Problem A. 86

Input file:	`standard input`
Output file:	`standard output`
Time limit:	4 seconds
Memory limit:	512 megabytes

The Republic of San Magnolia is at war against the autonomous Legion forces and is currently on the back foot. The Republic's military operates n combat units on the battlefield, and if too many of them are destroyed, the Republic will be forced to abandon the front.

Each day, the Legion launches an attack on every unit exactly once. However, not all attacks are guaranteed to destroy a unit, as each unit is equipped with defensive systems. For the i-th unit, there is a $\frac{x_i}{y_i}$ probability that a single attack will destroy it. Otherwise, the attack is repelled, and the unit remains operational for the day, taking no damage.

All attacks across different units and on different days are **independent**. Once a unit is destroyed, it remains destroyed and cannot become operational again.

You are a citizen trying to understand how long the Republic can continue the war. However, due to misinformation and limited access to the battlefield, different sources report different states of the front lines. Each report claims that a subset of units is operational, which means all others are already destroyed.

The Republic can continue fighting as long as at least k units remain operational. The front will collapse as soon as fewer than k units remain.

For each report, determine the **expected number of days** the Republic can continue fighting, on the condition that the report is accurate.

Input

The first line of the input contains a single integer t ($1 \le t \le 10$) — the number of test cases.

The first line of each test case contains two space-separated integers n ($1 \le n \le 17$) and r ($1 \le r \le 10^5$) — the number of combat units and the number of reports.

The second line of each test case contains n space-separated integers $x_1, x_2, \ldots, x_n$ ($1 \le x_i \le 10^6$) — the numerators of the probabilities of the units being destroyed in a single attack.

The third line of each test case contains n space-separated integers $y_1, y_2, \ldots, y_n$ ($x_i \le y_i \le 10^6$) — the denominators of the probabilities of the units being destroyed in a single attack.

The next $2r$ lines of each test case describe the r reports. Each report is described as follows:

The first line of each report contains two space-separated integers m and k ($1 \le k \le m \le n$) — the number of operational units, and the minimum number of operational units required to continue the war.

The second line of each report contains m distinct space-separated integers $b_1, b_2, \ldots, b_m$ ($1 \le b_i \le n$) — the indices of the units that are operational.

It is guaranteed that the sum of $(n \times r)$ over all test cases does not exceed 10^6.

Output

For each test case, output r space-separated integers in a line — the expected number of days the Republic can continue fighting for each report, modulo $(10^9 + 7)$.

Formally, if the expected number of days can be represented as an irreducible fraction $\frac{p}{q}$, output the integer $p \times q^{-1} \bmod (10^9 + 7)$, where q^{-1} is an integer such that $q \times q^{-1} \bmod (10^9 + 7) = 1$.

Example

standard input	standard output
3 3 2 1 1 2 7 5 9 1 1 1 2 1 2 3 5 1 1 1 1 1 1 1 1 1 1 1 5 2 1 2 3 4 5 10 3 1 1 1 1 1 1 1 1 1 1 2 2 2 2 2 2 2 2 2 2 1 1 10 10 1 1 2 3 4 5 6 7 8 9 10 10 10 1 2 3 4 5 6 7 8 9 10	7 617647070 1 2 254032441 94819161

Note

For the first test case, only one unit is operational according to the first report. Every day, the unit has a $\frac{1}{7}$ probability of being destroyed. The expected survival duration of that unit is 7 days. For the second report, the expected duration of the battle is $\frac{233}{34}$ days.

Problem B. All Your Base

Input file:	`standard input`
Output file:	`standard output`
Time limit:	1 second
Memory limit:	256 megabytes

This is an interactive problem. You must flush the output after printing each line. For example, in C++ use fflush(stdout), in Java — System.out.flush(), and in Python — sys.stdout.flush().

You have to find a hidden integer X ($1 \le X \le 10^{15}$) using an interactive system.

Let the representation of X in base n be: $X = d_k n^k + d_{k-1} n^{k-1} + \cdots + d_1 n^1 + d_0 n^0$, where $0 \le d_i < n$ for all i.

You may ask questions. In each question, you provide an integer base n ($2 \le n \le 100$) and a boolean value $b \in \{0, 1\}$.

The system will return:

- If $b = 0$, the sum of digits at even positions in the base n representation of X: $\sum d_{2i}$ for $i \in \{0, 1, 2, \dots\}$.
- If $b = 1$, the sum of digits at odd positions in the base n representation of X: $\sum d_{2i+1}$ for $i \in \{0, 1, 2, \dots\}$.

