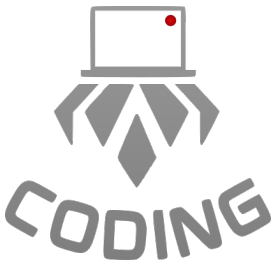
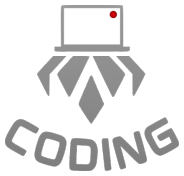


AOI 2022 Day 2

Anderson Secondary School - Robotics Club Coding Group

26 August 2022



Instructions

Some of these questions have very large input and output sizes. Some require both the use of C++ and its fast I/O. Here is a template for fast I/O that you may use.

```cpp
#include <bits/stdc++.h>
using namespace std;

int main(){
    ios_base::sync_with_stdio(0); // Warning: this will turn off all stdio functions
    // so printf and scanf and cannot be used anymore
    cin.tie(0); // Warning: this will mess with the timing of your output,
    // so do not worry if it does not output anything before you are done with
    // your input

    // All your code MUST go below these 2 lines or fast I/O will not be activated
    // Use cin and cout from now onward for input and output
}
```

Scoring

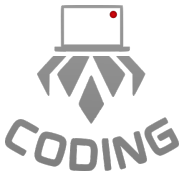
Scoring will be based on (in order of priority from top to bottom):

1. Total marks
2. Difficulty of question and subtasks solved
3. Number of questions fully solved

Time taken to solve will not be taken into account.

Rules

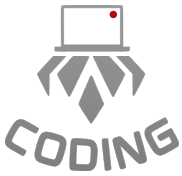
1. Strictly no communication is allowed aside from clarifications through Google Classroom.
2. You may NOT search online for any information.
3. You may use any of your pre-prepared Google Documents as you wish if approved by us. (Approach us beforehand)
4. You may also wish to use the AOI and C++ references we provided in Google Classroom.



Recommended advice to follow:

1. Try ALL questions, even if you think they are way out of your league.
2. Never give up. There's always a subtask or two or ten to grab.
3. You are allowed (and recommended) to submit the same code for different subtasks.
4. Read the questions carefully.

Happy programming! :)



Task B: GAY's Project

Doo the Dog is a computing student at Griffles-chan's Academy for Younglings (GAY). For his WAT (Weighted Assessment Three), he will need to design a basic coding language. However, he procrastinated and only started his project the night before the due date! Oh No! In addition, the teachers insist that at the very least, he needs to have a unique system to compare which strings is lexicomagically smaller compared to another.

However, as he is extremely washed due to working on that project at 2am, he wants you to help him write a program that compares two strings according to his system. The rules of the comparing system his 2am brain has thought of are: - Vowels (a, e, i, o, u) are lexicomagically smaller than consonants - All vowels are of the same lexicomagical order as each other - All consonants are of the same lexicomagical order as each other

In this system, to determine which is lexicomagically smaller, look at the first index such that one string has a vowel in that index but the other string has a consonant, and then the one with the vowel is the lexicomagically smaller string

Help Doo the Dog write the code to compare the lexicomagical order of two given strings. It is guaranteed that one of the strings is lexicomagically smaller than the other

Input Format

Your program must read from standard input.

The first line contains 2 strings containing only lowercase English letters, A and B.

Output Format

Your program must print to standard output.

The output should contain 1 string, the string that is the lexicomagically smallest.

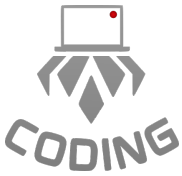
Subtasks

The maximum execution time on each instance is 2.0s, and the maximum memory usage on each instance 512MB. For all test cases, the input will satisfy the following bounds:

- $1 \leq$ Length of A and B $\leq 10^3$
- Length of A = Length of B

Your program will be tested on input instances that satisfy the following restrictions:

Subtask	Marks	Additional Constraints
1	10	A only contains vowels and B only contains consonants
2	20	The first letter of one of the strings is a consonant, the first letter of the other string is a vowel
3	70	No Additional Restrictions



Sample Testcase 1

Input	Output
aoi noi	aoi

Sample Testcase 1 Explanation

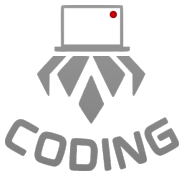
The first index which a vowel appears in aoi is 0. The first index which a vowel appears in noi is 1. Therefore, aoi is lexicomagically smaller than noi and is printed.

