Problem A. For Loop

OS Linux

A *for* loop is a programming language statement which allows code to be repeatedly executed.

The syntax is

```
for ( <expression_1> ; <expression_2> ; <expression_3> )
    <statement>
```

- *expression_1* is used for intializing variables which are generally used for controlling the terminating flag for the loop.
- *expression_2* is used to check for the terminating condition. If this evaluates to false, then the loop is terminated.
- *expression_3* is generally used to update the flags/variables.

A sample loop is

```
for(int i = 0; i < 10; i++) {
    ...
}
```

In this challenge, you will use a for loop to increment a variable through a range.

Input Format

You will be given two positive integers, a and b ($a \leq b$), separated by a newline.

Output Format

For each integer n in the inclusive interval $[a, b]$:

- If $1 \leq n \leq 9$, then print the English representation of it in lowercase. That is "one" for 1, "two" for 2, and so on.
- Else if $n > 9$ and it is an even number, then print "even".
- Else if $n > 9$ and it is an odd number, then print "odd".

Note: $[a, b] = \{x \in \mathbb{Z} |\ a \leq x \leq b\} = \{a,\ a+1, \ldots, b\}$

Input	Output
8 11	eight nine even odd

Problem B. Division?

Time limit 1000 ms
Mem limit 262144 kB

Problem C. Grab the Candies

Time limit 1000 ms
Mem limit 262144 kB

Mihai and Bianca are playing with bags of candies. They have a row a of n bags of candies. The i-th bag has a_i candies. The bags are given to the players in the order from the first bag to the n-th bag.

If a bag has an even number of candies, Mihai grabs the bag. Otherwise, Bianca grabs the bag. Once a bag is grabbed, the number of candies in it gets added to the total number of candies of the player that took it.

Mihai wants to show off, so he wants to reorder the array so that at any moment (except at the start when they both have no candies), Mihai will have **strictly more** candies than Bianca. Help Mihai find out if such a reordering exists.

Input

The first line of the input contains an integer t ($1 \leq t \leq 1000$) — the number of test cases.

The first line of each test case contains a single integer n ($1 \leq n \leq 100$) — the number of bags in the array.

The second line of each test case contains n space-separated integers a_i ($1 \leq a_i \leq 100$) — the number of candies in each bag.

Output

For each test case, output "`YES`" (without quotes) if such a reordering exists, and "`NO`" (without quotes) otherwise.

You can output the answer in any case (for example, the strings "`yEs`", "`yes`", "`Yes`" and "`YES`" will be recognized as a positive answer).

Sample 1

Input	Output
3 4 1 2 3 4 4 1 1 1 2 3 1 4 3	YES NO NO

Note

In the first test case, Mihai can reorder the array as follows: $[4, 1, 2, 3]$. Then the process proceeds as follows:

- the first bag has 4 candies, which is even, so Mihai takes it — Mihai has 4 candies, and Bianca has 0.
- the second bag has 1 candies, which is odd, so Bianca takes it — Mihai has 4 candies, and Bianca has 1.
- the third bag has 2 candies, which is even, so Mihai takes it — Mihai has 6 candies, and Bianca has 1.
- the fourth bag has 3 candies, which is odd, so Bianca takes it — Mihai has 6 candies, and Bianca has 4.

Since Mihai always has more candies than Bianca, this reordering works.

Problem D. Odd Set

Time limit 1000 ms
Mem limit 262144 kB

You are given a multiset (i. e. a set that can contain multiple equal integers) containing $2n$ integers. Determine if you can split it into exactly n pairs (i. e. each element should be in exactly one pair) so that the sum of the two elements in each pair is **odd** (i. e. when divided by 2, the remainder is 1).

Input

The input consists of multiple test cases. The first line contains an 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 n ($1 \le n \le 100$).

The second line of each test case contains $2n$ integers $a_1, a_2, \ldots, a_{2n}$ ($0 \le a_i \le 100$) — the numbers in the set.

Output

For each test case, print "`Yes`" if it can be split into exactly n pairs so that the sum of the two elements in each pair is **odd**, and "`No`" otherwise. You can print each letter in any case.

