Problem 01: FFT

Time limit: 2 seconds
Memory Limit: 32 MiB

FFT^2 is a revolutionary algorithm derived from FFT. All weights have been quantized to a single bit, a simplification which assumes away all complexity to achieve maximum optimality. FFT^2 computes the output vector Y from an input permutation vector X and weight bitvector W by sliding W across X, applying the mask, and taking the maximum response at each offset.

Formally,

Let $X = [x_0, x_1, \ldots, x_{n-1}]$ be the input vector, and $W = [w_0, w_1, \ldots, w_{n-1}]$ be the weight bitvector, where X **is a permutation from 1 to n** and $w_j \in \{0,1\}$. You have to compute the vector Y=FFT^2(X, W) where FFT^2 is defined as:

$$Y_i = \mathrm{FFT}^2(X, W)_i = \max_{0 \le j \le i}(x_j \cdot w_{i-j}), \quad 0 \le i < n$$

To conserve memory, input has been compressed. You are given integers n, p, q. Use the pseudocode below to generate X and W.

```
func roll(int q) -> int:
    return (q * 113 + 13337) % 1000000007 # 1e9+7

func construct(int n, int p, int q) -> Array<int>, Array<int>:
    X = [1,2,3, …, n]
    W = [0,0,0, …, 0] # size n
    for i = 0 to n-1 # n-1 inclusive
        q = roll(q)
        swap(X[i], X[q % (i + 1)])
    for i = 0 to n-1: # n-1 inclusive
        if i < p:
            W[i] = 1
        else:
            W[i] = 0
    for i = 0 to n-1: # n-1 inclusive
        q = roll(q)
        swap(W[i], W[q % (i + 1)])

    return X, W

# operation u % v is the remainder after division u by v, swap(u,
v) swaps values of u and v
```

Note the unusual memory limit in this problem.

Input:
The input consists of 3 space-separated integers satisfying the following

- $1 \leq p \leq n \leq 2 \cdot 10^5$
- $0 \leq q < 10^9 + 7$
- $q \neq 419642741$

Output:
Output N space separated integers on a single line $Y_0, Y_1, \ldots, Y_{n-1}$.

Sample Input	Sample Output
5 3 13	5 3 5 5 4

Note:
In test case 1, the uncompressed input arrays X and W are:

$$X = [5,3,4,2,1]$$

$W = [1,0,1,1,0]$

Problem 02: Whirlpool Trap Designer

Time limit: 0.5 second
Memory limit: 256 MiB

A rescue diver is caught in a circular water whirlpool. The whirlpool is shaped like a smooth bowl with maximum depth d at its center and rim radius w. The diver starts at radius r $(0 \le r < w)$ from the center and has an energy budget B. To escape, the diver must reach the rim (sea level).

For a whirlpool with radius w, the required escape energy from radius r is:

$$E(w) = g \times d \times (1 - (r / w)) + k \times (w - r)$$

where g is a gravity-like constant and k is a linear drag coefficient ($k \ge 1$). All parameters are integers, and w is constrained to be an integer.

You are designing the whirlpool radius w (integer) to trap the diver. Find the smallest integer w such that $w > r$ and $E(w) > B$.

If no such w exists up to a given limit W_{max} , output "**None**".

Input:

First line contains an integer T representing the number of test cases $(1 \le T \le 10^5)$

Each test case contains one line containing six space-separated integers: d r g k B W_{max}

Constraints: $1 \le k, d, g, W_{max} \le 10^5$

$0 \le r < W_{max}$

$0 \le B \le 10^9$

Output:

Print the smallest integer w $(r < w \le W_{max})$ such that $E(w) > B$.

If no such w exists, print "**None**".

To avoid floating point errors, compare using integer arithmetic by multiplying by w.

Sample Input	Sample Output
2 50 10 10 2 400 100 1 2 1 1 10 5	34 None

Problem 03: Air Attack Optimization

Time limit: 8 seconds
Memory limit: 512 MiB

Your country is at war with the enemy, who is much bigger than you in size and material. In a meeting of the combined military chiefs, it was analyzed that the enemy has several key ***defense hubs*** that form a hub for that area, and a very good network of transport systems that allows it to transport logistics between ***defense hubs*** in a very efficient manner. It was decided that the road/bridges infrastructure needs to be the focus of the target to create isolated pockets/regions of ***defense hubs***, which will be easier to attack and defend against. However, your country's Air Force has limited ammunition, and the targets need to be carefully selected to maximize damage using only limited resources.