Sample Testcase 2

Input	Output
ryangohca errorgorn	errorgorn

Sample Testcase 2 Explanation

The first index which a vowel appears in ryangohca is 2. The first index which a vowel appears in errorgorn is 0. Therefore, errorgorn is lexicomagically smaller than ryangohca and is printed.



Task D: Embarrassment

Joe the Jaguar has noticed that the students in the coding group have been starting to develop different skill levels. There are N students which are lined up in a single line for every cca session. Each of them have a different skill level. Namely, the i-th student from the left has a skill level of S_i.

The students are insecure about their skills and will be embarrassed if the students next to them have a higher skill level. Namely, the i-th student will have embarrassment level of $\max(S_{i-1} - S_i, S_{i+1} - S_i, 0)$. $S_{i-1} - S_i$ is the difference in skill level of student i and the person to his left. $S_{i+1} - S_i$ is the difference in skill level of student i and the person to his right. The 0 is there to make sure that no one has a negative embarrassment if he has the higher skill level.

Note: The 0-th student does not have anyone to his left, so the embarrassment will only consider his right and ignore the left. The $N-1$-th student does not have anyone to his right, so the embarrassment will only consider his left and ignore the right.

Joe the Jaguar wants to motivates his students and would like to minimise the sum of all embarrassments across all students. Help him find the minimum sum of embarrassments.

Input Format

Your program must read from standard input.

The first line contains 1 integer, N, the number of students in the coding group.
The second line contains N integers, the array S, S_0, S_1, S_2, ... , S_{N-2}, S_{N-1}.

Output Format

Your program must print to standard output.

The first line of your output should contain 1 integer, the minimum sum of all embarrassments.

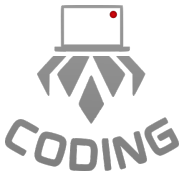
Implementation Note

You may need to use c++ and fast io with it. Refer to first page for fast io template.

Subtasks

The maximum execution time on each instance is 2.0s, and the maximum memory usage on each instance is 512MB. For all test cases, the input will satisfy the following bounds:

- $2 \leq N \leq 2 \cdot 10^6$
- $0 \leq S_i \leq 10^9$



Your program will be tested on input instances that satisfy the following restrictions:

Subtask	**Marks**	**Additional Constraints**
1	4	S_i are all equal
2	10	$2 \leq N \leq 10$
3	11	$2 \leq N \leq 10^3$
4	75	No Additional Restrictions

Sample Testcase 1

Input	Output
4 19 3 42 0	42

Sample Testcase 1 Explanation

42 19 3 0 is the optimal arrangement and the sum of embarrassments is max(19-42,0) + max(42-19, 3-19, 0) + max(19-3, 0-3, 0) + max(3-0, 0) = 0 + 23 + 16 + 3 = 42.

