
0 1 2 1
1 2
```

Sample Output 1

```
z
d
```

Sample Input 2

```
4
dztz
0 0 0 0
0 1 1 1
1 1
1 2
```

Sample Output 2

```
a
t
```

H – Panther Machining

Thanks to the advent of large language models for commercial use, AI has become *very* popular recently – so popular, in fact, that many companies are rushing to create their own models so that they can compete in the highly volatile environment. However, Gian believes that this is all a bubble and has his sights set on another frontier with potential for immense growth in the future: Analog Computing! Analog computing is a form of computing that processes data using continuous physical quantities, unlike digital computers. In order to make the next big breakthrough in computing, Gian has set out to build his very own analog computer using gears, as he believes that he can simulate complex procedures using *only* gears (he has not yet considered whether this is possible but still wants to give it a try anyway). Gian talks with you, his companion, to give you a deeper understanding of how he has architected his computer.

Gian's analog computer will use n gears, of which there will be m pairs of gears whose external teeth will be meshed together. In order to start his computer, he pulls a lever which causes gear 0 to turn counter-clockwise. Although he knows these pairs of gears, he is unfortunately not sure whether the computer will run smoothly with no issues, or whether the computer will get jammed trying to run. Therefore, Gian has asked you, a skilled programmer and trusted confidant (or so he hopes; he can't have anyone else finding out about his project, lest he miss out on the crazy profits he's projected to receive once he finishes his computer!) to determine if the computer will successfully run or not, given the number of gears in his computer and the pairings of gears whose teeth are meshed.

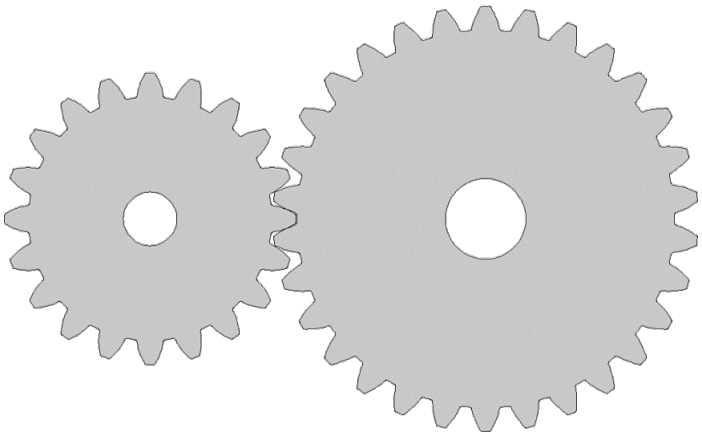


Figure for Problem H: Example of two gears whose teeth are meshed with each other.

Input

The first line of input will be two integers, n and m, separated by a space. Here $1 \leq n \leq 10^5$ is the number of gears in Gian's analog computer, and $n-1 \leq m \leq \min(\frac{n(n-1)}{2}, 2 \cdot 10^5)$ is the number of pairings of gears

whose teeth are meshed with each other. The next m lines of input will contain pairs of integers, a and b, separated by a single space. Here $1 \leq a, b \leq n$ and states that gears a and b have their teeth enmeshed. Each pair is unique.

Output

If Gian's computer is able to run without a hitch, print "`Big money to come!`". Otherwise, print "`Nothing to see here...`" (without quotation marks).

Sample Input 1

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

Sample Output 1

```
Big money to come!
```

Sample Input 2

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

Sample Output 2

```
Nothing to see here...
```

Sample Input 3

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

Sample Output 3

```
Big money to come!
```

I – Checkered Frenzy

It is no secret that the organizers of the FIU HSPC *love* a good game of chess; they'd be willing to stop anything they're doing just to play a game! This is due, in large part, to the beautiful checkerboard pattern of alternating white and black squares on the board. They routinely get mesmerized by it as they try to devise their next moves against their opponents! Because they love this problem so much, they've asked you to generate checkerboards whose individual squares have dimensions 2 characters by 2 characters using `#` for black squares, and `.` for white squares of arbitrary sizes. Can you rise to the challenge?

Input

You are given a single integer, n, where $1 \leq n \leq 100$ is the number of squares along the edge of the board they is needed.

Output

Output a board of $2n$ lines, each containing $2n$ characters: the checkerboard as was specified in the input. Please note that the organizers only play chess with checkerboards whose top-left cells are **black**!

Sample Input 1