After at most 10 questions, you have to output the value of X.

All numbers are in **base 10** unless otherwise specified.

Input

The first line contains an integer t ($1 \le t \le 10^4$), the number of test cases.

After reading the input for each test case, proceed with the interaction as described.

Interaction Protocol

To ask a question, print a line in the format "? n b" (without quotes), where n is the base and b is the boolean value. After each query, read an integer S — the sum of digits returned by the system.

When you are ready to provide the answer, print a line in the format "! X" (without quotes), where X is the hidden integer.

Your program can ask at most 10 questions. Note that the final line "! X" is not counted as a question.

Example

standard input	standard output
3 1 2 0 9 8	 ? 2 0 ? 3 1 ! 16 ? 5 0 ! 5 ? 8 0 ? 8 1 ! 2026

Note

For the third test case, the hidden integer is 2026.

The interaction proceeds as follows:

Participant	Jury	Explanation
? 8 0	9	$2026 = 3752_8$. Even positions (0, 2) contain digits $2, 7$ — so the sum is 9.
? 8 1	8	$2026 = 3752_8$. Odd positions (1, 3) contain digits $5, 3$ — so the sum is 8.
! 2026	–	The correct value of X is reported.

Problem C. Chander Gari

Input file:	`standard input`
Output file:	`standard output`
Time limit:	1 second
Memory limit:	256 megabytes

You are driving a jeep called *Chander Gari* along the hilly roads from Bandarban to Thanchi. The route consists of n segments, where segment i connects Checkpoint $i-1$ to Checkpoint i. The jeep is equipped with a fuel tank of maximum capacity c liters and you begin your journey at Checkpoint 0 with a **full tank**.

To travel from Checkpoint $i-1$ to Checkpoint i, the jeep consumes w_i liters of fuel. If the tank contains less than w_i liters of fuel before departure, the jeep stalls and the journey ends.

Upon arriving at any checkpoint i ($1 \le i \le n$), you have the option to refuel. Due to a severe crisis, fuel is strictly limited. If you choose to stop and refuel, the vendor will fill the jeep's tank with exactly r liters of fuel. If this causes the fuel level to exceed the capacity c, the excess fuel overflows and is wasted.

Your task is to determine a refueling strategy that ensures the jeep safely reaches Checkpoint n while minimizing the total number of times you stop to refuel.

Input

The first line contains a single integer t ($1 \le t \le 1000$) — the number of test cases. The description of the test cases follows.

The first line of each test case contains three integers n, c and r ($1 \le n \le 2 \cdot 10^5, 1 \le r \le c \le 10^9$) — the number of segments, the maximum tank capacity, and the fixed amount of fuel (in liters) provided per refill.

The second line of each test case contains n space-separated integers $w_1, w_2, \ldots, w_n$ ($1 \le w_i \le c$) — the fuel required to traverse each successive segment.

It is guaranteed that the sum of n over all test cases does not exceed 10^6.

Output

For each test case, output the result on a new line.

If it is impossible to reach Checkpoint n regardless of the strategy, print `"Impossible"` (without quotes).

Otherwise, print a binary string of length n. The i-th character of the string ($1 \le i \le n$) should be `1` if you want to refuel at Checkpoint i, and `0` otherwise.

The strategy must minimize the total number of `1`s. If multiple optimal strategies exist, you may output any one of them.

Example

standard input	standard output
6 2 5 5 1 3 2 9 4 9 2 1 10 9 3 9 5 4 5 4 2 1 3 5 4 2 5 2 5 2 5 5 3 5 4 3 4 2	00 10 0 110111110 Impossible 110

Note

In the first test case, initially the tank has 5 liters of fuel. After traversing the 1^{st} segment, 4 liters remain. The 2^{nd} segment requires 3 liters. Since we have 4 liters, we can complete the segment. There is no need to refuel at this point.

In the last test case, initially the tank has 5 liters of fuel. After traversing the 1^{st} segment, 2 liters remain. Since the 2^{nd} segment requires 4 liters and the tank does not have enough fuel, we need to refuel at Checkpoint 1. After refilling, the fuel level becomes $\min(2+4, 5) = 5$ liters.

Now, the 2^{nd} segment can be traversed. It consumes 4 liters, leaving 1 liter in the tank. The 3^{rd} segment requires 2 liters, but the tank has only 1 liter. So, we must refuel again at Checkpoint 2. After this refill, the fuel level becomes $\min(1+4, 5) = 5$ liters.

