Beta Release (beta)

Giorgio is developing a new online FPS game! As he should now have got rid of most bugs, he decided to start a beta program. The game is now released free to play, with a disclaimer kindly asking[1] each user to play only a *single* game (except of course for Giorgio's user, which can play as many games as he likes). Since the game is still in the development phase, it is currently run on a very small server, which is having trouble handling the traffic resulting from the beta release. Giorgio is thus starting to wonder whether some users are playing multiple games, ignoring his polite request!

Figure 1: Sample of the cutting-edge graphics of the new game by Giorgio.

You are given the server logs, which consist of the N user identifiers V_i ($0 \leq i < N$) of the games played so far. Among them are included the games played by Giorgio himself; however, the identifiers are anonymised so that you are unable to distinguish his one from the others. Help Giorgio check whether there are users playing multiple games!

> ☞ Among the attachments of this task you may find a template file `beta.*` with a sample incomplete implementation.

Input

The first line contains the only integer N. The second line contains N integers V_i.

Output

You need to write a single line with the string `'YES'` if there are certainly users playing the game multiple times, or `'NO'` if it may be the case that the only user playing multiple games is Giorgio.

Constraints

- $1 \leq N \leq 100\,000$.
- $1 \leq V_i \leq 10^9$ for each $i = 0 \ldots N-1$.

[1] Kindness always wins (or so they say).

Scoring

Your program will be tested against several test cases grouped in subtasks. In order to obtain the score of a subtask, your program needs to correctly solve all of its test cases.

– **Subtask 1** (0 points) Examples.

– **Subtask 2** (30 points) $N \leq 1000$, $V_i \leq 10^6$.

– **Subtask 3** (15 points) $N \leq 1000$.

– **Subtask 4** (35 points) $V_i \leq 10^6$.

– **Subtask 5** (20 points) No additional limitations.

Examples

input	output
5 17 42 42 31 58	NO
7 98 13 28 98 28 37 45	YES

Explanation

In the **first sample case**, Giorgio could have identifier 42, and thus it is possible that no player is ignoring the kind request.

In the **second sample case**, even if Giorgio has identifier 98 or 28, there is definitely another player ignoring the kind request and playing a second game.

Christmas Balls (`christmasballs`)

Christmas is coming, and Alessandro wants to buy a Christmas tree to decorate his living room. When he heard that in Pordenone a new tree shop just opened, he immediately decided to go there to poke around. This shop has a huge tree decorated with N balls, one at each branching point (the root, node 0, lies at the base of the tree). The balls are quite spectacular: Alessandro noticed that they come in C different colors in total.

The peculiarity of this shop is that you can buy the whole tree, or cut a branch and buy just a *subtree*! You can make the cut just below a ball, so that you take that ball and the subtree rooted (growing from) there.

But Alessandro doesn't want just a random tree, he wants the *nicest* one! His taste is in fact quite peculiar: he appreciates when the number of balls of each color is well distributed. More precisely, if we denote with x the number of balls painted with (one of) the most frequent color in a subtree, he wants to maximize the number of colors that appear in x balls each.

Help Alessandro find where to cut the tree by printing the maximum amount of most-frequent colors he can get.

> ☞ Among the attachments of this task you may find a template file `christmasballs.*` with a sample incomplete implementation.

Input

The first line contains the only integer N, the number of nodes of the tree. The second line contains N integers C_i, the colors of each ball. The third line contains $N-1$ integers: P_i (with $1 \leq i < N$) is the index of the parent of the i-th node. The parent of the root node (P_0) is not defined and therefore **not present in the input**.

Output

You need to write a single line with an integer: the number of colors with the highest frequency in the optimal subtree.

Constraints

- $1 \leq N \leq 100\,000$.
- $0 \leq C_i < N$, for all $i = 0 \ldots N-1$.
- $0 \leq P_i < i$, for all $i = 1 \ldots N-1$.

Scoring

Your program will be tested against several test cases grouped in subtasks. In order to obtain the score of a subtask, your program needs to correctly solve all of its test cases.

– **Subtask 1** (0 points) Examples.

– **Subtask 2** (7 points) $N \leq 1000$ and the tree is just a line: $P_i = i - 1$ for all $i = 1 \ldots N-1$.

– **Subtask 3** (9 points) The tree is just a line: $P_i = i - 1$ for all $i = 1 \ldots N-1$.

