Problem A. Katya and the Broken Keyboard

Time limit:	1 second
Memory limit:	512 megabytes

Katya was assigned to write an essay, but unfortunately, she realized that some of the keys on her laptop were not always responding.

Through trial and error, she found out that n keys were broken. Specifically, for each broken key c_i, she determined a number x_i. When pressed for the first time, the key c_i does not work, then it prints the letter for $x_i - 1$ times, then again does not work once, works again for $x_i - 1$ times, and so on.

Help Katya find out how many times she will have to press the keys to guarantee that she can write her essay.

Input

The first line is given a string s—the essay Katya wanted to type. The essay consists only of lowercase English letters, and the length of the string does not exceed 100 000.

The second line is given n ($0 \le n \le 26$)—the number of broken keys.

The following n lines contain c_i and x_i, where c_i is the broken key, and x_i ($2 \le x_i \le 100$) is the frequency of the key's failure to respond. It is guaranteed that c_i is a lowercase English letter.

Output

Output the minimum number of keystrokes that guarantees Katya will be able to write the essay.

Example

standard input
russiaopenhighschoolteamprogrammingcontest 2 s 3 o 5

standard output
46

Problem B. Cooperative Game on a Tree

Time limit:	1 second
Memory limit:	512 megabytes

Sergey and Azat enjoy playing board games. After many hours spent in fierce battles against each other, the guys got tired and decided to try playing *cooperative* games—in these games, players must cooperate with each other to achieve the best possible result. Luckily for the boys, Azat had just recently been gifted one such game.

In this game, there is a rooted tree with n vertices with the root at vertex 1, as well as a set of chips — one blue and many red. Initially, one blue and one red chip are placed in vertex 1. Next, the following steps are taken:

1. The first player moves the blue chip from its current vertex to one of its children.
2. If the blue chip ends up in a leaf (i.e., a vertex that has no children), the game ends.
3. The second player does the same with the red chip: he moves it from its current vertex to one of its children.
4. If the red chip ends up in a leaf, it stays there permanently and can no longer be moved, but a new red chip is placed in the vertex where the blue chip currently is. After that, the game continues from step 1.

The goal of the game is to end up with as many red chips on the tree as possible. However, the game developers did not specify what the best possible result in the game could be, so the guys are asking you to calculate this information.

Input

The first line contains the number n ($2 \le n \le 2 \cdot 10^5$) — the number of vertices in the tree. The next line specifies $n-1$ numbers $p_2, p_3, \ldots p_n$, where the number p_i means that vertex i is a child of vertex p_i ($1 \le p_i < i$).

Output

Print a single number — the maximum number of red chips that can end up on the tree at the end of the game.

Examples

standard input	standard output
4 1 1 3	2
3 1 2	1

Note

In the first example, the optimal sequence of actions is as follows: the first player moves his chip to vertex 3, and the second player moves his one to vertex 2. After that, a new red chip appears in vertex 3, the first player moves the blue chip to vertex 4 and the game ends there.

Problem C. Driving License Exam

Time limit:	1 second
Memory limit:	512 megabytes

Misha is a beginner driver. After successfully passing the theoretical part of the driving exam, he had little left to do — pass the actual driving exam. However, it turned out to be not so simple, as Misha does not know how to drive on ice.

The track where the exam takes place is located in a field and consists of n intersections, with a road of length d_i between the i-th and $(i+1)$-th intersections for i from 1 to $n-1$. Initially, there are w_i units of ice at the i-th intersection.

For the exam, the inspectors choose a continuous subsegment $[l, r]$ of the track: all intersections with numbers from l to r inclusive and all the roads between them. The track for the exam must be circular, so after choosing l and r, the inspectors also connect the intersections with numbers l and r with a temporary road of length x.

The inspectors do not want Misha to pass the exam, so all the roads of the exam track must be covered with ice. To do this, it is necessary to move the ice from the intersections that the road connects; one unit of ice can cover one unit of the road. Hence, the total amount of ice moved onto the road must be not less than its length. Of course, no more ice can be moved onto two adjacent roads from each intersection than is present there.

Unfortunately, the current amount of ice may not be enough to cover all the roads, so the inspectors are interested in the minimum total amount of ice that needs to be added to the intersections to make it possible to cover all the roads and prevent Misha from passing the exam.