As the lead scientist in the country, the air chief has asked you to join as a consultant and help identify such targets. You are provided with a map of the enemy road network (in the form of an undirected graph) that connects these ***defense hubs***, and asked to identify links/roads such that targeting them would increase the logistic disruption in the network. In particular, you are asked to identify all such roads that should be targeted such that damaging that road alone creates or increases the disconnection in the resulting logistic network (i.e., creating/increasing disconnected ***defense hubs***).

Input:

The first line contains the values N and M, where N $(0 \leq N \leq 10^4)$ is the number of ***defense hubs*** and M $(0 \leq M \leq 2 \times 10^6)$ is the number of roads. Each of the subsequent M lines contain a pair of space-separated integers u v that denote a road link between *defense hub* u and v. All the input values are integers.

Output:

The first line outputs the number of such roads identified, say K. The subsequent K lines list all such identified roads as a pair, u v, with $u<v$, one road per line, and sorted in ascending order of pairs $\{u, v\}$. If no such road exists, output '**None**'.

Sample Input 1	Sample Output 1
5 5 1 0 0 2 2 1 0 3 3 4	2 0 3 3 4

Sample Input 2	Sample Output 2
5 7 1 0 0 2 2 1 2 3 2 4 0 3 3 4	None

Problem 04: Karachi Kitchen Rat

Time limit: 1 second
Memory limit: 512 MiB

A determined rat is at the base of an old apartment building in Karachi's Saddar neighbourhood, eyeing an open kitchen window H centimetres above the ground. The rat starts at height 0.
The crumbling exterior wall has N ledges: cracked bricks, exposed pipes, and protruding AC units that the rat can use as footholds. Each ledge i is located at a distinct integer height $h[i]$ (where $0 < h[i] < H$) and has a slip probability $f[i]$ depending on how weathered or greasy it is.
The rat carries B pieces of chapli kebab stolen from a nearby street vendor. When consumed, a kebab gives the rat a burst of energy that may help it leap to the next ledge.

Time proceeds in discrete days. Each day consists of two phases:

Phase 1: The rat chooses exactly one of the following actions:

1. **Climb**: The rat attempts to scramble up the wall.

- With probability p_up: the rat moves up by 2 centimetres (**new height = current height + 2**).

- With probability $1 - p\_up$: the climb fails and the rat stays at its current height.

2. **Eat Kebab**: This action is only available if both conditions hold:

    1. The rat has at least one kebab remaining ($B > 0$).
    2. There exists at least one ledge strictly above the rat's current height.

    When used:

    - The kebab is consumed (regardless of success or failure).
    - With probability p_boost: the rat gets a burst of energy and leaps to the lowest ledge strictly above its current height.
    - With probability $1 - p\_boost$: the kebab was too spicy and the rat stays at its current height, recovering.

Phase 2: Immediately after the day action:

- If the rat's height is $\geq H$, the rat enters the kitchen and the process ends. This day counts as the final day.
- If the rat has not entered, proceed to Phase 3.

Phase 3: The rat may slip during the night:

- Let x be the rat's current height.
- The slip probability at height x is:
    - $f[i]$ if $x = h[i]$ for some ledge i

- f_0 (the default slip probability for the bare wall) otherwise

- With probability $f(x)$: the rat slips and falls by 1 centimetre (**new height = max(0, x - 1)**).
- With probability $1 - f(x)$: the rat maintains its position.

The day then ends, and a new day begins.

The rat plays optimally to minimize the expected number of days until it enters the kitchen. Compute this minimum expected number of days, starting from height 0 with B kebabs.

Input:

The first line contains three integers: H ($2 \le H \le 2000$), N ($0 \le N \le 500$), and B ($0 \le B \le 10$).