Now, the 3^{rd} segment can be traversed. It consumes 2 liters, leaving 3 liters in the tank. No further refueling is needed.

Problem D. Function Ordering

Input file:	`standard input`
Output file:	`standard output`
Time limit:	1 second
Memory limit:	256 megabytes

You are given a rooted tree consisting of n vertices, numbered from 1 to n. Vertex 1 is the root.

Each vertex v initially stores a function

$$f_v(x) = a_v x + b_v,$$

where a_v and b_v are positive integers.

You have to perform the following operation exactly $n - 1$ times until only the root vertex remains:

1. Choose a vertex u ($u \neq 1$) that is currently a leaf in the tree.
2. Let p be the parent of u.
3. Update the function stored at vertex p to be the composition of the function at u and the current function at p:
$$f_p(x) \leftarrow f_u(f_p(x))$$
4. Remove vertex u and the edge (u, p) from the tree.

After $n - 1$ operations, only vertex 1 remains with a final function $F(x)$. Your task is to maximize the value of $F(0)$.

Input

The first line contains an integer t ($1 \leq t \leq 100$) — the number of test cases.

For each test case:

- The first line contains an integer n ($1 \leq n \leq 2 \cdot 10^5$) — the number of vertices.
- Each of the next n lines contains two integers a_v and b_v ($1 \leq a_v, b_v \leq 10^9$) — the coefficients of the function at vertex v.
- Each of the next $n - 1$ lines contains two integers p and v ($1 \leq p, v \leq n$), indicating that p is the parent of v.

It is guaranteed that the sum of n over all test cases does not exceed $2 \cdot 10^5$, and the edges form a valid rooted tree with root 1.

Output

For each test case, print a single integer — the maximum possible value of $F(0)$.

It is guaranteed that, for all inputs, the maximum possible value of $F(0)$ fits in a signed 64-bit integer.

Example

standard input	standard output
2 3 1 5 4 2 2 3 1 2 1 3 5 2 1 2 3 3 1 2 2 4 1 1 2 1 3 2 4 2 5	54 148

Note

In the first test case, the root is 1 with $f_1(x) = x + 5$.

Its children are 2 $[f_2(x) = 4x + 2]$ and 3 $[f_3(x) = 2x + 3]$.

- If vertex 2 is deleted first and then vertex 3: $f_1(x) := f_3(f_2(f_1(x))) = 2(4(x+5)+2)+3 = 8x+47$. $F(0) = 47$.
- If vertex 3 is deleted first and then vertex 2: $f_1(x) := f_2(f_3(f_1(x))) = 4(2(x+5)+3)+2 = 8x+54$. $F(0) = 54$.

The maximum value is 54.

In the second test case, one optimal deletion order is $4, 5, 2, 3$. The final function is $96x + 148$, so the answer is 148.

Problem E. Helping Knuth

Input file:	`standard input`
Output file:	`standard output`
Time limit:	1 second
Memory limit:	256 megabytes

In the late 1950s, the renowned Bengali mathematician Professor Raj Chandra Bose was working to disprove Euler's long-standing conjecture on Latin Squares at Case Institute of Technology. During his visit, he noticed an exceptional undergraduate student attending his graduate level classes. Impressed by his ability, he invited the student, Donald Knuth, to assist with computational tasks.

As part of this work, Knuth needed to construct a permutation of length n, but only a partial version of the permutation is known.

He is given an array a of length n, where some elements are known and others are missing. Missing elements are represented by -1.

Your task is to help Knuth replace each occurrence of -1 in a with a value from 1 to n such that:

- The final array a becomes a valid permutation of integers from 1 to n.
- The value of the following expression is minimized:

$$\sum_{i=2}^{n}(a_i - a_{i-1})$$

If multiple valid permutations achieve the minimum value, print any of them.

Note: A permutation of length n is an array of n distinct integers from 1 to n, each appearing exactly once. For example, $[1]$, $[4,3,5,1,2]$, and $[3,2,1]$ are permutations, while $[1,1]$ and $[4,3,1]$ are not.

Input

The first line contains an integer t ($1 \le t \le 10^4$), the number of test cases. The description of the test cases follows.

For each test case, the first line contains an integer n ($2 \le n \le 2 \cdot 10^5$), the length of the array.