You will need to process three types of queries: change the amount of ice at a certain intersection, change the length of a certain road, and solve the problem for a subsegment of the track.

Input

The first line contains two integers n and q — the number of intersections and queries, respectively ($2 \le n \le 2 \cdot 10^5, 1 \le q \le 2 \cdot 10^5$).

The second line contains n integers w_i — the initial amount of ice at each intersection ($1 \le w_i \le 10^9$).

The third line contains $n-1$ integers d_i — the initial length of each road between adjacent intersections ($1 \le d_i \le 10^9$).

The next q lines describe the queries:

- 1 p x ($1 \le p \le n, 1 \le x \le 10^9$) — perform the assignment $w_p := x$;
- 2 p x ($1 \le p \le n-1, 1 \le x \le 10^9$) — perform the assignment $d_p := x$;
- 3 l r x ($1 \le l < r \le n$, $1 \le x \le 10^9$) — determine the minimum amount of ice that needs to be added to the intersections for the track formed by the subsegment of intersections from l to r inclusive, adding temporary road of length x closing the cycle. Note that the third query does not change the amount of ice at the intersections; you only need to answer how much ice needs to be added to the intersections to cover all the roads with ice. After each query, the inspectors dismantle the temporary road, so it does not need to be considered in other queries.

It is guaranteed that there is at least one query of the third type.

Output

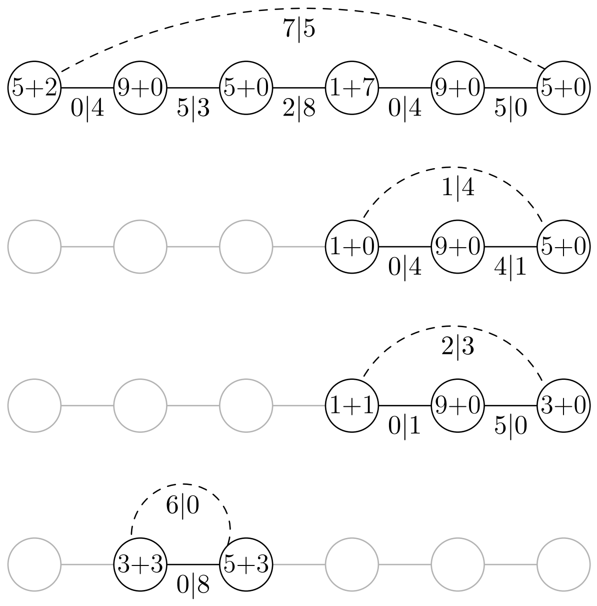
For each query of the third type, output the minimum total amount of ice that needs to be added to the intersections to cover the entire circular track with ice.

Example

standard input	standard output
6 7	9
5 9 5 1 9 5	0
4 8 10 4 5	1
3 1 6 12	6
3 4 6 5	
2 4 1	
1 6 3	
3 4 6 5	
1 2 3	
3 2 3 6	

Note

The figure shows illustrations for the example. Dotted lines indicate temporary roads closing the cycle. The current and added amounts of ice are indicated at the intersections, and the amount of ice moved from the intersections is indicated on the roads.



Problem D. Count the Christmas Trees

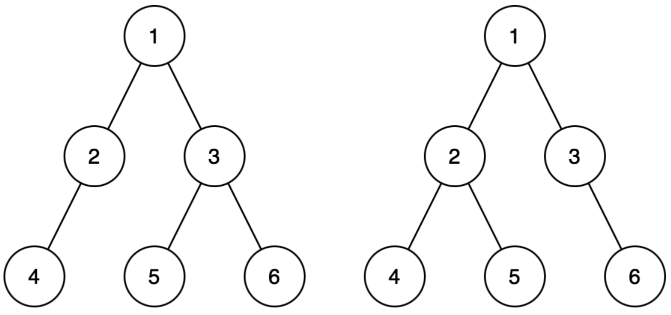
Time limit:	1 second
Memory limit:	512 megabytes

The New Year is coming soon! To celebrate, Matvey decided to buy himself a Christmas tree. He went to a store where graph-trees are sold. However, when he arrived, he couldn't decide because there was a huge selection. Then he decided to count how many Christmas trees of height n exist.