The second line contains three real numbers: p_up ($0.5 \le p\_up \le 1.0$), p_boost ($0 \le p\_boost \le 1.0$), and f_0 ($0 \le f_0 \le 0.9$)

The next N lines each contain an integer $h[i]$ ($0 < h[i] < H$) and a real number $f[i]$ ($0 \le f[i] \le 0.9$), describing a ledge at height $h[i]$ with slip probability $f[i]$. All $h[i]$ are distinct.

The input is guaranteed such that the expected number of days to enter the kitchen is finite.

Output:

Output a single real number: the minimum expected number of days for the rat to enter the kitchen.
Your answer will be accepted if the relative error from expected output does not exceed 10^{-6} i.e. **1e-6**

Sample Input 1	Sample Output 1
4 0 0 1.0 0.5 0.3	2.300000

Explanation: With $p\_up = 1.0$, every climb succeeds. **Day 1**: the rat climbs from 0 to 2. Since $2 < H = 4$, it hasn't entered yet. At night, it slips to 1 with probability 0.3 or stays at 2 with probability 0.7. **Day 2**: From height 2, climbing reaches $4 \ge H$ and the rat enters. From height 1, climbing reaches $3 < H$, requiring more days. The expected value accounts for these probabilistic paths.

Sample Input 2	Sample Output 2
6 2 1 0.5 0.9 0.5 2 0.1 4 0.1	5.862580

Problem 05: Balance Bulbs

Time limit: 5 seconds
Memory Limit: 1024 MiB

You are given **N** bulbs arranged in a straight line and numbered from **1** to **N**. Each bulb can be in one of two states:

1. **OFF**
2. **ON**

You must choose a state (**OFF** or **ON**) for every bulb, forming a sequence of length **N**. You are also given **M** conditions:

- The **i-th** condition consists of two integers L_i and R_i $(1 \le L_i < R_i \le N)$

For each condition:

- Consider all bulbs from position L_i to position R_i, inclusive.
- In this range, the number of bulbs that are **ON** must be equal to the number of bulbs that are **OFF**.

Among all bulb state sequences that satisfy all **M** conditions, find the lexicographically smallest sequence. Lexicographical order is defined as follows:

- **OFF** is considered smaller than **ON**.
- A sequence **X** is lexicographically smaller than sequence **Y** if, at the first position where they differ, **X** has **OFF** and **Y** has **ON**.

It is guaranteed that at least one valid bulb sequence exists that satisfies all conditions.

Return the lexicographically smallest valid sequence as a string of **0**s and **1**s, where:

- **0** represents an **OFF** bulb, and
- **1** represents an **ON** bulb.

Input

The first line contains an integer N $(2 \le N \le 10^6)$, the number of bulbs.

The second line contains an integer M $(1 \le M \le 200000)$, the number of conditions.

The next **M** lines each contain two space-separated integers L_i and R_i $(1 \le L_i < R_i \le N)$, representing a condition on the bulbs from position L_i to R_i.

For every condition, the length of the segment satisfies $(R_i - L_i + 1)$ is even.

No two conditions are identical, i.e., $(L_i, R_i) \ne (L_j, R_j)$ for $i \ne j$.

All values in the input are integers.

Output

Output a string of length **N** consisting only of characters **0** and **1**, where

0 represents an **OFF** bulb and **1** represents an **ON** bulb.

Sample Input 1	Sample Output 1
4 2 1 2 3 4	0101

Sample Input 2	Sample Output 2
6 2 1 4 3 6	001100

Problem 06: The Lost Artifact of Eldoria

Time limit: 2 seconds
Memory limit: 512 MiB

Long ago, in the ancient kingdom of Eldoria, there existed a legendary artifact known as the Orb of Luminara. Legends say the orb could illuminate hidden truths and was safeguarded in a secret chamber. Over centuries, its location was lost, though scholars chronicled hints scattered across the kingdom.

You are part of a modern expedition tasked with recovering this unique artifact. Eldorian records mention three notable relics, whose positions are precisely known:

- The Statue of Auralis
- The Crystal Monolith of Veyra
- The Obelisk of Tharion

These relics serve as beacons. While the beacons' coordinates are known, the directions to the orb remain a mystery. The text on each Beacon provides only the distance d_i from the relic to the orb, but not the direction and the Orb is somewhere on the circle with center on the relic location and radius d_i. Scholars have verified these distances.