– **Subtask 4** (15 points) $N \leq 1000$ and $C \leq 2$.

– **Subtask 5** (19 points) $N \leq 1000$.

– **Subtask 6** (17 points) $C \leq 2$.

– **Subtask 7** (33 points) No additional limitations.

Examples

input	output
8 1 2 0 2 0 0 1 1 0 0 1 1 3 4 4	3
5 0 1 1 0 0 0 1 2 3	2

Explanation

In the figures below, the small number at the top left is the number representing the color. We arbitrarily associate color 0 with green, color 1 with blue, and color 2 with orange.

In the **first sample case** there are 8 nodes in 3 different colors. The solution is to cut the tree and take the subtree rooted at node 1. This way we are left with 6 nodes: the most frequent color has 2 balls, and there are 3 colors with that many balls.

If, for example, we keep the entire tree, the most frequent color appears with 3 balls, but there are only 2 colors with 3 balls, and it would be worse than the aforementioned solution.

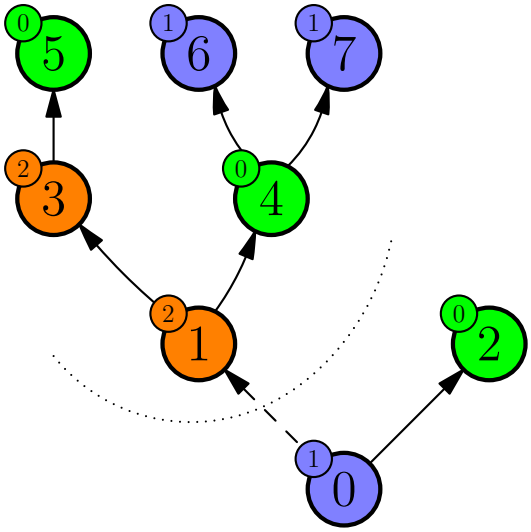

Figure 1: First sample case.

In the **second sample case** the tree forms a line with 5 nodes. The optimal solution is to cut the subtree rooted at node 1. Then, the most frequent color appears with 2 balls, and there are 2 colors with that many balls.

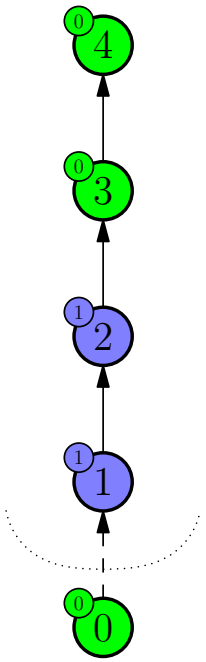

Figure 2: Second sample case.

Disk Failure 2 (disks2)

Edoardo is busy setting up the servers that will host the second OIS round. However, he notes that the servers have something odd. Edoardo immediately recognizes the reason: the disks are broken again!

Figure 1: It's not the first time it happens.

The server will host N contests in the next N days, each contest start from hour A_i to hour B_i of day i. Edoardo knows he will need T hours to repair the broken disks. During the repair the server has to be turned off, but Edoardo doesn't want to turn off the server during a contest, so he has to do it at night.

Since this is quite urgent, help Edoardo by finding the first contest after which he can turn off the server and repair the disks.

☞ Among the attachments of this task you may find a template file `disks2.*` with a sample incomplete implementation.

Input

The first line contains two integers N and T.

The next N lines each contain two integers A_i and B_i.

Output

You need to write a single line with an integer: the first night Edoardo can repair the disks or -1 if he can't repair the disks **before the last contest**.

Constraints

- $1 \leq N \leq 100\,000$.
- $1 \leq T \leq 24$.
- $0 \leq A_i < B_i \leq 24$ for each $0 \leq i \leq N-1$.
- $A_0 = 0$.
- Hour 24 of day i is the same as hour 0 of day $i+1$.

Scoring

Your program will be tested against several test cases grouped in subtasks. In order to obtain the score of a subtask, your program needs to correctly solve all of its test cases.

– **Subtask 1** (0 points) Examples.

– **Subtask 2** (40 points) $A_i = 0$ for each $0 \leq i \leq N - 1$.

– **Subtask 3** (30 points) $T = 1$.

– **Subtask 4** (30 points) No additional limitations.