The second line contains n integers $a_1, a_2, \ldots, a_n$ ($-1 \le a_i \le n$, $a_i \neq 0$), where $a_i = -1$ denotes a missing value.

It is guaranteed that all positive values in a are distinct.

The sum of n over all test cases does not exceed $2 \cdot 10^5$.

Output

For each test case, print a single line containing n space-separated integers, representing the completed permutation.

If there are multiple valid answers, print any of them.

Example

standard input	standard output
2 3 -1 3 -1 5 -1 2 -1 -1 3	2 3 1 5 2 4 1 3

Note

For the first test case, the missing numbers are $\{1, 2\}$.

We try all valid completions:

- $[1, 3, 2] \Rightarrow (3 - 1) + (2 - 3) = 1$
- $[2, 3, 1] \Rightarrow (3 - 2) + (1 - 3) = -1$

The second permutation gives the minimum value, so the best answer is $[2, 3, 1]$.

Problem F. How Many Valid Collections?

Input file:	`standard input`
Output file:	`standard output`
Time limit:	4 seconds
Memory limit:	32 megabytes

For any positive integer n, let S be its decimal representation without leading zeros. A subsequence of n of length k is formed by selecting k indices $1 \le i_1 < i_2 < \cdots < i_k \le |S|$.

Two subsequences are considered **different** if their sets of chosen indices are different, even if the resulting strings of digits are identical.

Let $F(n, k, d)$ be the number of different subsequences of n of length k that consist of **exactly** d distinct digit values.

You are given a range $[L, R]$ and Q independent queries. For each query (k, d), calculate:

$$\sum_{i=L}^{R} F(i, k, d)$$

Since the result can be large, output it modulo $(10^9 + 7)$.

Input

The first line contains a single integer t $(1 \le t \le 100)$ — the number of test cases. The description of the test cases follows.

The first line of each test case contains an integer L $(1 \le L \le 10^{100})$.

The second line of each test case contains an integer R $(L \le R \le 10^{100})$.

The third line of each test case contains a single integer Q $(1 \le Q \le 10^6)$ — the number of queries.

Each of the next Q lines contains two integers k and d $(1 \le k \le |R|, 1 \le d \le 10)$.

It is guaranteed that the sum of Q over all test cases does not exceed 10^6.

It is guaranteed that the sum of $|R|^2$ over all test cases does not exceed 12000, where $|R|$ denotes the number of digits in R.

Output

For each test case, output Q lines. For each query, print a single integer — the total number of valid subsequences modulo $(10^9 + 7)$.

Example

standard input	standard output
2 1 1 1 1 1 1 12 2 1 1 1 2	1 15 0

Note

In the first test case, $L = 1$ and $R = 1$. For the query $(k = 1, d = 1)$, the only integer is 1. Its only

subsequence of length 1 is "1", which has exactly 1 distinct digit. Thus, the answer is 1.

In the second test case, $L = 1$ and $R = 12$.

For $(k = 1, d = 1)$, any single digit is a valid subsequence. Thus, $F(n, 1, 1)$ is simply the number of digits in n:

- Numbers 1 to 9 each have 1 digit, contributing $9 \times 1 = 9$.
- Numbers $10, 11, 12$ each have 2 digits, contributing $3 \times 2 = 6$.

The total sum is $9 + 6 = 15$.

For the second query $(k = 1, d = 2)$: A subsequence of length $k = 1$ contains only one digit. It is mathematically impossible for a single digit to result in $d = 2$ distinct values. Therefore, $F(n, 1, 2) = 0$ for all n, and the sum is 0.

Problem G. Last Day Harvest

Input file:	`standard input`
Output file:	`standard output`
Time limit:	2 seconds
Memory limit:	256 megabytes

The Boro harvest in the haor region of Kishoreganj is in progress. A sudden flash flood alert warns that heavy upstream rains will flood the fields. To respond, all farming must be completed **exactly within 1 day**, neither earlier nor later.

The farming consists of two operations:

- Harvesting x units of paddy
- Drying y units of paddy

Harvesting is done first, followed by drying.

The village has unlimited workers. A configuration is defined by two positive integers (p, q), where p workers are assigned to harvesting and q workers are assigned to drying. Each worker completes 1 unit of work per day, and work is evenly divided among workers. A configuration is considered **valid** if the sum of the time required to complete harvesting and the time required to complete drying is exactly 1 day.

