Novak and Nikola

As You know, the best tennis player in the world is a famous Serbian sportsman, Mr Novak Djokovic. Also, the famous Serbian basketball player, Nikola Jokić has been named this year's NBA MVP. Novak Djokovic and Nikola Jokic are chatting on ViberSport about the approaching Olympiad and dates.

Novak Djokovic is called a joker. Guess what - Jokic has been also called The Joker. Novak on the other hand is very fond of informatics and decides to give his friend an interesting numeric joke: „Nikola, I'm writing to you 2 number sequences (8 digits per each). Each of those two sequences represents a valid date – the day is written by 2 digits, the month also by 2 digits and the year is written by 4 digits. The order of the day, the month and the year is unknown. That means both dates may be in one of the following formats DDMMYYYY, MMDDYYYY, DDYYYYMM, MMYYYYDD, YYYYDDMM or GGGGMMDD, note that the dates may be in different formats each. I'll also write down the number of the days between the dates. Knowing the number of the days of the period and the fact that both dates are from the same year you must find both dates. To make your task easier I'll tell you that there is only one solution to the task."

Please, help Nikola Jokic solving the task. Write a program, which given two groups of eight digits representing the scrambled dates and the number of the days between the two dates, determines and prints out both dates in format DD-MM-YYYY.

Input

On the first row of the standard input, a sequence of eight digits is given – the first "scrambled" date (it may be the beginning or the ending date of the period).

On the second row, another eight digits are given – the second "scrambled" date (again it may represent the beginning or the ending date of the period).

A positive integer is given on the third row – the count of the days between the two dates.

Both dates are counted in the final result (total number of days). For each test case there is exactly one solution ($1 \leq numberofdaysintheperiod \leq 350;$**).**

Output

Your program should print two dates in format DD-MM-YYYY: on the first row – the beginning date of the period, and on the second row – the ending date of the period.

Example 1

Input

03282021
20211205
46

Output

28-03-2021
12-05-2021

Example 2

Input

11111111
11111112
31

Output

11-11-1111
11-12-1111

Solution

There are no special tricks in this task, but we need to combine several subroutines in one program: obtaining the possible valid dates from each of the two sequences of 8 digits, combining each valid date obtained from the first sequence with each valid a date obtained from the second sequence and a calculation of the number of days between those dates, taking into account the possible leap year.

The main thing in solving the problem is to structure the solution correctly and to put some procedures (validation of a date, calculation of the number of days between dates) into separate modules.

It is pragmatic to arrange the possible dates for one end of the period in one array, and the possible dates for the other end - in another array. This will make it easier to test and exhaust all possibilities.

We must not forget the condition that for each test case there is a solution and it is the only one - this allows you to display the first solution that is found and stop the program. To check the validity of a date and find the number of days between two dates, it is best to use an array in which the element with index i ($1 \le i \le 12$) contains the number of days in the respective month. Especially for February, a check should be made whether the year is leap year and 28 or 29 should be written in the element of the array. The criterion for a leap year is:

Every year that is exactly divisible by four is a leap year, except for years that are exactly divisible by 100, but these centurial years are leap years if they are exactly divisible by 400. For example, the years 1700, 1800, and 1900 are not leap years, but the years 1600 and 2000 are.

Lovely Ada Lovelace

Augusta Ada Byron, Countess of Lovelace (1815 – 1852) is often regarded as the first computer programmer.
Lady Ada has already played all possible board games with two or more players, so from now on she plays alone.
She created an array of N non-negative integers.
Then, with each move (if possible), she divides the current array into two non-empty subarrays - left and right, so that the left and right subarrays have the same sum of elements. As result, she wins 1 point.
Then she chooses which of the two subarrays to remove and the new array remains with the numbers from the other subarray.
However, it is difficult for her to find the optimal strategy so that she wins the most points. Therefore, write a program that prints the maximum number of points scored for given T arrays, if the game is started with each of the given arrays.

Input

From the first row of standard input, read an integer T - the number of arrays.
There are two rows entered for each game, with the first row being the integer N - the size of the corresponding array, and the second row - the numbers in the array with which a certain game starts. ($1 \le N < 2 \times 10^4$, $1 \le T \le 100$, elements of the array are integers in the interval $[0;10^9]$)

Output

In the T line, print an integer - the maximum number of points that Ada can score, starting the game with the appropriate array.

Example

Input

```
2
7
3 1 4 0 3 2 3
5
7 7 7 7 7
```

Output

```
2
0
```

Clarification

If we start the game with the first array 3, 1, 4, 0, 3, 2, 3; then at the beginning the optimal strategy is to divide the array into 3, 1, 4 and 0, 3, 2, 3; to choose 3, 1, 4, to divide the array again into 3, 1 and 4, and now, no matter which subarray we choose, we have no more moves. So, there are 2 points (total sum). It is easy to see that for the second array we have no shifts from the beginning.

Solution

We will form prefix sum for the given array. Then we will use binary search over the possible answers for the positions between 0 and n-1 (included 0 and n-1). Actually we will use binary search to guess the answer of the problem. By guessing means that we will get several candidates for maximum points and "too high", "too low" responses, Finally we will find solution by comparation results for left and right subarrays.