Sample 1

Input	Output
5 2 2 3 4 5 3 2 3 4 5 5 5 1 2 4 1 2 3 4 1 5 3 2 6 7 3 4	Yes No No Yes No

Note

In the first test case, a possible way of splitting the set is $(2, 3)$, $(4, 5)$.

In the second, third and fifth test case, we can prove that there isn't any possible way.

In the fourth test case, a possible way of splitting the set is $(2, 3)$.

Problem E. Good Kid

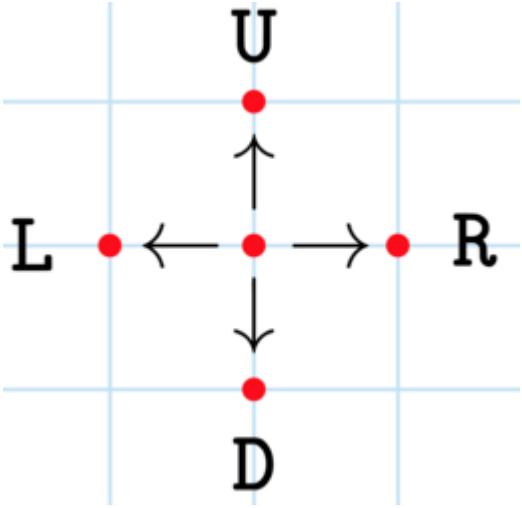
Time limit 1000 ms
Mem limit 262144 kB

Problem F. Following Directions

Time limit 1000 ms
Mem limit 262144 kB

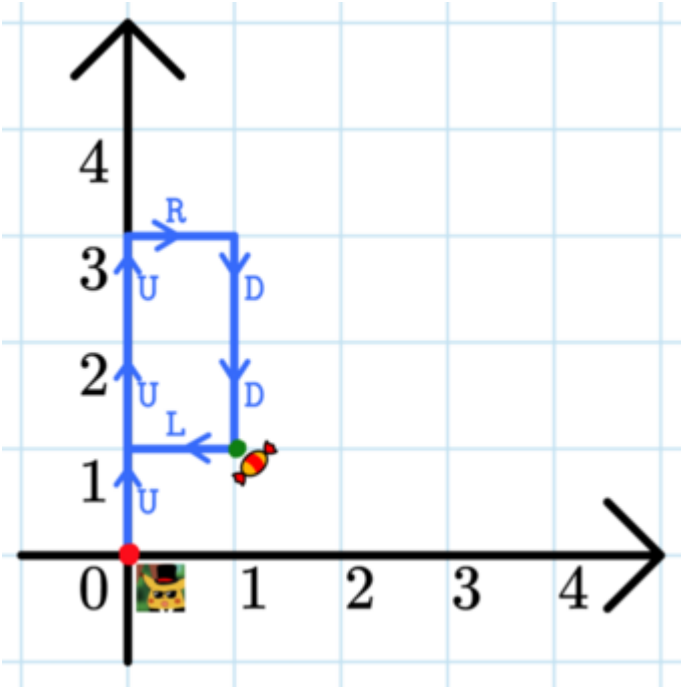
Alperen is standing at the point $(0, 0)$. He is given a string s of length n and performs n moves. The i-th move is as follows:

- if $s_i = \texttt{L}$, then move one unit left;
- if $s_i = \texttt{R}$, then move one unit right;
- if $s_i = \texttt{U}$, then move one unit up;
- if $s_i = \texttt{D}$, then move one unit down.



If Alperen starts at the center point, he can make the four moves shown.

There is a candy at $(1, 1)$ (that is, one unit above and one unit to the right of Alperen's starting point). You need to determine if Alperen ever passes the candy.



Alperen's path in the first test case.

Input

The first line of the input contains an integer t ($1 \leq t \leq 1000$) — the number of testcases.

The first line of each test case contains an integer n ($1 \leq n \leq 50$) — the length of the string.

The second line of each test case contains a string s of length n consisting of characters `L`, `R`, `D`, and `U`, denoting the moves Alperen makes.

Output

For each test case, output "`YES`" (without quotes) if Alperen passes the candy, and "`NO`" (without quotes) otherwise.

You can output the answer in any case (for example, the strings "`yEs`", "`yes`", "`Yes`" and "`YES`" will be recognized as a positive answer).