Matvey considers a tree with a root at vertex 1 to be a *Christmas tree of height* n if it satisfies the following conditions:

- let's assume that the root is on the first layer, its children are on the second layer, and so on, the children of the vertex on the i-th layer are on the $(i + 1)$-th layer. Then there must be exactly i vertices on the i-th layer;
- each vertex has no more than two children;
- number the vertices by layers from top to bottom with integers, starting with 1. On one layer, number the vertices from left to right. After that, consider two vertices u, v, such that they are on the same layer and $u < v$, then all children of vertex u, if any, must have a smaller number than the children of vertex v.

Matvey considers Christmas trees to be different if there are numbers i, j, such that one tree has an edge between vertices i and j, and the other tree does not have such an edge.



For height 3, there are two different Christmas trees, which are shown in the figure.

You need to calculate the number of different Christmas trees of height n.

Input

The first line contains one number n — the depth of the Christmas tree ($1 \le n \le 5000$).

Output

In a single line, output one number — the number of Christmas trees of depth n.

Since the answer can be very large, output it modulo $10^9 + 7$.

Examples

standard input	standard output
3	2
4	12

Problem E. Not Everything Is So Ambiguous

Time limit:	2 seconds
Memory limit:	512 megabytes

This is an interactive problem.

The jury's program has chosen an integer x between 1 and 10^9 and a base b for the numeral system ($2 \le b \le 2023$), after which it told you the number of digits in the representation of the number x in the notation with base b.

You can make queries "how many digits are there in the number $x + d$", where d is an integer from 1 to 10^{18}. Your task is to guess both x and b in no more than 100 queries.

Interaction Protocol

The interaction begins with the jury's program outputting n — the number of digits in the base-b representation of the number x.

After that, your program makes queries in the form of "`?` d", where $1 \le d \le 10^{18}$, and the jury's program outputs the answer — the number of base-b digits in the representation of $x + d$.

If you are ready to output the answer, print "`!` x b", where x and b are the guessed values of the number and the base of the numeral system. Outputting the answer does not count as a query.

Example

standard input	standard output
1	
	? 1
2	
	? 3
3	
	! 1 2

Note

In the example of interaction, queries and responses are separated by empty lines to visually demonstrate the interaction process. In the actual interaction with the jury's program, there will be no empty lines; they should also not be printed, but after each query, as well as after outputting the answer, a newline must be printed.

Problem F. Magic Square

Time limit:	1 second
Memory limit:	512 megabytes

In a distant magical kingdom, there lived a famous wizard named Merlin. He was known for his magical abilities and the skill to create magical items. One day, Merlin created a magic square of size n by n, with each cell containing an integer from 1 to n^2, and all numbers being distinct. The magic square had amazing properties: all rows and all columns had the same sum of numbers.

However, one day, when Merlin was momentarily distracted, the evil sorcerer Gargamel sneaked into his laboratory and decided to cause harm. He swapped two numbers in the magic square, hoping to spoil its magical power.

Now Merlin needs your help to determine which numbers were swapped.

Input

The first line contains a single number n ($3 \le n \le 1000$)—the size of the square. In the following n lines, there are n numbers a_{ij} ($1 \le a_{ij} \le n^2$). It is guaranteed that all numbers are distinct. It is guaranteed that there are two numbers that can be swapped to make the square magical again.

Output

You are required to output four numbers r_1, c_1, r_2, and c_2—the row number and the column number of the first number and the row number and the column number of the second number that need to be swapped to make the square magical again. Rows and columns are numbered from top to bottom and from left to right, respectively, starting with one.

Example

standard input	standard output

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

Problem G. Casino

Time limit: 1 second
Memory limit: 512 megabytes

Ocean and his friends decided to rob a casino by playing the slot machines. There are a total of n slot machines in the casino, and playing on the i-th machine will yield a win of a_i rubles.

Ocean has hacked the system and knows that he will win on each machine on the first try, but only once. Ocean is very greedy, so he will definitely play exactly once on each machine.

However, the casino's security system considers it suspicious if a player wins one or more times on the machines and the total winnings of the player are divisible by 3. The mathematician in Ocean's team wants to find the number of ways to choose the order of the machines to play on each machine exactly once and remain undetected by the security system. Since there can be very many ways, output the remainder of the number of ways divided by $10^9 + 7$.

Input

The first line contains a single integer n — the number of machines in the casino ($1 \le n \le 10^5$).