Examples

input	output
5 10 0 20 5 23 7 19 10 22 9 20	2
3 20 0 23 7 23 6 22	-1

Explanation

In the **first sample case**, Edoardo needs 10 hours to replace the disks.

- The first night the server is unused from 20:00 to 5:00 for 9 hours which is not enough.
- The second night the server is unused from 23:00 to 7:00 for 8 hours which is not enough.
- The third night the server is unused from 19:00 to 10:00 for 15 hours which is enough to replace the disks.

In the **second sample case**, Edoardo needs 20 hours to replace the disks.

- The first night the server is unused from 23:00 to 7:00 for 8 hours which is not enough.
- The second night the server is unused from 23:00 to 6:00 for 7 hours which is not enough.

Edoardo can't repair the disks before the last contest, so the answer is -1.

Kalindrome Strings (kalindrome)

William loves strings and their properties, and he is especially fond of palindromes. He likes palindromes so much, that he started reading lots of research papers on them and finally stumbled upon a new kind of property that generalizes the palindrome concept: the *kalindrome*.

We say that a string is a *kalindrome* if, for some positive integer K, it's possible to split the string in many substrings, each of them K characters long, so that they can be "read" from left to right in the same way as right to left. More precisely: if we concatenate the substrings starting from the last one, we will recreate the original string.

For example: the word "banaba" is a kalindrome, because if we choose $K = 2$ then we obtain the substrings: **ba**, **na**, **ba**, which can indeed be concatenated starting from the last one, to reconstruct the original word.

You might have noticed that, by this definition, any string is a kalindrome! That's true, in fact we could choose K to be equal to the string's length, and we would then obtain a single substring which would of course respect the definition! However, William is interested to know **what's the smallest K for a given string that would prove it is a kalindrome**.

Help him by writing a program that automates this test!

> ☞ Among the attachments of this task you may find a template file `kalindrome.*` with a sample incomplete implementation.

Input

The first line contains the only integer N, the length of the string to be tested. The second line contains a string S.

Output

You need to write a single line with an integer: the smallest positive integer K that proves the kalindromeness of the string S.

Constraints

- $1 \leq N \leq 1\,000\,000$.
- The string S is N-characters long and only contains lowercase English letters.

Scoring

Your program will be tested against several test cases grouped in subtasks. In order to obtain the score of a subtask, your program needs to correctly solve all of its test cases.

– **Subtask 1** (0 points) Examples.

– **Subtask 2** (40 points) The value of K is always either 1 or N.

– **Subtask 3** (40 points) $N \leq 1000$.

– **Subtask 4** (20 points) No additional limitations.

Examples

input	output
6 banaba	2
12 eszyciyciesz	3

Explanation

In the **first sample case**, we can split the string in three 2-characters long substrings: **ba**, **na**, **ba**.

In the **second sample case**, we can split the string in four 3-characters long substrings: **esz**, **yci**, **yci**, **esz**.

Pub Encounter (pickup)

Yesterday, Edoardo spent an amazing night at a bar. After a handful of drinks his charm was unleashed, and girls all around him were fighting for him (to the best of his memories of the night). One girl even wrote her number on his arm, for him to call her back!

Figure 1: Edoardo getting the phone number written on his arm.

Unfortunately, after waking up the next day, Edoardo realised to his dismay that the number was all smeared around and intelligible. He only remembers few things he noticed about the number:

- that it did not contain any zero as a digit,
- that the number was a multiple of A,
- that the sum of its digits was B,

Now there is only one sensible option left: enumerate all such numbers and try to call all of them in order. Help Edoardo track the girl, by computing the K-th number satisfying the constraints above!

☞ Among the attachments of this task you may find a template file `pickup.*` with a sample incomplete implementation.

Input

The first line contains the three integer A, B and K.

Output

You need to write a single line with an integer: the K-th multiple of A without zero digits and with sum of digits B.

Constraints

- $1 \leq A, B \leq 300$.
- $1 \leq K \leq 10^{18}$.
- There are at least K numbers as described above.

Scoring

Your program will be tested against several test cases grouped in subtasks. In order to obtain the score of a subtask, your program needs to correctly solve all of its test cases.

– **Subtask 1** (0 points) Examples.

– **Subtask 2** (20 points) $A = 1, K = 1$.