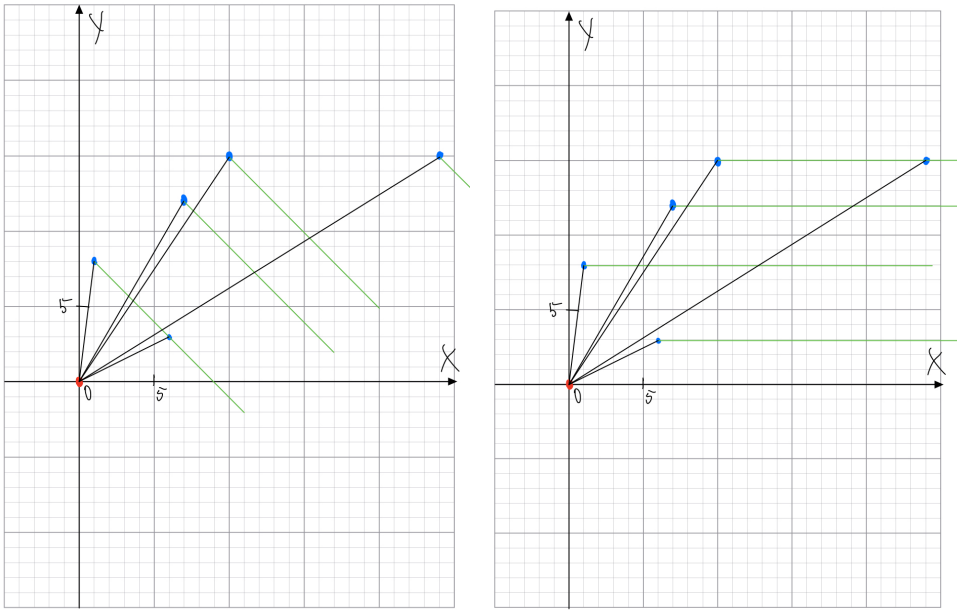
Task F: Physics go brrr

Shor the duck has made a new game in unity, Angry Robots! This game consists of the character, Mr Nathan, throwing robots at his worst enemies, irresponsible excos. Shor the duck would like to publish this game out for everyone to play with. However, he has encountered a bug with the game's physics.

There are T rounds in Angry Robots and at every round, Mr Nathan will throw 1 robot at 1 target, the irresponsible exco. Mr Nathan at will be located at the origin (0, 0), the target will be located at (A_i, B_i) on a 2D plane for the i-th round.

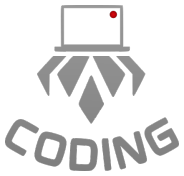
Note: A_i, B_i will always be positive as Shor the duck has negativenumbersphobia.

However, Mr Nathan can only throw the robots to S squares with coordinates at (X_i, Y_i) for all $1 \leq i \leq S$ as the others contain black holes that will consume the robot. After it has travelled to a square, it will travel in another straight line towards the RIGHT ONLY with a gradient of M_i, for the i-th round, forever. Note that the coordinates which the robot can travel through does not necessarily have to be an integer. For example, the robot can travel through (0.5, 2.35). A gradient of m means that the y value will increase by m when x value increases by 1. For example, m=-4 means that when x increases by 1, y increases by -4.



The red dot is where Mr Nathan is at, (0, 0) and the blue dots are the squares that Mr Nathan can only throw at. The black lines represent the initial path and green lines represent second path. Mr Nathan can only hit the target if the target is in a safe square or if its in the way of a GREEN path, the BLACK path does not account for hitting the target. The first picture shows a gradient of -1, second picture shows a gradient of 0

Since Shor the duck is lazy, he has decided to make the buggy physics a feature rather than a bug as long as it is possible for Mr Nathan to win all rounds. Help him determine whether Mr Nathan can hit the target for T rounds.



Input Format

Your program must read from standard input.

The first line contains an integer, T.
For T instances, the first line contains 6 integers, S, M, A, B, the number of squares Mr Nathan can throw at, the gradient after it reaches a square, X and Y coordinates of the irresponsible exco.
For the next S lines, there will be 2 integers, X_i and Y_i, the x coordinate and y coordinate of the i-th square Mr Nathan can throw at.

Output Format

Your program must print to standard output.

The output should contain T lines. The i-th line being "YES" or "NO", without the double quotes and case sensitive. YES if Mr Nathan can hit the target on the i-th round and NO if Mr Nathan cannot.

Implementation Note

You might need to use c++ and fast io with it. Refer to first page for fast io template.