For example, when $x = 1$ and $y = 2$:

- If $(p, q) = (3, 3)$, then harvesting takes $\frac{1}{3}$ day and drying takes $\frac{2}{3}$ day, so the total time is 1 day.
- If $(p, q) = (2, 4)$, then harvesting takes $\frac{1}{2}$ day and drying takes $\frac{1}{2}$ day, so the total time is 1 day.

Thus, both configurations are valid.

Your task is to find all valid configurations.

Input

The first line contains an integer t $(1 \le t \le 100)$, the number of test cases.

Each of the next t lines contains two integers x and y $(1 \le x, y \le 10^7)$.

Output

For each test case, print an integer n on a single line, the number of valid configurations.

Then print n lines. Each line should contain two positive integers p and q separated by a single space, representing a valid configuration.

You may print the configurations in any order.

Example

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

Problem H. Mango Mania

Input file:	`standard input`
Output file:	`standard output`
Time limit:	1 second
Memory limit:	256 megabytes

Rabiul is the quality inspector of a Bangladeshi fruit exporting company. Today, he is standing beside a conveyor belt carrying n freshly harvested Himshagor mangoes, each tagged with a **sweetness value** a_i. A buyer from New Zealand has placed an order for a *premium batch*.

A **batch** is defined as a contiguous segment of at least k mangoes from the belt. A batch is called **premium** if its representative sweetness is exactly x.

For a batch of size m with sweetness values $b_1, b_2, \ldots, b_m$, the **representative sweetness** is defined as the sweetness value of the element at position $\left\lfloor \frac{m+1}{2} \right\rfloor$ after sorting the m sweetness values in non-decreasing order. For example, the representative sweetness of the batch $[4, 7, 5, 7]$ is 5.

Help Rabiul determine whether there exists any such premium batch on the conveyor belt.

Input

The first line contains a single integer t ($1 \le t \le 10^3$) — the number of test cases. The description of the test cases follows.

The first line of each test case contains three space-separated integers n, k, and x ($1 \le k \le n \le 3 \cdot 10^5$, $1 \le x \le 10^9$) — the number of mangoes on the conveyor belt, the minimum required batch size, and the required representative sweetness.

The second line of each test case contains n space-separated integers $a_1, a_2, \ldots, a_n$ ($1 \le a_i \le 10^9$), where a_i denotes the sweetness value of the i-th mango.

It is guaranteed that the sum of n over all test cases does not exceed 10^6.

Output

For each test case, output `"Yes"` (without quotes) if there exists a premium batch. Otherwise, output `"No"` (without quotes). Output each answer on a separate line.

You may output each letter in any case (for example, strings `"yES"`, `"yes"` and `"YES"` will be recognized as a positive response).

Example

standard input	standard output
4 5 2 7 4 4 8 8 7 6 4 1 3 2 1 4 5 1 8 5 5 2 2 3 3 4 4 5 5 8 3 2 3 8 2 4 9 6 1 2	Yes Yes No Yes

Note

In the first test case, the batches (of length at least 2) with representative sweetness 7 are: $[8, 7]$, $[4, 8, 8, 7]$, and $[4, 4, 8, 8, 7]$.

In the third test case, no batch of length 5 or more has a representative sweetness equal to 5.

Problem I. Perils and Patrols

Input file:	`standard input`
Output file:	`standard output`
Time limit:	6 seconds
Memory limit:	32 megabytes

Alice has recently been appointed as the commander of a mercenary guild in the dangerous realm of *Iron Escort*. Her first contract requires her to travel safely through N consecutive regions in order, from region 1 to region N, but the roads are treacherous.

Alice starts her journey alone, with 0 guards. Before stepping into any region, she evaluates her current party size, let's denote it as u, and decides on a new party size, v. Due to logistical constraints, she can adjust her party size by at most K guards at a time. This means the absolute difference between u and v cannot exceed K, and her party size v can never be negative.

Managing her escort party requires spending coins:

- Alice pays a hiring fee of H coins for every new guard she hires.
- Alice pays a wage of S coins to every guard currently in her newly chosen party of v.

If Alice decides to dismiss guards to reach her new party size, she receives no refunds, but she successfully avoids paying wages for those dismissed individuals. Overall, her total logistical cost before entering a region is exactly $H \cdot \max(0, v - u) + S \cdot v$.

