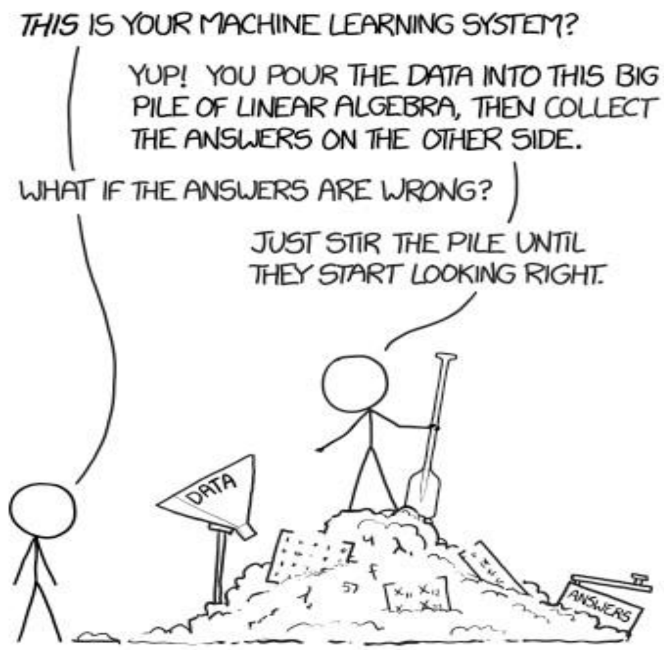
Main Round
23 Dec. 2021

Fuzzy Pairs

Time limit: 1 s

Memory limit: 512 MB

After learning about fuzzy logic, Barish decided to use it to train his brand-new machine learning model for predicting the results of IOI 2022. Although his previous model was achieving 99.1921% accuracy in the validation set from IOI 2019, Barish wants the new model to work perfectly. However, he also realized that applying fuzzy logic will not suffice: he also needs to design a better evaluation algorithm to improve the model. This is where he came up with the “fuzzy pairs” validation algorithm that allows Barish to train his model in a semi-supervised learning.



In short, he creates N disjoint pairs using all numbers from 1 to $2N$ (inclusive). Then, he gives the model an access to a function `distinct(S)` that takes an argument `S` which is a subset of the set $\{1, 2, 3, \ldots, 2N\}$ and reports number of *distinct pairs that have at least one element in that set*. For example, if $N = 2$ and the pairs are $(1, 4)$ and $(2, 3)$, the following results will be true: `distinct({1})=1`, `distinct({1,2})=2`, `distinct({1,2,3})=2`, `distinct({1,4})=1`. The final goal is to find out all pairs.

Obviously, fuzzy gradient descent is a perfect algorithm to train this model, but Barish ran out of free GPU usage in Google Colaboratory, so he needs your urgent help to manually simulate his model so he knows if the model achieves 100% accuracy.

Implementation details

This is an interactive task. You will be given the file "`train.cpp`" in which you should implement the function `train(N)`:

`vector<pair<int, int>> train(int N);` This function will be called once. It can call the function `distinct(S)` several times. At the end, it should return the correct pairs in any order. Please note that each pair should be a `std::pair<int, int>` object and they should be returned in a `std::vector` container.

`int distinct(vector<int> S);` This function will take the subset described in the problem statement and will return the number of distinct pairs that have at least one element inside the subset. If `S` violates the rule of being the subset of $\{1, 2, 3, \dots, 2N\}$, the function will return -1. If you ever get -1, you should immediately terminate the function `train(N)`, otherwise you may get the verdict "`Execution timed out`".

Note: *In case you don't use the provided file "`train.cpp`", do not forget to write function prototype "`int distinct(vector<int> S);`" in your code, otherwise you will get the verdict* "`Compilation failed`".

Local testing

You will be given the file "`grader.cpp`", which you can compile together with your program to test it. It will read N pairs in any order (first two numbers represent the first pair, the next two numbers represent the second pair, etc.) and then call the function `train(N)` and check your answer.

Constraints

- $N = 128$
- $q \le 2^{15}$, here q is the number of calls you make to the function `distinct(S)`. Note that the score you will get from this problem will depend on this constraint. See the scoring section below.