The second line contains n integers $a_1, a_2, \ldots, a_n$ ($1 \le a_i \le 10^9$).

Output

Output a single integer — the number of ways to play on each machine exactly once, without ever having a positive total win that is a multiple of 3, taken modulo $10^9 + 7$.

Examples

standard input	standard output
3 100 21 892	4
4 11 19 30 32	6
3 4 298 28	0

Problem H. Scooter Numbers

Time limit:	1 second
Memory limit:	512 megabytes

Residents of Partoburg often violate traffic rules, get into accidents involving scooters, and steal other people's scooters. In order to control scooter-related incidents, the sheriff of Partoburg has introduced a scooter registration system.

On the first day of each month the sheriff chooses a natural number n and prints all possible scooter numbers in the format $[a_1 + a_2 + \ldots + a_k]$, where $a_1 + a_2 + \ldots + a_k = n$, and natural numbers a_i are ordered non-decreasingly: $a_1 \leq a_2 \leq \ldots \leq a_k$. Each number is unique: there cannot be two scooters with the same number. The numbers are assigned to scooter owners. If an owner does not have a number in a given month, he cannot use the scooter during that month.

The sheriff himself also moves around on a scooter. To easily identify his scooter, it has a special number format. The sheriff's scooter number is a single number calculated as follows. For all scooter numbers in the current month $[a_1 + a_2 + \ldots + a_k]$, the value $\mathrm{mex} a_1, a_2, \ldots, a_k$ is calculated — the minimum natural number absent in the number. These values are summed. The number of the sheriffs scooter is a remainder of the division of the resulting sum by $10^9 + 7$.

The sheriff wants to automate the scooter number assignment system and started with his own scooter. This month he chose the number n. Help determine what number his scooter will receive.

Input

In the first line of input is a single natural number n — the number chosen by the sheriff for this month ($1 \leq n \leq 1000$).

Output

Output a single integer — the sheriff's scooter number: an integer from 0 to $10^9 + 6$, the remainder of the division by $10^9 + 7$ of the sum of all the scooter numbers' mex values for this month.

Examples

standard input	standard output
1	2
3	6

Note

In the first example $n = 1$. The only possible number is [1]. In this case the sheriff's scooter number will be $\mathrm{mex} 1 = 2$.

In the second example $n = 3$. All possible numbers are $[1 + 1 + 1]$, $[1 + 2]$, $[3]$. In this case the sheriff's scooter number will be $\mathrm{mex} 1, 1, 1 + \mathrm{mex} 1, 2 + \mathrm{mex} 3 = 2 + 3 + 1 = 6$.

Problem I. Squares

Time limit:	2 seconds
Memory limit:	512 megabytes

Consider an infinite grid. An infinite set of 2×2 squares is *covering set* if each cell of the plane is covered by exactly one square and they cover all the cells of the plane.

A set of squares is *good* if it is a subset of some *covering set*.

You have an initially empty set of squares S and n queries for adding and removing squares (x_i, y_i), where the pair of numbers (x_i, y_i) describes a square that covers the cells (x_i, y_i), $(x_i + 1, y_i)$, $(x_i, y_i + 1)$, and $(x_i + 1, y_i + 1)$.

After each query, you are required to output a single number —the size of the largest *good* subset of the set S.

Input

The first line contains a single number n ($1 \le n \le 200\,000$) — the number of queries.

The following n lines contain two integers x_i, y_i ($1 \le x_i, y_i \le 10^9$). If at the moment of the i-th query the square defined by the pair (x_i, y_i) was contained in S, then it is removed from the set, otherwise—it is added.

Output

Output n lines, in the i-th line output the size of the largest good subset of S after executing the first i queries.