Subtasks

The maximum execution time on each instance is 2.0s, and the maximum memory usage on each instance is 512MB. For all test cases, the input will satisfy the following bounds:

- $1 \leq T \leq 10^3$
- $1 \leq S \leq 10^3$
- $-10^2 \leq M \leq 10^2$
- $1 \leq A, B, X_i, Y_i \leq 10^7$

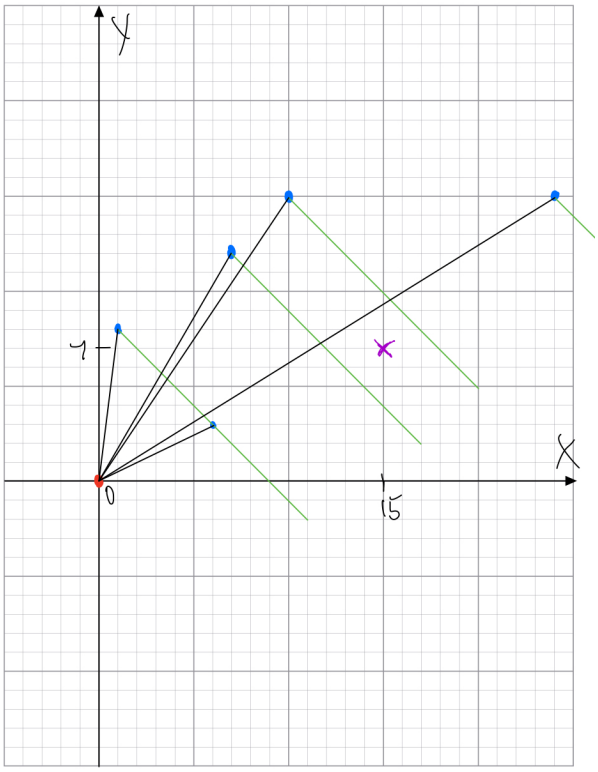
Your program will be tested on input instances that satisfy the following restrictions:

Subtask	Marks	Additional Constraints
1	10	$X_i = 10^7$
2	15	$M = 0$
3	25	$1 \leq A, B \leq 10$, $1 \leq S \leq 10^2$
4	50	No Additional Restrictions

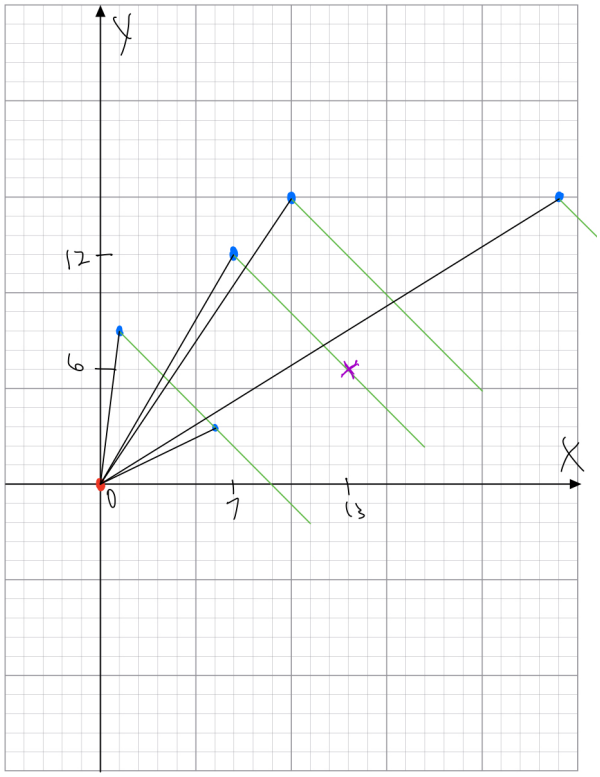
Sample Testcase 1

Input	Output
2 5 -1 15 7 1 8 6 3 7 12 24 15 10 15 5 -1 13 6 1 8 6 3 7 12 24 15 10 15	NO YES

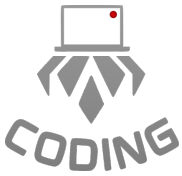
Sample testcase 1 Explanation



The purple cross represents target. For round 1, the target does not lie on any of the green paths or at any blue dots, so it is not possible for Mr Nathan to hit it.