– **Subtask 3** (20 points) The answer is less than 10^6.

– **Subtask 4** (20 points) $A = 1$.

– **Subtask 5** (20 points) $B \leq 20$.

– **Subtask 6** (20 points) No additional limitations.

Examples

input	output
3 9 11	126
17 6 3	3111

Explanation

In the **first sample case**, the first numbers multiple of 3 with sum of digits 9 are:

9 18 27 36 45 54 63 72 81 117 126

Thus, the 11th number is 126.

Note that 90 and 108 are not valid as they contain the digit '0'.

In the **second sample case**, the only numbers multiple of 17 with sum of digits 6 are 51, 1122, 3111.

Police Investigation 5 (police5)

The police is still searching fearsome William! Given the skills of the criminal, who is apparently able to drive his car and find his way through the M roads of the city, the police has placed bombs in some of those roads to stop and eventually catch him. A button pressed at time T will trigger the simultaneous explosion of all the bombs, effectively making some roads unusable.

Figure 1: William driving: apparently is very calm in a deserted city.

Labelling the N intersections with integers, William is now at the intersection 0 and wants to reach his nest at the intersection $N-1$. Thanks to his network of fellow criminals, he has come into possession of the plan of the police, knowing for each road from intersection A_i to B_i whether it will explode or not.

William knows how many seconds it will cost him to drive through each road, but obviously cannot drive on roads that are already exploded or will explode while driving on them. Is he going to be able to reach his nest and get away with it? If so, how many seconds is it going to take?

> ☞ Among the attachments of this task you may find a template file `police5.*` with a sample incomplete implementation.

Input

The first line contains three integers: N, M and T. Each of the following M lines describes a road with four integers: the start A_i, the destination B_i, the cost C_i in seconds to drive through it, and whether it will explode ($E_i = 1$) or not ($E_i = 0$).

Output

You need to write a single line with an integer: the time in seconds required for William to reach his nest, or -1 if he cannot make it.

Constraints

- $2 \leq N \leq 10\,000$.
- $1 \leq M \leq 100\,000$.
- $1 \leq T \leq 10\,000\,000$.
- $0 \leq A_i, B_i < N$ for each $i = 0 \ldots N-1$.
- $1 \leq C_i \leq 100$ for each $i = 0 \ldots N-1$.
- $E_i = 0$ or $E_i = 1$ for each $i = 0 \ldots N-1$.
- Each road has a directionality and William can only drive from A_i to B_i.
- It is possible to drive on a road that will explode exactly at the moment in which William will reach its end.

Scoring

Your program will be tested against several test cases grouped in subtasks. In order to obtain the score of a subtask, your program needs to correctly solve all of its test cases.

– **Subtask 1** (0 points) Examples.

– **Subtask 2** (10 points) There are just $N-1$ roads that form a "path" from 0 to $N-1$, i.e. from 0 to 1, from 1 to 2, ..., and from $N-2$ to $N-1$.

– **Subtask 3** (20 points) $C_i = 1$ for each $i = 0 \ldots N-1$.

– **Subtask 4** (20 points) $T = 10\,000\,000$.

– **Subtask 5** (30 points) $N \leq 100$.

– **Subtask 6** (20 points) No additional limitations.

Examples

input	output
4 4 100 0 1 10 1 1 3 50 1 0 2 20 1 2 3 60 1	60
4 4 20 0 1 10 0 1 3 50 1 0 2 20 1 2 3 60 0	80

Explanation

In the **first sample case** William can reach his destination driving through intersection 0, 1 and 3 taking $10+50 = 60$ seconds. All the roads contain bombs and will explode at $T = 100$, but by that time William will be already safe.

In the **second sample case** William cannot drive again from 1 to 2 because at time $T = 20$ that road will explode, and he will be in the middle of that (10 seconds after having left intersection 1). He can use the alternative path 0, 2, and 3. The road from 0 to 2 will explode too, but exactly at the moment at which William will not be driving on it anymore.

Investing in Stocks (stocks)

An ever-increasing number of young people decides to try to make money online using live streaming platforms. Luca didn't want to miss out and a while ago started streaming gameplays, investing the earnings in his favourite company's stocks.

Luca's friends make fun of him using this meme.
Rightfully so: be careful before investing in stocks!