Sample 1

Input	Output
7 7 UUURDDL 2 UR 8 RRRUUDDD 3 LLL 4 DUUR 5 RUDLL 11 LLLLDDRUDRD	YES YES NO NO YES YES NO

Note

In the first test case, Alperen follows the path

$$(0,0) \overset{\texttt{U}}{\to} (0,1) \overset{\texttt{U}}{\to} (0,2) \overset{\texttt{U}}{\to} (0,3) \overset{\texttt{R}}{\to} (1,3) \overset{\texttt{D}}{\to} (1,2) \overset{\texttt{D}}{\to} \mathbf{(1,1)} \overset{\texttt{L}}{\to} (0,1).$$

Note that Alperen doesn't need to end at the candy's location of $(1,1)$, he just needs to pass it at some point.

In the second test case, Alperen follows the path

$$(0,0) \overset{\texttt{U}}{\to} (0,1) \overset{\texttt{R}}{\to} \mathbf{(1,1)}.$$

In the third test case, Alperen follows the path

$$(0,0) \overset{\texttt{R}}{\to} (1,0) \overset{\texttt{R}}{\to} (2,0) \overset{\texttt{R}}{\to} (3,0) \overset{\texttt{U}}{\to} (3,1) \overset{\texttt{U}}{\to} (3,2) \overset{\texttt{D}}{\to} (3,1) \overset{\texttt{D}}{\to} (3,0) \overset{\texttt{D}}{\to} (3,-1).$$

In the fourth test case, Alperen follows the path

$$(0,0) \overset{\texttt{L}}{\to} (-1,0) \overset{\texttt{L}}{\to} (-2,0) \overset{\texttt{L}}{\to} (-3,0).$$

Problem G. Target Practice

Time limit 1000 ms
Mem limit 262144 kB

Problem H. QAQ

Time limit 1000 ms
Mem limit 262144 kB

"QAQ" is a word to denote an expression of crying. Imagine "Q" as eyes with tears and "A" as a mouth.

Now Diamond has given Bort a string consisting of only uppercase English letters of length n. There is a great number of "QAQ" in the string (Diamond is so cute!).

illustration by 猫屋 https://twitter.com/nekoyaliu

Bort wants to know how many subsequences "QAQ" are in the string Diamond has given. Note that the letters "QAQ" don't have to be consecutive, but the order of letters should be exact.

Input

The only line contains a string of length n ($1 \le n \le 100$). It's guaranteed that the string only contains uppercase English letters.

Output

Print a single integer — the number of subsequences "QAQ" in the string.

Sample 1

Input	Output
QAQAQYSYIOIWIN	4

Sample 2

Input	Output
QAQQQZZYNOIWIN	3

Note

In the first example there are 4 subsequences "QAQ": "**QAQ**AQYSYIOIWIN", "**QA**QA**Q**YSYIOIWIN", "**Q**AQ**AQ**YSYIOIWIN", "QA**QAQ**YSYIOIWIN".

Problem I. Taxi

Time limit 3000 ms
Mem limit 262144 kB

After the lessons n groups of schoolchildren went outside and decided to visit Polycarpus to celebrate his birthday. We know that the i-th group consists of s_i friends ($1 \le s_i \le 4$), and they want to go to Polycarpus together. They decided to get there by taxi. Each car can carry at most four passengers. What minimum number of cars will the children need if all members of each group should ride in the same taxi (but one taxi can take more than one group)?

Input

The first line contains integer n ($1 \le n \le 10^5$) — the number of groups of schoolchildren. The second line contains a sequence of integers $s_1, s_2, ..., s_n$ ($1 \le s_i \le 4$). The integers are separated by a space, s_i is the number of children in the i-th group.

Output

Print the single number — the minimum number of taxis necessary to drive all children to Polycarpus.

Sample 1

Input	Output
5 1 2 4 3 3	4

Sample 2

Input	Output
8 2 3 4 4 2 1 3 1	5

Note

In the first test we can sort the children into four cars like this:

- the third group (consisting of four children),
- the fourth group (consisting of three children),
- the fifth group (consisting of three children),
- the first and the second group (consisting of one and two children, correspondingly).

There are other ways to sort the groups into four cars.