Example

standard input	standard output

5
1 1
2 2
3 3
4 4
1 1
	1
	1
	2
	2
	2

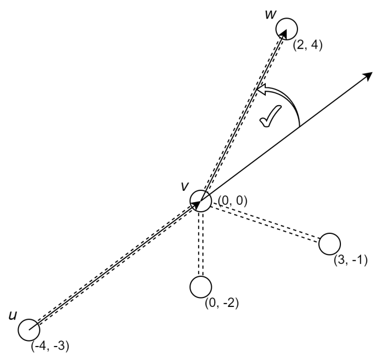
Problem J. Streets of Flatland

Time limit:	2 seconds
Memory limit:	512 megabytes

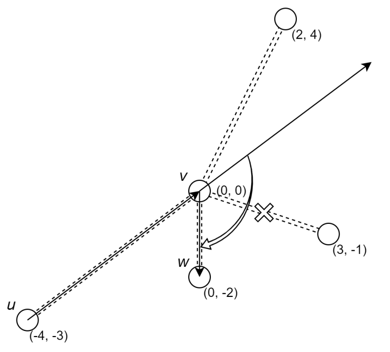
Flatland is a country located on a plane. There are a total of n cities in Flatland, connected by $n-1$ bidirectional roads in such a way that it is possible to travel from any city to any other. City number i is located at the point with coordinates (x_i, y_i).

Recently, Flatland decided to carry out a road reform and assign a name to each road. However, to save resources it was decided to use as few different names as possible. Several roads can be given the same name if all these roads form a simple path and for any two consecutive roads (u, v) and (v, w) on this path, the pair of directed roads $u \to v$ and $v \to w$ is *compatible* and the pair of directed roads $w \to v$ and $v \to u$ is *compatible.*

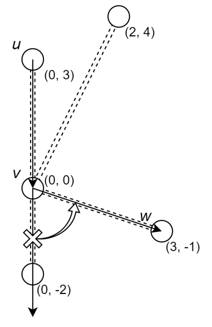
In this case a pair of roads $u \to v$ and $v \to w$ will be called compatible if no other road starting from city v is encountered when rotating the vector $\overrightarrow{uv}$ drawn from point v to the vector $\overrightarrow{vw}$ around the point v.



Compatible pair $u \to v$ and $v \to w$



Incompatible pair $u \to v$ and $v \to w$



Co-directed road

Note that the roads must be compatible in both directions, while the illustration shows compatibility only in one direction. Also, if the vector $\overrightarrow{uv}$ drawn from v turns out to be co-directed with some road, it is considered the only one compatible with the road (u, v).

A path of one or more roads with the same name is called a *highway.* Determine the minimum number of highways with different names into which all the roads of Flatland can be divided.

Input

The first line of input contains a single integer n — the number of cities in Flatland ($1 \le n \le 2 \cdot 10^5$).

In the i-th of the following n lines, there are two integers x_i and y_i — the coordinates of the i-th city ($|x_i|, |y_i| \le 10^9$). It is guaranteed that no two cities are located at the same point on the plane.

The next $n-1$ lines describe the roads in Flatland — the i-th line contains the numbers of the cities u_i and v_i connected by the i-th road ($1 \le u_i, v_i \le n$). It is guaranteed that there is exactly one path between any two cities.

It is also guaranteed that no two roads emanating from the same city are co-directed. However, it is *not guaranteed* that the segments on the plane corresponding to the roads do not intersect.

Output

Output a single integer — the minimum number of highways after naming all the roads of Flatland.

Examples

standard input	standard output

5
0 0
2 4
3 -1
0 -2
-4 -3
1 2
1 3
1 4
1 5
	2

5
0 0
2 4
3 -1
0 -2
0 3
1 2
1 3
1 4
1 5
	3

Note

The problem statement includes illustrations for two examples. In the first example, it is possible to combine the paths $(2, 1, 4)$ and $(3, 1, 5)$ into one highway. In the second example, it is not possible to get less than three highways because the roads $(1, 5)$ and $(1, 4)$ are only compatible with each other, while $(1, 2)$ and $(1, 3)$ are not compatible.

Problem K. Guess the String

Time limit:	1 second
Memory limit:	512 megabytes

This is an interactive problem.

The jury's program has a string s of length n, which consists of the letters 'a', 'b', and 'c'. Your program must guess it.

To guess the string, your program can make queries. Each query has form of a number i $(1 \le i \le n-1)$ and a string u of length two that consisting of lowercase letters of the English alphabet. In response to a query, the jury's program reports how many of the statements $s_i = u_1$ and $s_{i+1} = u_2$ are true.

You are required to guess the thought string, making no more than $\lceil \frac{4}{3}n \rceil$ queries. Here $\lceil \ldots \rceil$ denotes rounding up.

Interaction Protocol

In this problem, you will need to play with the jury's program several times. It is guaranteed that the number of games in each run does not exceed 100.