Your mission is to find the exact location of the Orb of Luminara on a discrete grid of known locations. Each cell in the grid contains a word, a code for the relic, which may repeat for identical copies of relics or be unique. But there is only one orb and a unique word corresponds to the hidden orb.

Input

- The first line contains integer n, $1 \leq n \leq 1000$, representing the size of the grid, e.g., n=**10** means a grid of **10x10**.
- The next n lines contain n words each, separated by spaces. Words contain only lowercase letters and have length **1–15**.
- The next line contains integer $k = 3$, the number of beacons.
- Next k lines contain three space-separated numbers each: $r_i\ c_i\ d_i$, where r_i, c_i $(0 \leq r_i, c_i < n)$ are the row and column of the relic, and d_i is the Euclidean distance to the orb.

Notes

- Beacons are distinct and non-collinear, and do not coincide with the hidden artifact.
- The hidden artifact is unique in the grid, but other words may appear once or multiple times.
- The distances are floating-point numbers; due to rounding, comparing distances using exact equality may fail. It is recommended to allow a small tolerance, when determining if a point satisfies all distance constraints.

Output

For each test case, print a single word: the **word at the cell containing the orb**.

Constraints

3. $1 \leq n \leq 1000$

4. Grid words: lowercase English letters, **length ≤ 15**

5. Beacons: exactly **3**, all non-collinear

6. The hidden orb is unique in the grid.

7. Other words may repeat or be unique but do not identify the orb.

8. Distances d_i are floating points, have up to **6** decimal places and may not match exactly.

Sample Input 1	Sample Output 1
10 alpha bravo charlie delta echo foxtrot golf hotel india juliet kilo lima mike november oscar papa quebec romeo sierra tango uniform victor whiskey xray yankee zulu alpha bravo charlie delta echo foxtrot golf hotel india juliet kilo lima mike november oscar papa quebec romeo sierra tango uniform victor whiskey xray yankee zulu alpha bravo charlie delta echo foxtrot golf hotel india juliet kilo lima mike november oscar papa quebec romeo sierra tango uniform victor whiskey xray luminar alpha bravo charlie delta echo foxtrot golf hotel india juliet kilo lima mike november oscar papa quebec romeo sierra tango uniform victor 3 0 0 9.219544 0 9 7.615773 9 0 6.324555	luminar

Explanation

The hidden orb is at coordinates **(7,6)**, corresponding to the word **luminar**. The distances to the three relics match exactly:

- **Statue of Auralis (0,0):** $\sqrt{(7-0)^2+(6-0)^2} = \sqrt{85} \approx 9.219544$
- **Crystal Monolith of Veyra (0,9):** $\sqrt{(7-0)^2+(6-9)^2} = \sqrt{58} \approx 7.615773$
- **Obelisk of Tharion (9,0):** $\sqrt{(7-9)^2+(6-0)^2} = \sqrt{40} \approx 6.324555$

Diagram of Grid and Relics

Tharion

(9,0) T	(9,1)	(9,2)	(9,3)	(9,4)	(9,5)	(9,6)	(9,7)	(9,8)	(9,9)
(8,0)	(8,1)	(8,2)	(8,3)	(8,4)	(8,5)	(8,6)	(8,7)	(8,8)	(8,9)
(7,0)	(7,1)	(7,2)	(7,3)	(7,4)	(7,5)	(7,6) Orb O	(7,7)	(7,8)	(7,9)
(6,0)	(6,1)	(6,2)	(6,3)	(6,4)	(6,5)	(6,6)	(6,7)	(6,8)	(6,9)
(5,0)	(5,1)	(5,2)	(5,3)	(5,4)	(5,5)	(5,6)	(5,7)	(5,8)	(5,9)
(4,0)	(4,1)	(4,2)	(4,3)	(4,4)	(4,5)	(4,6)	(4,7)	(4,8)	(4,9)
(3,0)	(3,1)	(3,2)	(3,3)	(3,4)	(3,5)	(3,6)	(3,7)	(3,8)	(3,9)
(2,0)	(2,1)	(2,2)	(2,3)	(2,4)	(2,5)	(2,6)	(2,7)	(2,8)	(2,9)
(1,0)	(1,1)	(1,2)	(1,3)	(1,4)	(1,5)	(1,6)	(1,7)	(1,8)	(1,9)
(0,0) A	(0,1)	(0,2)	(0,3)	(0,4)	(0,5)	(0,6)	(0,7)	(0,8)	(0,9) V