Problem J. Chips on the Board

Time limit 2000 ms
Mem limit 262144 kB

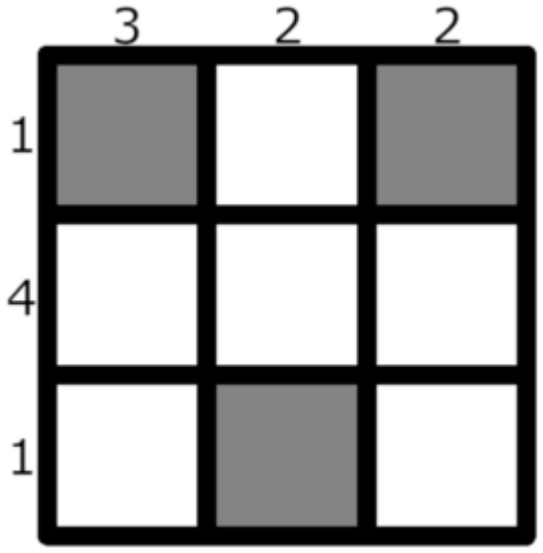
You are given a board of size $n \times n$ (n rows and n colums) and two arrays of positive integers a and b of size n.

Your task is to place the chips on this board so that the following condition is satisfied for every cell (i, j):

- there exists at least one chip in the same column or in the same row as the cell (i, j). I. e. there exists a cell (x, y) such that there is a chip in that cell, and either $x = i$ or $y = j$ (or both).

The cost of putting a chip in the cell (i, j) is equal to $a_i + b_j$.

For example, for $n = 3$, $a = [1, 4, 1]$ and $b = [3, 2, 2]$. One of the possible chip placements is as follows:



White squares are empty

The total cost of that placement is $(1 + 3) + (1 + 2) + (1 + 2) = 10$.

Calculate the minimum possible total cost of putting chips according to the rules above.

Input

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

The first line of each test case contains a single integer n ($1 \le n \le 3 \cdot 10^5$).

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

The third line contains n integers $b_1, b_2, \ldots, b_n$ ($1 \le b_i \le 10^9$).

The sum of n over all test cases doesn't exceed $3 \cdot 10^5$.

Output

For each test case, print a single integer — the minimum possible total cost of putting chips according to the rules.

Sample 1

Input	Output
4 3 1 4 1 3 2 2 1 4 5 2 4 5 2 3 5 5 2 4 5 3 3 4 2 1 5	10 9 13 24

Note

The first test case of the example is described in the statement.

Problem K. Adding Reversed Numbers

Time limit	5000 ms
Mem limit	1572864 kB
Code length Limit	50000 B
OS	Linux

The Antique Comedians of Malidinesia prefer comedies to tragedies. Unfortunately, most of the ancient plays are tragedies. Therefore the dramatic advisor of ACM has decided to transfigure some tragedies into comedies. Obviously, this work is very hard because the basic sense of the play must be kept intact, although all the things change to their opposites. For example the numbers: if any number appears in the tragedy, it must be converted to its reversed form before being accepted into the comedy play.

Reversed number is a number written in Arabic numerals but the order of digits is reversed. The first digit becomes last and vice versa. For example, if the main hero had 1245 strawberries in the tragedy, he has 5421 of them now. Note that all the leading zeros are omitted. That means if the number ends with a zero, the zero is lost by reversing (e.g. 1200 gives 21). Also note that the reversed number never has any trailing zeros.

ACM needs to calculate with reversed numbers. Your task is to add two reversed numbers and output their reversed sum. Of course, the result is not unique because any particular number is a reversed form of several numbers (e.g. 21 could be 12, 120 or 1200 before reversing). Thus we must assume that no zeros were lost by reversing (e.g. assume that the original number was 12).

Input

The input consists of *N* cases (equal to about 10000). The first line of the input contains only positive integer *N*. Then follow the cases. Each case consists of exactly one line with two positive integers separated by space. These are the reversed numbers you are to add.

Output

For each case, print exactly one line containing only one integer - the reversed sum of two reversed numbers. Omit any leading zeros in the output.

Example

```
Sample input:
3
24 1
4358 754
```

```
305 794

Sample output:
34
1998
1
```

Problem L. Repetitions

Time limit 1000 ms
Mem limit 524288 kB

Problem M. Permutations

Time limit 1000 ms
Mem limit 524288 kB

A permutation of integers $1, 2, \ldots, n$ is called *beautiful* if there are no adjacent elements whose difference is 1.