Once the logistics are settled and wages are paid, Alice's party steps into the i-th region. Danger strikes immediately upon crossing the border, where the party faces the region's innate danger level, D_i. If Alice's chosen guard count v is strictly less than D_i, bandits ambush the under-protected caravan, forcing her to pay a steep penalty of $P_i \cdot (D_i - v)$ coins, dictated by the region's specific ambush penalty multiplier, P_i.

After surviving the initial perils of the border, the party must travel deeper into the local warlord's territory. These warlords are fiercely territorial and strictly enforce a martial limit, M_i, refusing to let massive private armies freely roam their lands. If Alice's party arrives at these checkpoints with a guard count exceeding M_i, the local forces permanently confiscate the excess guards without offering any compensation. This abruptly forces her party size down to exactly M_i guards before she can proceed to the next region, whereas a smaller party is allowed to pass with its numbers intact.

Help Alice find the minimum total number of coins she needs to spend to successfully complete her contract and travel through all N regions.

Input

The first line of the input contains four integers N, K, H, and S ($1 \le N \le 10^5$, $0 \le K, H, S \le 10^4$) — the number of regions, the maximum change in guards allowed per step, the hiring cost, and the wage, respectively.

The second line contains N integers $D_1, D_2, \ldots, D_N$ ($0 \le D_i \le 2000$) — the danger levels of the regions.

The third line contains N integers $P_1, P_2, \ldots, P_N$ ($0 \le P_i \le 10^4$) — the ambush penalty multipliers.

The fourth line contains N integers $M_1, M_2, \ldots, M_N$ ($0 \le M_i \le 2000$) — the martial limits of the regions.

Output

Print a single integer denoting the minimum number of coins Alice must spend to complete her journey.

Example

standard input	standard output
2 5 10 5 3 2 100 100 1 2	65

Note

In the first step, Alice starts with 0 guards. Before entering the first region, she chooses to hire 3 guards. She pays a hiring fee of 30 coins and a wage of 15 coins. Her guard count $v = 3$ exactly matches the danger level $D_1 = 3$, so she avoids the massive ambush penalty. However, the martial limit for this region is $M_1 = 1$. Because her party size of 3 exceeds this limit, the local warlord confiscates 2 of her guards without compensation. She leaves the first region with exactly 1 guard. The cost for this step is 45 coins.

Before entering the second region, Alice currently has 1 guard. To safely navigate the danger level $D_2 = 2$, she chooses to hire 1 additional guard, bringing her party size to $v = 2$. She pays a hiring fee of 10 coins for the single new guard and a wage of 10 coins for the 2 guards currently in her party. She avoids the ambush and passes the martial limit safely since $2 \leq 2$. The cost for this step is 20 coins.

The total minimum cost to complete the journey is $45 + 20 = 65$ coins.

Problem J. Stable Gears

Input file:	`standard input`
Output file:	`standard output`
Time limit:	1 second
Memory limit:	256 megabytes

In the heart of 17th-century Dhaka, the construction of the Lalbagh Fort is underway. To secure the southern gate, the royal architect has designed a complex locking mechanism consisting of n interlocking brass gears.

The total number of gears, n, is determined by the number of days k spent on the gate's construction. On the 1st day, 1 gear was installed. On each subsequent day, the builders added the next consecutive odd number of gears (3 on the second day, 5 on the third, and so on).

Each gear in the system is unique, numbered $1, 2, \ldots, n$. A gear numbered x has exactly x teeth. For the mechanism to rotate without friction, the architect only selects gears that are *mechanically stable*.

The stability of a gear is determined by its rotational partitions. A gear with x teeth can be partitioned into m equal-sized segments, where m is any integer that divides x. A gear is **mechanically stable** only if the number of ways to partition it into segments containing an even number of teeth is perfectly balanced by the number of ways to partition it into segments containing an odd number of teeth.

For example, consider gear 6. It can be partitioned into segments of size 1, 2, 3, or 6.

- Segments of size 2 and 6 are even-sized.
- Segments of size 1 and 3 are odd-sized.

Since there are exactly two even-sized partition options and two odd-sized partition options, gear 6 is stable.

Given the number of construction days k, calculate how many mechanically stable gears are in the Lalbagh gate.

Input

The first line contains a single integer t ($1 \le t \le 10^6$) — the number of test cases.

Each of the next t lines contains a single integer k ($1 \le k \le 2^{32}$) — the number of construction days.

Output

For each test case, output a single integer — the number of mechanically stable gears among all gears numbered from 1 to n.

Example

standard input	standard output
4 1 2 5 10	0 1 6 25