```
1
```

Sample Output 1

```
##
##
```

Sample Input 2

```
2
```

Sample Output 2

```
##..
##..
..##
..##
```

Sample Input 3

```
4
```

Sample Output 3

```
##..##..
##..##..
..##..##
..##..##
##..##..
##..##..
..##..##
..##..##
```

J – Scary Exam

Eric is a junior at FIU, and he's taking one of the scariest classes that exist in Computer Science: Computer Algorithms! Eric is of course *very* afraid of taking this exam, as there are many difficult questions on the test, which consequently make this test very scary. Eric has come to you, his trusted Teaching Assistant, pleading with you to change the contents of the test so that the test is no longer scary for Eric. Each problem has an assigned difficulty level, and the request to modify the test will depend on a certain problem difficulty specified by Eric. The demands he makes come in one of two distinct forms:

1. Given a difficulty d, Eric requests you to remove the *easiest* problem strictly harder than d (i.e., the easiest problem whose difficulty x is **strictly** greater than d).

2. Given a difficulty d, Eric requests you to remove the *hardest* problem not harder than d (i.e., the hardest problem whose difficulty x is **less than or equal to** d).

After any one of these queries, he would like you to report the difficulty of the problem that was removed as a result of the query. In either query, if no such problem exists, you need to report `-1`.

Input

The first line of input contains two integers, n and q, separated by a single space. Here $1 \leq n \leq 3 \cdot 10^5$ is the number of questions on the exam, and $1 \leq q \leq 10^5$ is the number of requests Eric makes.
The next line of input contains a sequence D of n space-separated integers representing the difficulties of the n problems on the exam. Note that $1 \leq D_i \leq 10^9$ where D_i denotes the difficulty of the i-th problem.
The next q lines each contain two space-separated integers, T and x, where $1 \leq T \leq 2$ denotes the type of request Eric makes, and $1 \leq x \leq 10^9$ represents the difficulty specified in this query.

Output

For each query, print one line of output containing a single integer: the difficulty of the problem removed. Print -1 if no problem is removed.

Sample Input 1

```
3 4
10 10 11
1 10
1 10
1 9
1 5
```

Sample Output 1

```
11
-1
10
10
```

Sample Input 2

```
3 4
10 10 11
2 10
2 10
2 10
2 15
```

Sample Output 2

```
10
10
-1
11
```

K – Dirty Dishes

Since coming to FIU, Jason has run into some trouble with the Financial Aid office, and must pay for this semester's classes out of pocket! Luckily, some of his contacts at FIU were able to secure him a job at the 8th street kitchen in the Graham Center as a dish washer! Jason is delighted to hear that he can finally try to cover his costs, but he quickly comes to realize that washing dishes is a cumbersome task, and that washing some dishes is easier than washing other dishes. He ranks each dish, giving them each a satisfaction value s. The plates he must wash are inside of a sink, stacked vertically. These are also the *only* plates in the sink. Since Jason wants to start washing dishes from the top, he would like the plate on the top to have as high a satisfaction value as he can manage using *exactly* k moves. This is to be done before he starts washing. Jason loves playing games and has devised the following rules for each move he can make:

- If there are plates in the sink, he can remove and set aside the top-most plate. Setting aside a plate that is not at the top of the stack is not allowed.
- If there are any plates set aside, Jason can take any one of them and add them back to the top of the pile in the sink. Note that the plates set aside are not stacked on top of each other.

While Jason is good at games, he has other things to think about, like how he is going to perform on his first test in Data Structures which is coming up very soon! Please help Jason determine the highest satisfaction value of a plate that he can achieve to be placed on the top of the pile in the sink after his k moves.

Input

The first line of input contains two space-separated integers, n and k, where $1 \leq n \leq 10^5$ is the number of plates in the sink, and $0 \leq k \leq 10^9$ is the number of moves Jason has to make.
The next line of input contains a sequence S of n space-separated integers, giving the satisfaction values of the plates. Here $0 \leq S_i \leq 10^9$ is the satisfaction value of the i-th plate in the sink. The first integer on this line denotes the satisfaction value of the plate at the top of the pile.

Output

The first and only line of output should contain a single integer, denoting the highest satisfaction level that can be achieved. In the event that it is not possible to have even one plate in the sink after k moves, print `-1`.

Sample Input 1

```
6 5
5 2 2 4 0 6
```

Sample Output 1

```
6
```

Sample Input 2

```
1 1
2
```

Sample Output 2

```
-1
```

L – Circular Chaos

Suveer is a recent graduate from FIU having majored in Electrical Engineering – an exciting degree, to be sure! His job search following graduation was successful, and he is now a computer architect at **H**igh **S**ecurity **P**erformance **C**omputing (HSPC). Suveer has been placed on one of HSPC's more critical projects: a unique type of parallel computing device! Suveer is dying to talk to you about it. He has a circular disk with n evenly-spaced ports along the edge of the disk (consecutively numbered ports are adjacent to each other, with the i-th and $i+1$-th port being adjacent to each other for $1 \leq i \leq n-1$, in addition to the n-th and first ports being adjacent as well). These two ports come in exactly two distinct varieties: power sources and computing units. Each power source can power at most one computing unit. Suveer would like to power the computing units by connecting them to power sources with wire (which he has an unlimited amount of), but there is a catch: the computing units do not perform optimally when wires cross, and so his product managers have told him to power these computing units so that wires *do not* cross. Suveer wants to maximize the number of powered computing units given the constraints imposed by his product managers. Suveer is overwhelmed at work and has provided you with a generous incentive to work on solving this problem on his behalf.

Input

The first line of input will contain a single integer, n, where $1 \leq n \leq 400$ is the number of ports along the edge of Suveer's disk. The next n lines will each contain a single integer, p, where $0 \leq p \leq 1$ denotes the types of the n ports. The i-th of these lines contains the type of the i-th port. Power sources are denoted by a port of type `0`, while computing units are denoted by a port of type `1`.

Output

Output one line containing a single integer denoting the maximum number of computing units that can be powered without crossing the wires connecting the power sources and computing units.

Sample Input 1