At the beginning of each game, your program should read from the standard input stream the number n $(2 \le n \le 100)$. If $n = 0$, this means that your program should terminate its execution. Otherwise, you start the game with a thought string of length n, and your program can make no more than $\lceil \frac{4}{3}n \rceil$ queries.

To make a query, you need to print to the output stream "? i u where $1 \le i \le n-1$, u is a string of two lowercase English letters. In response, you need to read from the standard input stream an integer a, equal to 0, 1, or 2 — how many of the statements from $s_i = u_1$ and $s_{i+1} = u_2$ are true.

When your program has guessed the thought string, it is necessary to print to the standard output stream "! s where s is the string of length n that the jury's program has thought of. Printing the answer does not count as a query. After your program prints the guessed string, it can immediately proceed to the next game.

The jury's program in this problem may be adaptive. In other words, the jury's program can tailor its responses in such a way that there exists a string for which all responses are correct, but it is not fixed in advance and changes depending on the queries of your program. In particular, if there is a string for which all responses to queries are correct, but different from the string that your program guessed, the solution may receive a verdict of "Wrong answer".

Example

standard input	standard output
3	
	? 1 ab
2	
	? 2 ba
1	
	? 2 bb
1	
	? 2 bc
2	
	! abc
0	

Note

In the example of interactive interaction, queries and responses are separated by empty lines to visually show the interaction process. In the actual interaction with the jury's program, there will be no empty lines, nor should they be printed, but after each query, as well as after printing the answer, it is necessary to print a newline.

Problem L. a, ab, ba Strings

Time limit:	1 second
Memory limit:	512 megabytes

Given a string S of length n, consisting of the characters '`a`' and '`b`', and q queries. There are two types of queries:

1. replace the i-th character with the opposite one (that is, '`a`' is changed to '`b`', and '`b`' is changed to '`a`').
2. check if it is possible to split the substring from the l-th to the r-th character into substrings '`a`', '`ab`', and '`ba`'.

For each query of the second type, output the response to the query.

Input

The first line contains a single number n — the length of the string ($1 \le n \le 100\,000$).

The second line contains the string S itself, consisting of the characters '`a`' and '`b`'.

The third line contains the number q — the number of queries ($1 \le q \le 100\,000$).

Each of the following q lines starts with an integer *type* — the type of query ($1 \le type \le 2$).

In the first type of queries, an integer i follows ($1 \le i \le n$).

In the second type of queries, a pair of integers l, r follows ($1 \le l \le r \le n$).

Output

For each second type of query, print '`YES`' if the string can be split as described in the condition, and '`NO`' otherwise.

Example

standard input	standard output
7 abbabba 4 2 3 4 2 3 5 1 6 2 3 7	YES NO YES

Note

In the first example, the substring from the first query can be represented as '`ba`', the substring from the second query '`bab`' cannot be split into valid substrings, after the third query the string looks like '`abbabaa`', the substring from the fourth query can be split as follows: '`ba|ba|a`'

Problem M. Three Suitcases

Time limit:	1 second
Memory limit:	512 megabytes

When Katya goes on a trip, she always takes three suitcases with her. She packs all her clothes in first, which weighs x kilograms. In the second, she packs cosmetics weighing y kilograms. And in the third suitcase, she packs shoes weighing z kilograms.

Katya checked the airline's website for baggage fees in advance. If she checks in the suitcases with a total weight of less than 5 kilograms, the cost will be a rubles. If the weight is 5 or more but less than 10 kilograms, the cost will be b rubles. And if the weight is 10 kilograms or more, the price will be c rubles.

Help Katya send all the suitcases and pay as little as possible.

Input

The first line contains the number x ($1 \leq x \leq 10$)—the weight of the first suitcase. The second line contains the number y ($1 \leq y \leq 10$)—the weight of the second suitcase. The third line contains the number z ($1 \leq z \leq 10$)—the weight of the third suitcase. The fourth line contains the number a ($1 \leq a \leq 100$)—the cost for sending less 5 kilograms. The fifth line contains the number b ($1 \leq b \leq 100$)—the cost for sending from 5 kilograms inclusive to less than 10 kilograms. The sixth line contains the number c ($1 \leq c \leq 100$)—the cost for sending 10 kilograms or more.

Output

Print the minimum cost of sending the three suitcases.

Example

standard input	standard output
2 3 5 10 15 25	25