The purple cross represents target. For round 2, the target lies on the green path of the blue dot at (7, 12), so it is not possible for Mr Nathan to hit it.



Task H: What The Quack

Shor the duck just finished his math exam but he did not have enough time to finish the last question, "What The QUACK!", he shouted. The last question consisted of a complex and unpredictable function, $f(x)$, that was strictly increasing as x increases (they said its derivative was always positive or something, but let's be honest no one cares about this). He was asked to solve for the smallest value of x such that $f(x) \geq K$. However, he was drunk on segment trees and could not think properly to solve it.

Now, Shor the duck is mad and wants redemption on the question. He has made a vow that he will answer any queries asking for the smallest value of x such that $f(x) \geq K$ so that no one will ever have to suffer like he did.

The function $f(x)$ is defined as such, where x must be an integer:

Let value of $f(x)$ be val, initially equal to 0

```
int val = 0;
while(x > 0){
    val += (x*x);
    x /= 2; // Rounded down
}
```

At the end, val is the value of $f(x)$.

For example $f(3) = 10$ because
val $= 3^2 + 1^2$

Shor the duck's friends have prepared Q queries with Q different integers, the i-th one being K_i. Despite saying that he would answer those queries, he did not expect his friends to come up with so many! Looking to avoid this responsibility, he tasked you to help him prepare a program that can answer all their queries at blazing fast speeds.

He has already declined all queries that result in $f(x)$ such that x > M, but he still has a lot of queries to answer!

Input Format

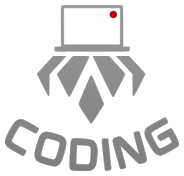
Your program must read from standard input.

The first line contains 1 integer, Q. The subsequent Q lines contains 1 integer, K_i.

Output Format

Your program must print to standard output.

The output should contain Q lines, the i-th line containing the answer for query i. The answer being the smallest number, x, such that $f(x) \geq K_i$.



Subtasks

The maximum execution time on each instance is 2.0s, and the maximum memory usage on each instance is 512MB. For all test cases, the input will satisfy the following bounds:

- $1 \leq Q \leq 10^5$
- $0 \leq K_i \leq 10^{16}$
- $M = 10^8$

Your program will be tested on input instances that satisfy the following restrictions:

Subtask	**Marks**	**Additional Constraints**
1	7	$K_i = 0$
2	26	$0 \leq K_i \leq 10^2, 1 \leq Q \leq 10^2$
3	33	$M = 2 \cdot 10^5$
4	34	No Additional Constraints

Sample Testcase 1

Input	Output
3 10 15 462	3 4 19

Sample Testcase 1 Explanation

$f(3) = 10$ as we have established earlier and this is exactly 10, so 3 is the smallest value of x. $f(3) = 10$ and $f(4) = 4^2 + 2^2 + 1^2 = 21$, so $f(4)$ is the smallest one that is more than or equal to 15, so 4 is the smallest value of x. $f(18) = 18^2 + 9^2 + 4^2 + 2^2 + 1^2 = 426$ and $f(19) = 19^2 + 9^2 + 4^2 + 2^2 + 1^2 = 463$, so 19 is the smallest possible answer as $f(19)$ is the first number to be more than or equal to 462.

Special DP Task: Royal Guards

Shor the Duck is the emperor of Duck Land. You are not to question his rule. However, he seems to have amassed a few political opponents who are not entirely keen on letting Shor live. (little do they know, he's immortal)

Still, Shor would very much prefer to stay alive. He's picked out N candidates for his royal guard, and arranged them in an order such that the closer together they are in the list, the more similar their skillsets are. We want to have a large variety of different skillsets, while also minimising the amount of overlap of similar skillsets. Thus, we will prefer to pick candidates that are as far apart on the list as possible.