Given n, construct a beautiful permutation if such a permutation exists.

Input

The only input line contains an integer n.

Output

Print a beautiful permutation of integers $1, 2, \ldots, n$. If there are several solutions, you may print any of them. If there are no solutions, print "NO SOLUTION".

Constraints

- $1 \leq n \leq 10^6$

Example 1

Input	Output
5	4 2 5 3 1

Example 2

Input	Output
3	NO SOLUTION

Problem N. Make A and B equal

Time limit	1000 ms
Code length Limit	50000 B
OS	Linux

Chef is given two arrays A and B of length N each.

In one operation Chef can choose one element of A and one element of B and increase them by 1.

More formally:
Chef can pick two integers i, j $(1 \leq i, j \leq N)$ and increment A_i and B_j by 1.

Determine the **minimum** number of operations required to make A and B equal.

Output -1 if it is not possible to make A and B equal.

Input Format

- The first line of input will contain a single integer T, denoting the number of test cases.
- Each test case consists of multiple lines of input.
  - The first line of each test case contains a single integer N - denoting the length of arrays A and B.
  - The second line of each test case contains N space separated integers $A_1, A_2, A_3, \ldots A_N$ - denoting the array A.
  - The third line of each test case contains N space separated integers $B_1, B_2, B_3, \ldots B_N$ - denoting the array B.

Output Format

For each test case, output the **minimum** number of operations to make A and B equal or -1 if they cannot be made equal.

Constraints

- $1 \leq T \leq 2 \cdot 10^4$
- $2 \leq N \leq 10^5$
- $1 \leq A_i, B_i \leq 10^9$
- Sum of N over all test cases do not exceed 10^5.

Sample 1

Input	Output
3 2 1 2 2 1 3 1 1 2 2 2 1 3 4 6 8 5 7 6	1 -1 2

Test case 1: We can choose $i = 1$ and $j = 2$ and increment A_i and B_j by 1. Thus, both arrays become $[2, 2]$ and are equal. We require only 1 operation to make these arrays equal. It can be proven that the arrays cannot be made equal in less than one operation.

Test case 2: Both the arrays cannot be made equal using any number of operations.

Test case 3: We perform 2 operations as follows:

- Choose $i = 1, j = 3$: The arrays become $A = [5, 6, 8]$ and $B = [5, 7, 7]$.
- Choose $i = 2, j = 3$: The arrays become $A = [5, 7, 8]$ and $B = [5, 7, 8]$.

Thus, both arrays can be made equal using 2 operations.

Problem O. Mean and Median

Time limit	500 ms
Code length Limit	50000 B
OS	Linux

Chef has two numbers X and Y. Chef wants to find three integers A, B, and C such that:

- $-1000 \leq A, B, C \leq 1000$
- $mean([A, B, C]) = X$
- $median([A, B, C]) = Y$

Can you help Chef find such three integers?

As a reminder, $mean([P, Q, R]) = \frac{P+Q+R}{3}$ and $median([P, Q, R])$ is the element at the 2^{nd} (middle) position after we sort $[P, Q, R]$ in non-decreasing order.

Input Format

- The first line contains a single integer T — the number of test cases. Then the test cases follow.
- The first and only line of each test case contains two space-separated integers X and Y — the required mean and median of the three integers.

Output Format

For each test case, output three integers A, B, C which satisfy the given conditions.

It is guaranteed that an answer always exists under the given constraints.

If multiple answers exist, output any.

Constraints

- $1 \leq T \leq 10^5$
- $-100 \leq X, Y \leq 100$

Sample 1

Input	Output
3 5 5 67 100 4 5	5 5 5 0 100 101 0 5 7

Test Case 1: $mean([5, 5, 5]) = \frac{5+5+5}{3} = 5, median([5, 5, 5]) = 5.$

Test Case 2: $mean([0, 100, 101]) = \frac{0+100+101}{3} = \frac{201}{3} = 67, median([0, 100, 101]) = 100.$

Test Case 3: $mean([0, 5, 7]) = \frac{0+5+7}{3} = 4, median([0, 5, 7]) = 5.$