A. Captain Flint and Crew Recruitment

time limit per test: 1 second
memory limit per test: 256 megabytes
input: standard input
output: standard output

Despite his bad reputation, Captain Flint is a friendly person (at least, friendly to animals). Now Captain Flint is searching worthy sailors to join his new crew (solely for peaceful purposes). A sailor is considered as worthy if he can solve Flint's task.

Recently, out of blue Captain Flint has been interested in math and even defined a new class of integers. Let's define a positive integer x as **nearly prime** if it can be represented as $p \cdot q$, where $1 < p < q$ and p and q are prime numbers. For example, integers 6 and 10 are nearly primes (since $2 \cdot 3 = 6$ and $2 \cdot 5 = 10$), but integers 1, 3, 4, 16, 17 or 44 are not.

Captain Flint guessed an integer n and asked you: can you represent it as *the sum of* 4 ***different positive*** *integers* where **at least** 3 of them should be *nearly prime*.

Uncle Bogdan easily solved the task and joined the crew. Can you do the same?

Input

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

Next t lines contain test cases — one per line. The first and only line of each test case contains the single integer n $(1 \le n \le 2 \cdot 10^5)$ — the number Flint guessed.

Output

For each test case print:

- YES and 4 **different** positive integers such that at least 3 of them are nearly prime and their sum is equal to n (if there are multiple answers print any of them);
- NO if there is no way to represent n as *the sum of* 4 ***different positive*** *integers* where at least 3 of them are nearly prime.

You can print each character of YES or NO in any case.

Example

input

```
7
7
23
31
36
44
100
258
```

output

```
NO
NO
YES
14 10 6 1
YES
5 6 10 15
YES
6 7 10 21
YES
2 10 33 55
YES
10 21 221 6
```

Note

In the first and second test cases, it can be proven that there are no four different positive integers such that at least three of them are nearly prime.

In the third test case, $n = 31 = 2 \cdot 7 + 2 \cdot 5 + 2 \cdot 3 + 1$: integers 14, 10, 6 are nearly prime.

In the fourth test case, $n = 36 = 5 + 2 \cdot 3 + 2 \cdot 5 + 3 \cdot 5$: integers 6, 10, 15 are nearly prime.

In the fifth test case, $n = 44 = 2 \cdot 3 + 7 + 2 \cdot 5 + 3 \cdot 7$: integers 6, 10, 21 are nearly prime.

In the sixth test case, $n = 100 = 2 + 2 \cdot 5 + 3 \cdot 11 + 5 \cdot 11$: integers 10, 33, 55 are nearly prime.

In the seventh test case, $n = 258 = 2 \cdot 5 + 3 \cdot 7 + 13 \cdot 17 + 2 \cdot 3$: integers 10, 21, 221, 6 are nearly prime.

B. Captain Flint and a Long Voyage

time limit per test: 2 seconds
memory limit per test: 256 megabytes
input: standard input
output: standard output

Captain Flint and his crew keep heading to a savage shore of Byteland for several months already, drinking rum and telling stories. In such moments uncle Bogdan often remembers his nephew Denis. Today, he has told a story about how Denis helped him to come up with an interesting problem and asked the crew to solve it.

In the beginning, uncle Bogdan wrote on a board a positive integer x consisting of n digits. After that, he wiped out x and wrote integer k instead, which was the concatenation of binary representations of digits x consists of (without leading zeroes). For example, let $x = 729$, then $k = 111101001$ (since $7 = 111$, $2 = 10$, $9 = 1001$).

After some time, uncle Bogdan understood that he doesn't know what to do with k and asked Denis to help. Denis decided to wipe last n digits of k and named the new number as r.

As a result, Denis proposed to find such integer x of length n that r (as number) is maximum possible. If there are multiple valid x then Denis is interested in the minimum one.

All crew members, including captain Flint himself, easily solved the task. All, except cabin boy Kostya, who was too drunk to think straight. But what about you?

Note: in this task, we compare integers (x or k) as numbers (despite what representations they are written in), so $729 < 1999$ or $111 < 1000$.

Input

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

Next t lines contain test cases — one per test case. The one and only line of each test case contains the single integer n ($1 \le n \le 10^5$) — the length of the integer x you need to find.

It's guaranteed that the sum of n from all test cases doesn't exceed $2 \cdot 10^5$.

Output

For each test case, print the minimum integer x of length n such that obtained by Denis number r is maximum possible.

Example

input

```
2
1
3
```

output

```
8
998
```

Note

In the second test case (with $n = 3$), if uncle Bogdan had $x = 998$ then $k = 100110011000$. Denis (by wiping last $n = 3$ digits) will obtain $r = 100110011$.