Sample Interaction

`train(2)` function starts:

`distinct({1,2})` returns 2,

`distinct({1,4})` returns 1,

The function returns a vector container with pairs $\{1,4\}$ and $\{2,3\}$. Indeed, this is the correct answer. This example is from the problem statement.

Scoring

If you ask any wrong query or the number of queries exceeds the limit 2^{15} or just the set of pairs you found is wrong, you will get 0 points and the verdict "`Wrong Answer`". In other cases, your score will be determined as follows:

- If $2^{14} < q \leq 2^{15}$, you will get 5 points.
- If $2^{13} < q \leq 2^{14}$, you will get 13 points.
- If $2^{12} < q \leq 2^{13}$, you will get 23 points.
- If $2^{11} < q \leq 2^{12}$, you will get 37 points.
- If $2^{10} < q \leq 2^{11}$, you will get 63 points.
- If $960 < q \leq 2^{10}$, you will get 96 points.
- If $q \leq 960$, you will get 100 points.

Fuzzy Graph

Time limit: 1 s

Memory limit: 512 MB

Rashad is a famous researcher in the Institute of Lovely Graphs. His latest research focuses on determining the social comfort of the given people network, represented as a graph. After long and detailed observations, Rashad hypotheses that the properties of coprime numbers are significant in social environments. Thus, he defines a fuzzy graph as a simple graph (undirected graph without any self-loops or multiple-edges) in which for all edges (u, v), the property $gcd(degree[u], degree[v]) = 1$ is true (i.e., degrees of vertices u and v are coprime). The degree of a vertex is defined as the number of edges it is connected to. Note that the fuzzy graph does not need to be connected.

To test his hypothesis, Rashad invited N previous IOI participants to his renowned experiment. He wants to prove that even with this extreme setting, he can achieve 100% social comfort. That is why he wants to create a fuzzy graph with N vertices and *as many edges as possibl*e to apply it to the experiment subjects and observe the situation. Your job, as Rashad's research assistant, is to do his all work (i.e., design the fuzzy graph) while he checks out other lovely graph problems to focus on in the future.

Input

You are given one integer N – the number of vertices the fuzzy graph will need to have. The vertices will be numbered as $1, 2, \ldots, N$ in the graph.

Output

In the first line, print M, the number of edges you have achieved in your fuzzy graph. Print the edges in the next M lines as pairs of numbers. Note that if the printed graph is not fuzzy, you will get 0 points as the experiment subjects will leave because of high levels of social anxiety.

Constraints

- $3 \leq N \leq 1000$

Examples

Input	Output	Explanation
3	1 1 2	This is a correct answer, but jury has a better answer with 2 edges. So, the score for such an answer will be: $$2 \times 0.01^{1-\frac{1}{2}} = 0.2 \text{ points.}$$ See the scoring section below.

Scoring

In this task there are **50 tests**, each worth **2 points**. These 50 different values of N are given in a text file named **"`tests.txt`"** as an attachment. Please download it from the *"Attachments"* section of the task in the judging system.

Each test will be scored separately.

Let E be the number of edges in contestant's and J in jury's answer in a particular test.

- If $E = 0$ or the graph is not fuzzy, you will get 0 points and "`Wrong Answer`".
- If $0 < E < J$, you will get $2 \times 0.01^{1-\frac{E}{J}}$ points.
- If $E >= J$, you will get 2 points.

Fuzzy Uniqueness

Time limit: 3 s

Memory limit: 512 MB

After noticing Barish's machine learning attempts to predict IOI 2022 results and Rashad's social comfort experiment, Rahim thought he should do something *unique* instead of those boring activities. However, it is very hard to find something unique to do, since everyone is doing everything these days! The only thing Rahim could think of doing turned out to be finding the uniqueness of the activities people do. So given a sequence of activities, Rahim defined its fuzzy uniqueness as the number of contiguous subarrays of the given sequence that have uniqueness value between L and R. The uniqueness value of the subarray of activities is defined as the number of activities that only occurs once in the subarray (note that it still may occur more than once in the entire sequence).

Formally, Rahim represents all activities as an array of N integers, and he wants to count the number of contiguous subarrays that have uniqueness value between L and R (inclusive), where the uniqueness value of a subarray is defined as the number of unique elements in it. Unfortunately, Rahim himself got fuzzy thinking about all this complex stuff, but you are here to help him answer his question of fuzzy uniqueness.