Auralis

Veyra

Problem 07: Chromatic Equilibrium

Time limit: 1 second
Memory limit: 256 MiB

Dr. Combino has a collection of $2n$ colored balls. The balls come in k distinct colors. You are provided with an integer array B of size k, where b_i represents the number of balls of the i^{th} color. It is guaranteed that the sum of all balls is an even number.

Dr. Combino decides to perform a randomized experiment. He takes all $2n$ balls, shuffles them uniformly at random, and then distributes them into two distinct boxes: **Box A** and **Box B**.

- The first n balls from the shuffled sequence are placed into **Box A**.
- The remaining n balls are placed into **Box B**.

Because the boxes are distinct, the order of distribution matters. For instance, if there is one ball of color u and one ball of color v, the distribution **[Box A: {u}, Box B: {v}]** is considered different from **[Box A: {v}, Box B: {u}]**.

For example, if we have **2** balls, **1** of color #0 and **1** of color #1, then there are only two possible permutations of the balls = $\frac{2!}{1!1!}$ **=2**, which are **(0, 1)** and **(1, 0)**

1. Permutation **(0, 1)**: **Box A** gets **{0}**, **Box B** gets **{1}**. Distinct colors: **A=1, B=1**. (Match)
2. Permutation **(1, 0)**: **Box A** gets **{1}**, **Box B** gets **{0}**. Distinct colors: **A=1, B=1**. (Match)
   Thus, Probability = **2/2 = 1.000000**

Now, if we have **4** balls total, **2** of color #0, **1** of color #1, **1** of color #2, then total permutations = $\frac{4!}{2!1!1!}$ **= 12**. After distributing **2** balls in **Box A** and **Box B** each, out of these **12** cases, there are **8** cases where the count of unique colors in **A** equals the count of unique colors in **B**. Thus, probability **= 8/12 = 0.666667**.

Your task is to calculate the probability that, after the distribution, **Box A** and **Box B** contain the same number of **distinct** colors.

Input:

The first line of input contains a single integer k $(1 \leq k \leq 10)$, denoting the number of distinct colors. The second line contains k space-separated integers b_i $(1 \leq b_i \leq 10)$ where b_i is the count of balls of color i.

It is guaranteed that the sum of all b_i is even.

Output:

Print the probability as a floating-point number with exactly 6 digits after the decimal point. Answers within 10^{-6} absolute error will be accepted.

Sample Input 1	Sample Output 1
2 1 1	1.000000

Sample Input 2	Sample Output 2
3 2 1 1	0.666667

Sample Input 3	Sample Output 3
4 1 2 1 2	0.600000

Problem 08: OnlyQueens

Time limit: 1 second
Memory limit: 256 MiB

You are given a rectangular chess board. A new piece is introduced that can move both like a queen and a knight. Now, imagine that piece is placed in each and every cell of the chessboard.

What are the number of attacks on all cells modulo **1,000,000,007**?

Note that there can be multiple attacks on a cell. Also note that if a cell is occupied by a piece, it is not considered under attack by that piece.

Things to remember:

- A queen can move vertically, horizontally and diagonally in any direction.
- A knight can move in an L-shape, from a cell (i, j) to (i ± 2, j ± 1) or (i ± 1, j ± 2)

Input:

The first line contains an integer T **($1 \leq T \leq 200{,}000$)**, representing the number of test cases.

Each of the following **T** lines contain two numbers, **W** and **H**, representing the width and height of the rectangular chessboard **($2 \leq W \leq H \leq 10^9$)**

Output:

For each test case, print the output modulo **1,000,000,007**.