It can be proved that the 100110011 is the maximum possible r Denis can obtain and 998 is the minimum x to obtain it.

C. Uncle Bogdan and Country Happiness

time limit per test: 2 seconds
memory limit per test: 256 megabytes
input: standard input
output: standard output

Uncle Bogdan is in captain Flint's crew for a long time and sometimes gets nostalgic for his homeland. Today he told you how his country introduced a happiness index.

There are n cities and $n - 1$ undirected roads connecting pairs of cities. Citizens of any city can reach any other city traveling by these roads. Cities are numbered from 1 to n and the city 1 is a capital. In other words, the country has a tree structure.

There are m citizens living in the country. A p_i people live in the i-th city but all of them are working in the capital. At evening all citizens return to their home cities using the shortest paths.

Every person has its own mood: somebody leaves his workplace in good mood but somebody are already in bad mood. Moreover any person can ruin his mood on the way to the hometown. **If person is in bad mood he won't improve it**.

Happiness detectors are installed in each city to monitor the happiness of **each** person who visits the city. The detector in the i-th city calculates a happiness index h_i as the number of people in good mood minus the number of people in bad mood. Let's say for the simplicity that *mood of a person doesn't change inside the city*.

Happiness detector is still in development, so there is a probability of a mistake in judging a person's happiness. One late evening, when all citizens successfully returned home, the government asked uncle Bogdan (the best programmer of the country) to check the correctness of the collected happiness indexes.

Uncle Bogdan successfully solved the problem. Can you do the same?

More formally, *You need to check: "Is it possible that, after all people return home, for each city i the happiness index will be equal exactly to h_i".*

Input

The first line contains a single integer t $(1 \le t \le 10000)$ — the number of test cases.

The first line of each test case contains two integers n and m $(1 \le n \le 10^5; 0 \le m \le 10^9)$ — the number of cities and citizens.

The second line of each test case contains n integers $p_1, p_2, \ldots, p_n$ $(0 \le p_i \le m; p_1 + p_2 + \ldots + p_n = m)$, where p_i is the number of people living in the i-th city.

The third line contains n integers $h_1, h_2, \ldots, h_n$ $(-10^9 \le h_i \le 10^9)$, where h_i is the calculated happiness index of the i-th city.

Next $n-1$ lines contain description of the roads, one per line. Each line contains two integers x_i and y_i $(1 \le x_i, y_i \le n; x_i \ne y_i)$, where x_i and y_i are cities connected by the i-th road.

It's guaranteed that the sum of n from all test cases doesn't exceed $2 \cdot 10^5$.

Output

For each test case, print YES, if the collected data is correct, or NO — otherwise. You can print characters in YES or NO in any case.

Examples

input

```
2
7 4
1 0 1 1 0 1 0
4 0 0 -1 0 -1 0
1 2
1 3
1 4
3 5
3 6
3 7
5 11
1 2 5 2 1
-11 -2 -6 -2 -1
1 2
1 3
1 4
3 5
```

output

```
YES
YES
```

input

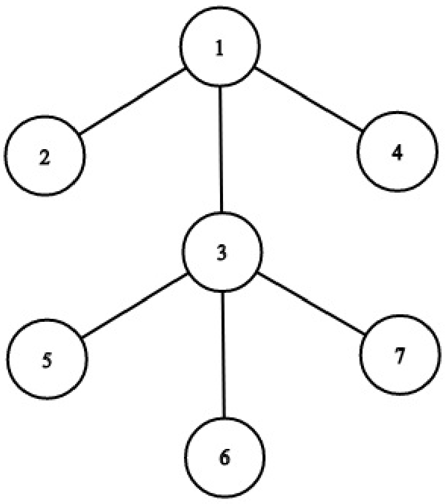
```
2
4 4
1 1 1 1
4 1 -3 -1
1 2
1 3
1 4
3 13
3 3 7
13 1 4
1 2
1 3
```

output

```
NO
NO
```

Note

Let's look at the first test case of the first sample:



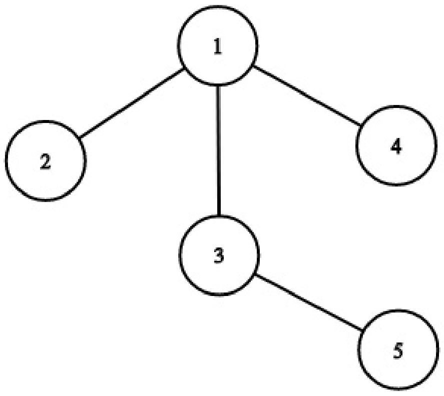
At first, all citizens are in the capital. Let's describe one of possible scenarios:

- a person from city 1: he lives in the capital and is in good mood;
- a person from city 4: he visited cities 1 and 4, his mood was ruined between cities 1 and 4;
- a person from city 3: he visited cities 1 and 3 in good mood;
- a person from city 6: he visited cities 1, 3 and 6, his mood was ruined between cities 1 and 3;

In total,

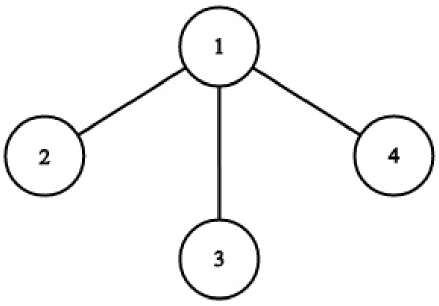
- $h_1 = 4 - 0 = 4$,
- $h_2 = 0$,
- $h_3 = 1 - 1 = 0$,
- $h_4 = 0 - 1 = -1$,
- $h_5 = 0$,
- $h_6 = 0 - 1 = -1$,
- $h_7 = 0$.

The second case of the first test:

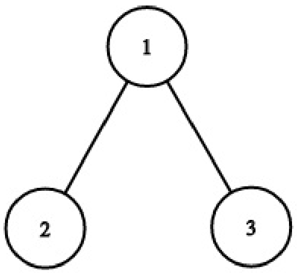


All people have already started in bad mood in the capital — this is the only possible scenario.

The first case of the second test:



The second case of the second test:



It can be proven that there is no way to achieve given happiness indexes in both cases of the second test.

D. Captain Flint and Treasure

time limit per test: 2 seconds
memory limit per test: 256 megabytes
input: standard input
output: standard output

Captain Fint is involved in another treasure hunt, but have found only one strange problem. The problem may be connected to the treasure's location or may not. That's why captain Flint decided to leave the solving the problem to his crew and offered an absurdly high reward: one day off. The problem itself sounds like this...

There are two arrays a and b of length n. Initially, an ans is equal to 0 and the following operation is defined:

1. Choose position i ($1 \le i \le n$);
2. Add a_i to ans;
3. If $b_i \neq -1$ then add a_i to a_{b_i}.

What is the maximum ans you can get by performing the operation on each i ($1 \le i \le n$) *exactly once*?

Uncle Bogdan is eager to get the reward, so he is asking your help to find the optimal order of positions to perform the operation on them.

Input

The first line contains the integer n ($1 \le n \le 2 \cdot 10^5$) — the length of arrays a and b.

The second line contains n integers $a_1, a_2, \ldots, a_n$ ($-10^6 \le a_i \le 10^6$).

The third line contains n integers $b_1, b_2, \ldots, b_n$ ($1 \le b_i \le n$ or $b_i = -1$).

Additional constraint: it's guaranteed that for any i ($1 \le i \le n$) the sequence $b_i, b_{b_i}, b_{b_{b_i}}, \ldots$ is not cyclic, in other words it will always end with -1.

Output

In the first line, print the maximum ans you can get.

In the second line, print the order of operations: n different integers $p_1, p_2, \ldots, p_n$ ($1 \le p_i \le n$). The p_i is the position which should be chosen at the i-th step. If there are multiple orders, print any of them.

Examples

input
```
3
1 2 3
2 3 -1
```
output
```
10
1 2 3
```

input
```
2
-1 100
2 -1
```
output
```
99
2 1
```

input
```
10
-10 -1 2 2 5 -2 -3 -4 2 -6
-1 -1 2 2 -1 5 5 7 7 9
```
output
```
-9
3 5 6 1 9 4 10 7 8 2
```

E. Uncle Bogdan and Projections

time limit per test: 4 seconds
memory limit per test: 512 megabytes
input: standard input
output: standard output

After returning to shore, uncle Bogdan usually visits the computer club "The Rock", to solve tasks in a pleasant company. One day, uncle Bogdan met his good old friend who told him one unusual task...

There are n non-intersecting horizontal segments with ends in integers points on the plane with the standard cartesian coordinate system. All segments are strictly above the OX axis. You can choose an arbitrary vector (a, b), where $b < 0$ and coordinates are real numbers, and project all segments to OX axis along this vector. *The projections shouldn't intersect but may touch each other.*

Find the minimum possible difference between x coordinate of the right end of the rightmost projection and x coordinate of the left end of the leftmost projection.