So far N purchases have been made, some with money coming directly from the streaming service (indicated with $S_i = 1$), others with money coming from different sources (thus, not from the streaming: $S_i = 0$). Luca kept a detailed log of the value of the stocks bought in the i-th purchase, amounting to V_i euros.

He is now reviewing the performance of his stock portfolio and notices that some periods have been particularly profitable. We consider an uninterrupted sequence of stock purchases **profitable** when the purchases have been only made with money coming from the streaming service *and* the amount bought each time never decreased.

He is now wondering: how profitable was the most profitable sequence? How much money has Luca invested during the **most profitable** sequence?

> ☞ Among the attachments of this task you may find a template file `stocks.*` with a sample incomplete implementation.

Input

The first line contains the only integer N. The second line contains N integers V_i. The third line contains N integers S_i.

Output

You need to write a single line with an integer: the total value of the stocks Luca bought in the most profitable sequence.

Constraints

- $1 \leq N \leq 1\,000\,000$.
- $1 \leq V_i \leq 1000$ for each $i = 0 \ldots N-1$.
- $S_i = 0$ or $S_i = 1$ for each $i = 0 \ldots N-1$.
- A sequence of just one purchase made with money from the streaming can already be considered a *profitable* one. It is always guaranteed that at least one profitable sequence exists.

Scoring

Your program will be tested against several test cases grouped in subtasks. In order to obtain the score of a subtask, your program needs to correctly solve all of its test cases.

– **Subtask 1** (0 points) Examples.

– **Subtask 2** (35 points) $S_i = 1$ for each $i = 0 \ldots N-1$.

– **Subtask 3** (25 points) $N \leq 1000$.

– **Subtask 4** (25 points) $N \leq 10\,000$.

– **Subtask 5** (15 points) No additional limitations.

Examples

input	output
4 100 120 130 80 1 1 1 1	350
4 100 120 130 80 1 1 0 1	220

Explanation

In the **first sample case** all the purchases have been made with money coming from the streaming service. The most profitable sequence consists of the first, the second and the third purchase, with an overall value of $100 + 120 + 130 = 350$ euros.

In the **second sample case** all the purchases *but the third one* have been made with money coming from the streaming service. The most profitable sequence consists just of the first and the second purchase, with an overall value of $100 + 120 = 220$ euros. Despite the non-decreasing price of the third purchase, that one was made with money coming from other sources.

Plowing Fields (tractor)

Marco is tired of preparing the OIS rounds, so he decided to quit his job and become a farmer.

Figure 1: Marco's new tractor.

As a farmer, he now has to plow his field. The field is divided into N rows, and to plow the field Marco has to plow each row one at a time. Every row must be plowed **exactly once**.

Marco starts from row 0 on one side of the field, he plows the row to the other side, then turns the tractor and repeats in another row. If the tractor is in row x, when it turns it can go to row y with cost $\text{abs}(x - y)$. However, the tractor is very large hence needs space to turn around. Then, it can only go from row x to row y iff $\text{abs}(x - y) \geq K$.

Help Marco find a way to plow the field **minimizing** the total cost.

> ☞ Among the attachments of this task you may find a template file `tractor.*` with a sample incomplete implementation.

Input

The first line contains the integers N and K.

Output

You need to write two lines. The first line containts a single integer: the cost of plowing the field. The second line consists N integers: the order in which the rows are plowed.

Constraints

- $1 \leq N \leq 100\,000$.
- $1 \leq K \leq 8$.
- Marco starts from row 0, however, he can end on any row.
- It is guaranteed that exists at least one way to plow the field.

Scoring

Your program will be tested against several test cases grouped in subtasks. Your score on a subtask will be equal to the minimum score for one of its testcases multiplied by the value of the subtask. Your score on a test case will be:

- 0 if your output doesn't represent a valid solution;
- 1 if $T_{\text{out}} \leq T_{\text{ref}}$;
- $0.9 \cdot \left(\frac{T_{\text{ref}}}{T_{\text{out}}}\right)^{1.5}$ otherwise.

Where T_{out} is the cost of plowing the field of your solution and T_{ref} is the cost of a reference solution (not necessarily optimal).

– **Subtask 1** (0 points) Examples.

– **Subtask 2** (20 points) $N \leq 10$, $K \leq 5$.

– **Subtask 3** (20 points) $N \leq 30$, $K \leq 5$.