Sample Input	Sample Output
2 3 3 5 4	72 288

Problem 09: Badshah ka Wazir kaun?

Time limit: 10 seconds
Memory limit: 256 MiB

The royal court is in turmoil. The Great Jewel of the Realm has vanished, and the King demands justice. You have been summoned as a neutral observer to identify the three key players hiding among the courtiers:

- The **Badshah**: The ruler, who considers lying beneath his dignity.
- The **Chor**: The thief, who steals not just jewels but the truth itself.
- The **Wazir**: The envious minister, who weaves a web of truth and lies to serve his own ambition.
- All other courtiers are ordinary **Sipahis**, loyal soldiers who always speak honestly.

You cannot speak to anyone directly. All communication goes through the **Qadi**, the court clerk. You may ask questions about any courtier, and the Qadi will relay their answer back to you as a simple **Yes** or **No**.

However, be warned: the Qadi is fiercely protective of his King. **Exactly once**, when someone is asked to identify the true Badshah, the Qadi will deliberately misreport the answer to protect the monarch's anonymity. After that single act of loyalty, he will report all answers faithfully.

The Wazir is a complex figure who maintains two separate mindsets depending on the subject of your inquiry. His ambition makes him jealous of the **Badshah**; thus, when questioned about the King (**B**-type), he begins with a **lie**, but cannot sustain it, alternating strictly between truth and lies thereafter (Lie, Truth, Lie...). Conversely, he fears being implicated in the crime; when asked about the **Chor** (**C**-type), his desire to appear helpful compels him to speak the **truth** initially, but his deceptive nature soon returns, causing him to alternate in the opposite pattern (Truth, Lie, Truth...). These two behavioral tracks operate independently.

The court's patience is finite. You have enough time to question everyone if you are efficient, but waste too much time and you will be dismissed.

There are N courtiers sitting in a circle, numbered from **1** to N.

- Exactly one is the Badshah.
- Exactly one is the Chor.
- Exactly one is the Wazir.
- The remaining $N - 3$ are Sipahis.

The roles are fixed and hidden.

Interaction:

Each interaction begins with two space-separated integers: **N** and **Q** where **Q** is the number of queries.
Each query must be of the form:

```
? i j T
```

where:

- i $(1 \leq i \leq N)$ is the courtier being questioned.
- **j** $(1 \leq j \leq N)$ is the subject of the question.
- T is one of:
  - **B**: "Is person **j** the Badshah?"
  - **C**: "Is person **j** the Chor?"

The judge replies with:

- **1** meaning **Yes**
- **0** meaning **No**

Final Answer

When you have identified the roles, output:

```
! b c w
```

where b, c, w are the indices of the Badshah, Chor, and Wazir respectively.

Constraints:

- $4 \leq N \leq 10000$
- The number of allowed queries **Q** is sufficient to solve the problem using an efficient logical deduction.
- All roles are guaranteed to exist.

Sample Interaction:

Sample Input 1	Sample Output 1
4 4 0 1 1 1	? 1 1 B ? 1 1 B ? 1 3 C ? 2 2 B ! 1 3 2

Problem 10: Optimal Response Distances

Time limit: 1 seconds
Memory limit: 256 MiB

A social media platform is battling the rapid spread of misinformation. The platform is modeled as a directed network of **N** users and **M** connections. A directed edge **u→v** indicates that user **v** follows user **u** and will see any content **u** posts.

When "fake news" is posted by an arbitrary user x, it propagates through the network along the shortest directed paths. To combat this, the platform has organized several fact-checking teams. A team **S** is a set of **k** users who have the tools to immediately flag and stop the spread of misinformation once it reaches them.

For a proposed team **S**, the platform evaluates its effectiveness using the Total Response Distance. For every user x not in **S**, let $d(x, S)$ be the length of the shortest directed path from x to any member of **S**. The Total Response Distance is the sum of these distances over all users $x \notin S$. If there is at least one user x who cannot reach any member of the fact-checking team S through a directed path, the team is considered ineffective for that segment of the network.
Your task is to calculate the Total Response Distance for several proposed teams.

