Problem Setting (`problemsetting`)

This problem is loosely based on real events.

The organizing committee of an unspecified programming contest (UPC) is planning to hold many rounds this year, each round consisting of N problems of varying difficulties. The problems in a round are numbered from 1 to N.

As such, there have been a lot of problem proposals. Each problem proposal has a difficulty rating, which is either

- an integer i $(1 \leq i \leq N)$ – meaning that this problem is appropriate as problem i in a round; or
- a value in the form $i.5$ for some integer i $(1 \leq i < N)$ – meaning that this problem is appropriate as either problem i or problem $i + 1$ in a round.

Figure 1: Creating a balanced round is more difficult than you'd think.

Given the value of N and the number of problem proposals for each difficulty rating, find the maximum number of rounds that can be created using these problems. Each problem can only be used in at most one round.

> ☞ Among the attachments of this task you may find a template file `problemsetting.*` with a sample incomplete implementation.

Input

Each test contains multiple test cases. The first line of input contains a single integer T, the number of test cases.

The first line of each test case contains the integer N, the number of problems in an UPC round.

The second line of each test case contains N integers $A_1, A_2, \ldots, A_N$, where A_i denotes the number of problem proposals with a difficulty rating of i.

The third line of each test case contains $N - 1$ integers $B_1, B_2, \ldots, B_{N-1}$, where B_i denotes the number of problem proposals with a difficulty rating of $i.5$.

Output

For each test case, print a single integer, the maximum number of rounds that can be created from the available problems.

Constraints

- $1 \le T \le 10\,000$.

- $2 \le N \le 200\,000$.

- $0 \le A_i \le 1\,000\,000\,000$ for each $i = 1 \ldots N$.

- $0 \le B_i \le 1\,000\,000\,000$ for each $i = 1 \ldots N - 1$.

- The sum of N across all test cases does not exceed $200\,000$.

Scoring

Your program will be tested against several test cases grouped in subtasks. In order to obtain the score of a subtask, your program needs to correctly solve all of its test cases.

– **Subtask 1** (0 points)   Examples.

– **Subtask 2** (5 points) $B_i = 0$ for each $i = 1 \ldots N - 1$.

– **Subtask 3** (10 points) $A_i \le 1$ for each $i = 1 \ldots N$; $B_i \le 1$ for each $i = 1 \ldots N - 1$.

– **Subtask 4** (30 points) $A_i \le 100$ for each $i = 1 \ldots N$; $B_i \le 100$ for each $i = 1 \ldots N - 1$.

– **Subtask 5** (20 points) $A_i = 0$ for each $i = 1 \ldots N$.

– **Subtask 6** (35 points)   No additional limitations.

Examples

input	output
5 6 0 1 1 0 1 1 1 0 1 0 1 5 0 1 1 1 0 1 0 1 0 2 1 1 2 6 0 0 0 0 0 0 5 2 5 100 9 7 1 3 0 4 1 0 2 4 3 1 3 1 5	1 0 2 3 4

Explanation

In the first test case, there are 7 problem proposals, which have difficulty ratings of 1.5, 2, 3, 3.5, 5, 5.5 and 6, respectively.

We can create a single round using these problems:

- The problem with a difficulty rating of 1.5 will be used as the first problem in the round.

- The problem with a difficulty rating of 2 will be used as the second problem in the round.

- The problem with a difficulty rating of 3 will be used as the third problem in the round.

- The problem with a difficulty rating of 3.5 will be used as the fourth problem in the round.

- The problem with a difficulty rating of 5.5 will be used as the fifth problem in the round.

- The problem with a difficulty rating of 6 will be used as the sixth problem in the round.

In the second test case, we can show that it is impossible to create a complete round from the given proposals.

In the third test case, it is possible to create the following rounds from the given proposals:

- $[1, 1.5]$

- $[1.5, 2]$

In the fourth test case, it is possible to create the following rounds from the given proposals:

- $[1.5, 1.5, 3.5, 3.5, 5.5, 5.5]$

- $[1.5, 1.5, 3.5, 3.5, 5.5, 5.5]$

- $[1.5, 2.5, 2.5, 3.5, 5.5, 5.5]$