```
13
1 1 1 1 0 1 0 0 0 1 0 0 0
```

Sample Output 1

```
6
```

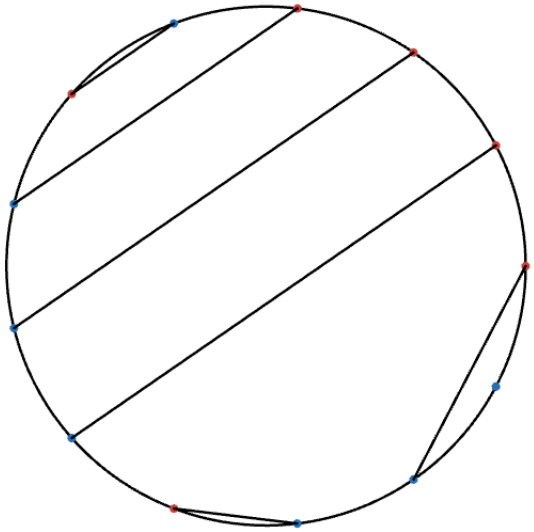


Figure shows how to connect 6 power sources and 6 computing units given in Sample Input 1 where the first computing unit is located at the 3 o'clock position and then going counterclockwise.

Sample Input 2

```
8
1 0 1 1 1 0 0 1
```

Sample Output 2

```
3
```

Sample Input 3

```
4
0 0 1 1
```

Sample Output 3

```
2
```

M – Boisterous Benchmarks

Samarth has recently gotten *deeply* into doing *reaction time tests* and has been working constantly to improve his score on these tests. Despite being addicted to them, his scores tend to be dismal. In a single game, a light turns on at various times and he needs to click a *response button* as close as possible to the time when the light turns on. He is scored using the following system. The system pairs the times at which light is turned on with the button click times. For each pairing, the *Scoring Table* below is used to compute the points earned, which is then summed up. His final score is determined as the maximum score possible among all possible pairings that satisfy the following constraint. The constraint for pairings is as follows. If A_i, B_i and A_j, B_j are two pairings such that $A_i < A_j$, then $B_i < B_j$ must also hold true. A and B here denote the sequence of times when the light turns on and when the button is clicked, respectively.

The *Scoring Table* is as follows:

Time Differences	Points Earned
0 – 12 ms	8
13 – 25 ms	4
26 – 50 ms	2
51 – 100 ms	1
101 ms or more	0

Input

The first line of input will contain two space-separated integers, n and m, where $1 \leq n, m \leq 5 \cdot 10^3$ are the lengths of the sequences A and B, respectively.

The next n lines of input will each contain a single integer from the sequence A, denoting the times $1 \leq A_i \leq 10^9$ (in milliseconds) when the light turns on. These n numbers will be given in **increasing** order and will be **unique**.

The next m lines of input will each contain a single integer from the sequence B, denoting the times $1 \leq B_i \leq 10^9$ (in milliseconds) when the button is clicked. These will also be given in **increasing** order and will be **unique**.

Output

The first and only line of output should contain a single integer, denoting the maximum score Samarth could possibly get, assuming the pairings are made to be as generous as possible for him.

Sample Input 1

```
3 4
100
200
300
99
201
240
323
```

Sample Output 1

```
20
```

Sample Input 2

```
4 3
1000
2000
2500
3000
1103
2598
4000
```

Sample Output 2

```
1
```

Sample Input 3

```
2 2
100
144
56
100
```

Sample Output 3

```
8
```

N – Grading Guidelines

Another beautiful Miami day, another exam to grade! Dr. Giri has been patiently and meticulously grading exams for his *Algorithmic Techniques* class, and it has become apparent to him that a large number of students are performing poorly. Dr. Giri wants his students to get at least 75% on this exam so that they have a reasonable chance of passing the class this semester. Since it has just come to Dr. Giri's attention that he has an unlimited amount of time to help students in need (a dream come true, truly!), he wishes to help those whose scores are less than 75%. Given the scores of each student on this exam, how many students must Dr. Giri speak with?

Input

The first line of input contains a single integer, n, where $1 \leq n \leq 100$ is the number of students in Dr. Giri's class.

The next line of input contains a sequence, S, of n space-separated integers, where $0 \leq S_i \leq 100$ and denotes the score of the i-th student on this exam.

Output

The first and only line of output should contain a single integer: the number of students Dr. Giri will speak with.

Sample Input 1

```
5
74 75 87 67 92
```

Sample Output 1

```
2
```

Sample Input 2

```
3
75 75 75
```

Sample Output 2

```
0
```

Sample Input 3

```
10
94 33 47 75 79 100 100 89 77 50
```

Sample Output 3

```
3
```