Input

- The first line contains two integers N $(1 \le N \le 1{,}000)$ and M $(0 \le M \le 2{,}000)$, representing the number of users and directed connections.
- The next **M** lines each contain two integers **u** and **v** $(0 \le u, v < N, u \ne v)$, indicating a directed edge **u→v**.
- The next line contains an integer Q $(1 \le Q \le 1{,}000)$, the number of proposed fact-checking teams.
- Each of the next **Q** lines describes a team:
  An integer k $(1 \le k \le 50)$
  Followed by **k** distinct integers representing the 0-indexed IDs of the team members.

Output

For each team, print a single integer: the Total Response Distance. If any user x cannot reach at least one member of the team **S**, print **-1**.

Sample Input 1	Sample Output 1
6 7 0 1 1 2 2 3 3 4 1 4 4 5 2 5 2 2 4 5 1 4	5 -1

Problem 11: Collatz Fourier Transform (Hard Version)

Time limit: 0.5 seconds
Memory limit: 256 MiB

You are given $P(x) \in F_2[x]$ (coefficients **0** or **1**). Perform the following until **P(x) = 1**:

- **Butterfly**: If the constant term is **1**, compute **P(*x*) x (*x* + 1) + 1** over **Z** (set of integers), then apply coefficient-wise mod2 reduction
- **Decimation**: If the constant term is **0**, divide **P(*x*)** by ***x***

Task: Output the number of operations required.

Input:

First line: $N\ (0 \le N \le 20)$, the degree of the polynomial.

Second line: **N** space-separated integers, representing coefficients (**0** or **1**), the last of which is always **1**

Output:

Print a single integer: the number of operations to reach **P(*x*)=1**.

Sample Input 1	Sample Output 1
3 1 0 0 1	11

Explanation:

Starting with **P(*x*) = *x*^3 + 1**:S

1. Butterfly: ***(x^3+1)(x+1)+1 = x^4+x^3+x+2 => x^4+x^3+x***
2. Decimation: ***x^3+x^2+1***
3. Butterfly: ***x^4+x^2+x***
4. Decimation: ***x^3+x+1***
5. Butterfly: ***x^4+x^3+x^2***
6. Decimation: ***x^3+x^2+x***
7. Butterfly: ***x^2+x+1***
8. Decimation: ***x^3***
9. Decimation: ***x^2***
10. Decimation: ***x***
11. Butterfly: **1**

Total: **11** operations.

Problem 12: The Sufi Trail Guardians

Time limit: 2 seconds
Memory limit: 256 MiB

Along the ancient Grand Trunk Road that winds through the Punjab, there lie **N** sacred Sufi shrines (dargahs) numbered **1** to **N**. These spiritual centers are connected by **N-1** pilgrimage pathways forming a tree structure. There is exactly one route between any two shrines, with no cycles.

The Urs season approaches, and the Waqf Board must station qawwali ensembles at certain shrines. A shrine is considered spiritually covered if it either hosts an ensemble itself, or is directly connected by a pathway to a shrine that hosts one. Each shrine ***i*** has a specific hosting cost **C[i]** (in thousands of rupees) due to varying sizes of the courtyards.

Determine the minimum total expenditure required to ensure every shrine is spiritually covered.

Input:

- The first line contains an integer $\mathbf{N(2 \le N \le 2 \times 10^5)}$, the number of shrines.
- The second line contains **N** integers **C[1]**, **C[2]**, **C[3]**, …, **C[N]** ($1 \le C[i] \le 10^9$), where **C[i]** is the cost of hosting an ensemble at shrine **i**.
- The next **N-1** lines each contain two integers **u** and **v** $\mathbf{(1 \le u, v \le N, u \ne v)}$, indicating a pathway connecting shrine **u** and **v**. It is guaranteed that the given edges form a tree kind of structure.

Output:

Output a single integer: the minimum total cost to ensure every shrine is spiritually covered.

Constraints

- $\mathbf{2 \le N \le 2 \times 10^5}$
- $\mathbf{1 \le C[i] \le 10^9}$

Sample Input 1	Sample Output 1
3 1 2 3 1 2 2 3	2

Sample Input 2	Sample Output 2
4 10 1 1 1 1 2 1 3 1 4	3