However, computing the total effectiveness is not easy. Firstly, you must sum up the effectiveness of all the guards you have selected. Then, we need to look at the differences in index of each pair of adjacent selected guards such that no other selected guard is in between them. Square the individual differences in indices, divide each square by K (rounded down), and sum them up. Add this to the sum of effectiveness to get the total effectiveness. Please see the examples below.

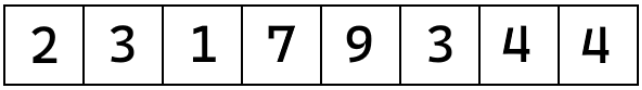


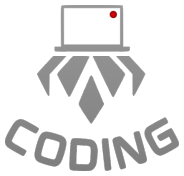
Fig 1.1: A possible arrangement of candidate guards

Fig 1.2: A POSSIBLE (not optimal) selection of guards, with red coloured squares indicating chosen guards.

In Fig 1.2, the index pairs of adjacent selected guards are (0, 1), (1, 4), and (4, 6) [This is not the optimal configuration!]. This is because these are the pairs of indices of red squares that are the closest to each other. (0-indexed)

The differences in indices of the pairs are thus 1, 3 and 2 respectively. We need to square these values to get 1, 9 and 4. Now, we divide EACH of these values by K, which will be 2 for just this testcase. We get 0, 4 and 2 as a result (rounded down). The sum of the results is then 6.

Now we add in the total effectiveness of all the guards selected, 2, 3, 9 and 4 respectively to get 18. We add this to our differences in indices, and we get our final total effectiveness of this configuration of 24.



Input Format

Your program must read from standard input.

The first line contains two integers, N and K, the number of guards and the number you should divide each of the sum of squares of index differences by respectively.
The second line will contain N integers, denoting the effectiveness E_i of each guard.

Output Format

Your program must print to standard output.

Output 1 integer, the highest achievable total effectiveness.

Implementation Note

You might need to use C++ and the fast I/O template if you are daring enough to sacrifice all your points to the final impossible subtask. Otherwise, it is likely not necessary (but go ahead and do it anyways)

Subtasks

The maximum execution time on each instance is 2.0s, and the maximum memory usage on each instance is 512MB. For all test cases, the input will satisfy the following bounds:

- $1 \leq N \leq 10^6$ (Note: this is for impossible subtask)
- $0 \leq abs(E_i) \leq 10^3$
- $1 \leq K \leq 10$

Your program will be tested on input instances that satisfy the following restrictions:

Subtask	Marks	Additional Constraints
1	11	$E_i = -(10^3), N \leq 10$
2	71	$N \leq 10$
3	92	$N \leq 10^2$
4	126	$N \leq 2 * 10^3$ (last serious subtask)
5	5000	No additional constraints (impossible, if it's not in the contest means we had no time to set, but feel free to try and solve it after if you are looking for pain)

Note

You are allowed to take no guards.

Sample Testcase 1

Input	Output
8 2 2 3 1 7 9 3 4 4	34

Sample Testcase 1 Explanation

The figure above illustrates the optimal selection for this testcase, omitting only the guard with index 2. Taking the differences of indices of adjacent selected squares (see previous illustration if confused), we have the values 1,1,2,1,1,1. We square all of them to get the values 1,1,4,1,1,1. We divide all of them by K while rounding down, resulting in 0,0,2,0,0,0. We add them all together to get 2 as our bonus for diverse skillsets.

We then add that to our sum of effectiveness of chosen guards, which is 2+3+7+9+3+4+4 = 32, to get our final answer of 34.

Sample Testcase 2

Input	Output
9 3 -2 3 -1 -7 9 11 1 5 -9	32

Sample Testcase 2 Explanation

The best configuration we can get is to pick the guards with indices 1, 4, 5 and 7.