Input

The first line contains the single integer n $(1 \le n \le 2000)$ — the number of segments.

The i-th of the next n lines contains three integers xl_i, xr_i and y_i $(-10^6 \le xl_i < xr_i \le 10^6; 1 \le y_i \le 10^6)$ — coordinates of the corresponding segment.

It's guaranteed that the segments don't intersect or touch.

Output

Print the minimum possible difference you can get.

Your answer will be considered correct if its absolute or relative error doesn't exceed 10^{-6}.

Formally, if your answer is a and jury's answer is b then your answer will be considered correct if $\frac{|a-b|}{\max(1,|b|)} \le 10^{-6}$.

Examples

input

```
3
1 6 2
4 6 4
4 6 6
```

output

```
9.000000000
```

input

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

output

```
6.333333333
```

input

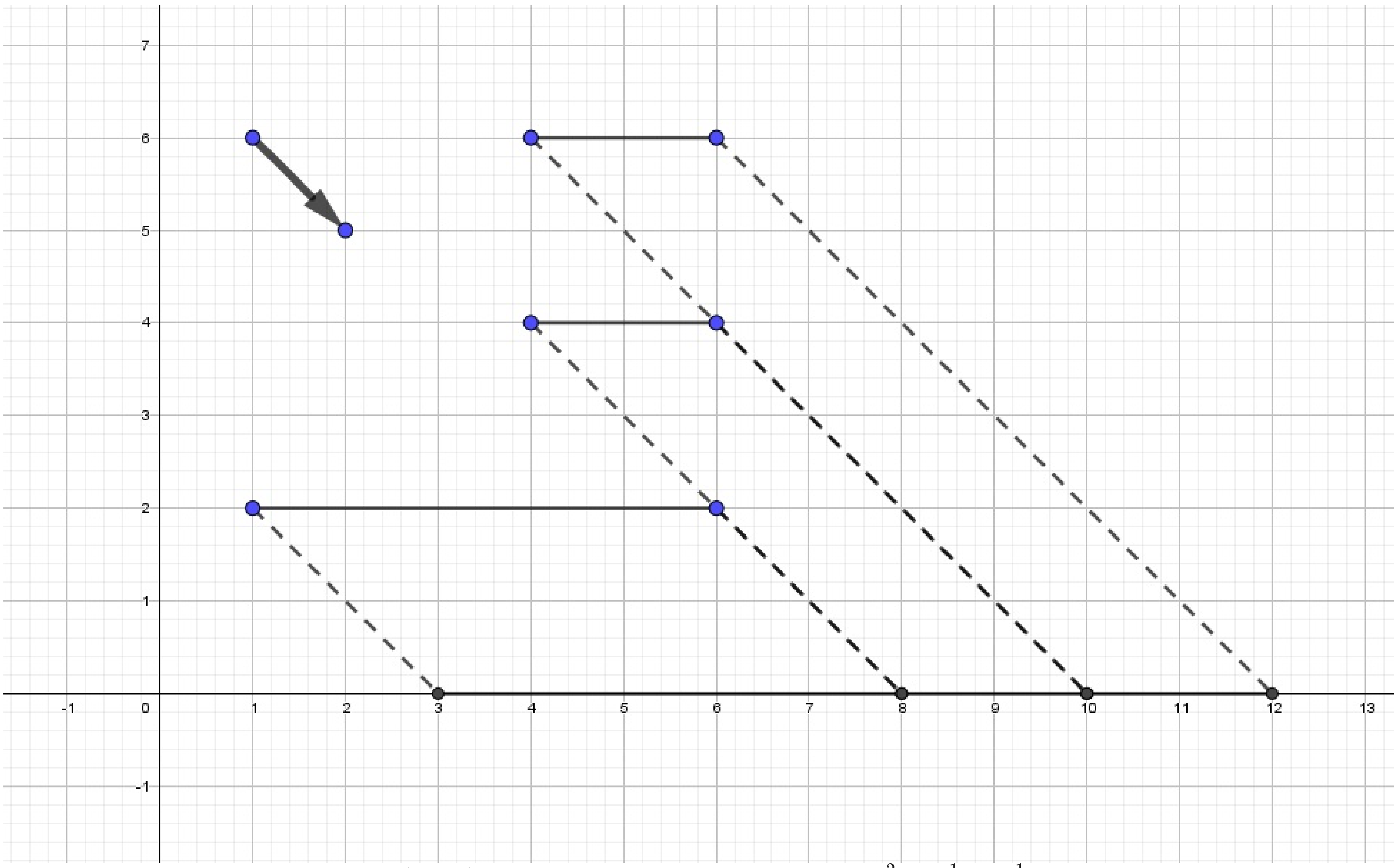
```
2
1 3 1
4 7 1
```

output

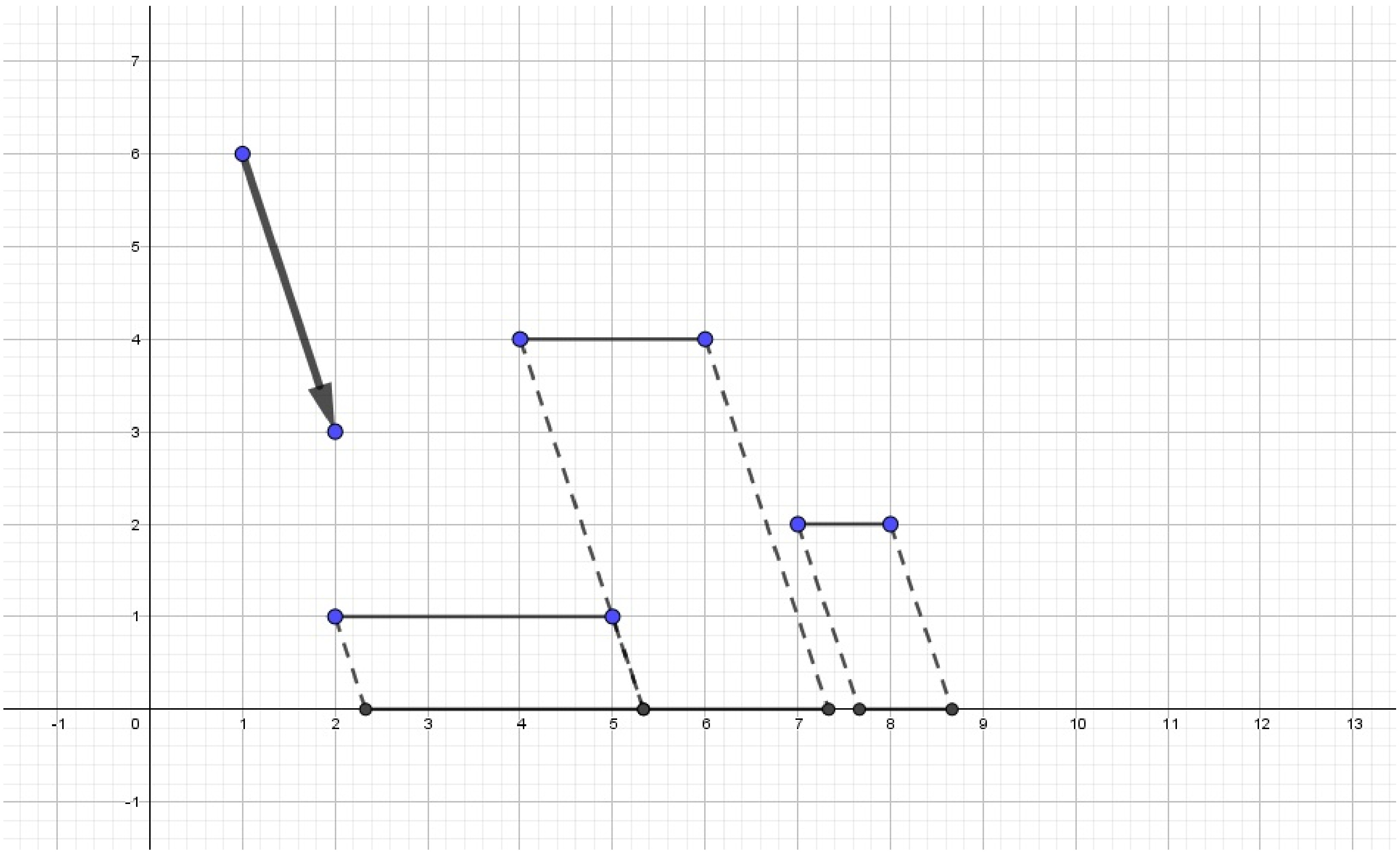
```
6.000000000
```

Note

In the first example if we project segments along the vector $(1, -1)$ then we get an answer $12 - 3 = 9$ and (it can be proven) it is impossible to get less.



It is optimal to project along the vector $(1, -3)$ in the second example. The answer is $8\frac{2}{3} - 2\frac{1}{3} = 6\frac{1}{3}$


7
6
5
4
3
2
1
0
-1
-1
1
2
3
4
5
6
7
8
9
10
11
12
13