Input

In the first line, you are given integers N, L, R, the number of total activities and two bounds defining fuzzy uniqueness, in order. The next line consists of N integers, $a_1, a_2, \ldots, a_N$, the representation of the activities.

Output

Print the number of contiguous subarrays that have uniqueness value between L and R (inclusive).

Constraints

- $1 \leq N \leq 150000$
- $0 \leq L \leq R \leq N$
- $1 \leq a_i \leq 10^9$

Examples

Input	Output	Explanation
5 1 2 1 2 2 1 3	12	Contiguous subarrays that have uniqueness value 1: [1,1], [2,2], [3,3], [4,4], [5,5], [1,3], [2,4], [1,5] Contiguous subarrays that have uniqueness value 2: [1,2], [3,4], [4,5], [2,5] Here $[i, j]$ indicates the subarray $a_i, \dots, a_j$

Subtasks

This task contains 5 subtasks as described below:

Subtask	Additional constraints	Scoring
1	$N \leq 100$	8 points
2	$N \leq 2000$	13 points
3	$L = R = 1$ and the number of distinct integers in the activities array is at most 500	18 points
4	L = R = 0	22 points
5	No additional constraints	39 points

Christmas Tree

Time limit: 3 s

Memory limit: 512 MB

Having so many "Fuzzy" stuff may get you tired and since it is New Year's Eve, Jury decided to give you a Christmas Tree. Now it is up to you to make this tree as beautiful as you can.

A Christmas Tree is a tree consisting of N nodes connected by $N-1$ edges. At each node of the Christmas Tree there is a bulb. The more lightful the tree, the more beautiful it is. So why not just to light up all the bulbs? The problem is that you have only one socket.

You will choose a subset of bulbs to be lit up, and a *source* node where you will put the socket. The *intensity* of light at the socket is N, and it decreases by 1 unit when it travels through an edge. On top of this, each bulb has a heat rate h_i, which negatively affects the beauty of the tree. So, the beauty of your Christmas tree will decrease by $max(h_x)$ (maximum of all h_x, where x is any bulb from your chosen subset).

Formally the beauty value of the Christmas tree will be $(\sum intensity(x)) - max(h_x)$ where x is any bulb from your chosen subset, and *intensity* is defined as aforementioned.

Find maximum possible beauty of the given Christmas Tree.

Input

In the first line, you are given an integer N – the number of nodes (as well as bulbs) in the Christmas Tree. The second line consists of N integers, $h_1, h_2, \dots, h_N$ – the heat rates of the bulbs. In each of the next $N-1$ lines you will be given two integers u_i and v_i – the edges of the tree.

Output

Print the maximum possible beauty of the given Christmas Tree.

Constraints

- $1 \le N \le 2 \cdot 10^5$
- $0 \le h_i \le 10^{18}$
- $1 \le u_i, v_i \le N$, *the given edges form a tree.*

Examples

Input	Output	Explanation
7 1 1 1 1 1 1 1 1 2 1 3 2 4 2 5 3 6 3 7	38	It is optimal to include all the bulbs in your chosen subset, and to choose node 1 as *source* node to put the socket at. Then the *intensity* at node1 = 7, node2 = 6, node3 = 6, node4 = 5, node5 = 5, node6 = 5, node7 = 5 and $max(h_x) = 1$. So, the maximum beauty is: $7 + 6 + 6 + 5 + 5 + 5 + 5 - 1 = 38$
7 100 1 1 100 100 100 100 1 2 1 3 2 4 2 5 3 6 3 7	11	It is optimal to include the bulbs on nodes 2 and 3 in your chosen subset, and to choose node 1 as *source* node to put the socket at. Then the *intensity* of node2 = 6, node3 = 6, $max(h_x) = 1$. So, the maximum beauty is: $6 + 6 - 1 = 11$

Subtasks

This task contains 5 subtasks as described below:

Subtask	Additional constraints	Scoring
1	$N \leq 16$	11 points
2	$N \leq 2000$	17 points
3	Each node has at most 2 neighbors.	17 points
4	The given tree is a perfect binary tree with root node at 1. That is $N = 2^k - 1$ for some k and all the edges are of the form $(i, 2i)$ or $(i, 2i + 1)$.	23 points
5	No additional constraints	32 points