– **Subtask 4** (20 points) $N \leq 100$, $K \leq 5$.

– **Subtask 5** (20 points) $K \leq 3$.

– **Subtask 6** (20 points) No additional limitations.

Examples

input	output
8 3	23 0 3 6 2 5 1 4 7
10 3	33 0 3 6 9 5 8 2 7 4 1

Explanation

In the **first sample case**, Marco can plow the field in the following way:

- start from row 0;
- go to row 3 with cost $|3 - 0| = 3$;
- go to row 6 with cost $|6 - 3| = 3$;
- go to row 2 with cost $|2 - 6| = 4$;
- go to row 5 with cost $|5 - 2| = 3$;
- go to row 1 with cost $|1 - 5| = 4$;
- go to row 4 with cost $|4 - 1| = 3$;
- go to row 7 with cost $|7 - 4| = 3$.

The total cost is $3+3+4+3+4+3+3=23$.

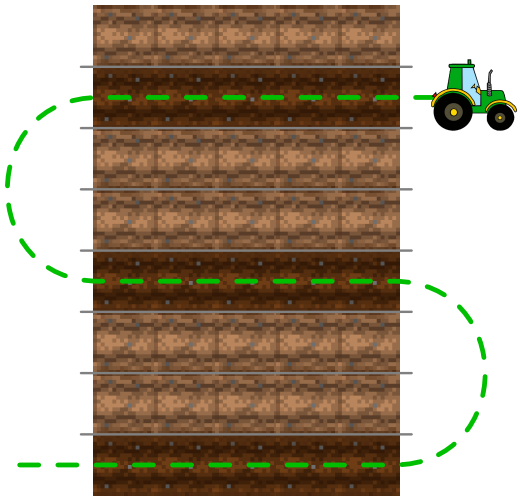

Figure 2: Step 1

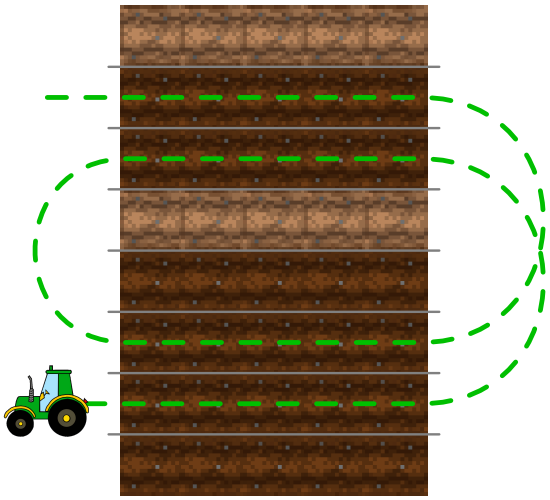

Figure 3: Step 2

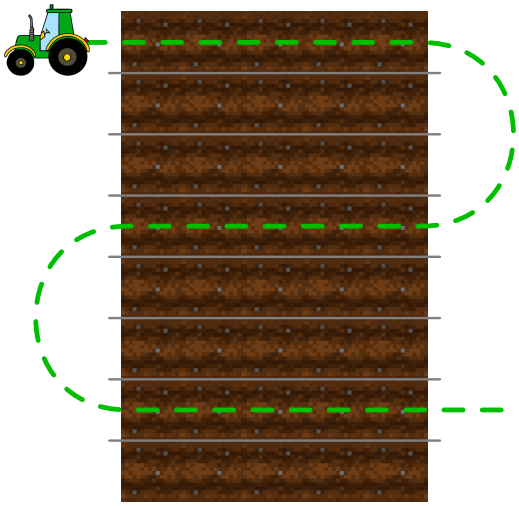

Figure 4: Step 3

In the **second sample case**, Marco can plow the field in the following way:

- start from row 0;
- go to row 3 with cost $|3-0|=3$;
- go to row 6 with cost $|6-3|=3$;
- go to row 9 with cost $|9-6|=3$;
- go to row 5 with cost $|5-9|=4$;
- go to row 8 with cost $|8-5|=3$;
- go to row 2 with cost $|2-8|=6$;
- go to row 7 with cost $|7-2|=5$.
- go to row 4 with cost $|4-7|=3$.
- go to row 1 with cost $|1-4|=3$.

The total cost is $3+3+3+4+3+6+5+3+3=33$.