Problem S: Solitaire

Maggie has a special deck of cards. There are $4n$ cards in the deck: For each i between 1 and n, inclusive, there are four cards with the value i – the i of spades, the i of hearts, the i of diamonds, and the i of clubs. Maggie uses this deck of cards to play a simple solitaire game.

At the beginning of the game the deck of cards is placed face down on the table. There are four empty slots – one for each suit. Additionally, there is an empty discard pile next to the deck. The goal of the game is to place all cards of each suit onto the corresponding slot. The cards have to be placed onto their slot in ascending order. A more precise description of the game follows.

During the game, Maggie repeats the following steps:

1. Pick up the topmost card of the deck.
2. Look at its suit and look at the slot for the corresponding suit.
3. If the card is an ace (i.e., its value is 1), place it onto the corresponding empty slot.
4. If the value of her card is $v > 1$ and the topmost card on the corresponding slot has value $v - 1$, place the card with value v on top of the card with value $v - 1$. For example, the 8 of spades can be placed onto the 7 of spades.
5. If you were unable to place the current card onto its slot, place it face up onto the discard pile. For example, if your current card is the 8 of spades and the top card on the corresponding slot is the 3 of spades, the 8 of spades goes onto the discard pile.
6. If there are no cards left in your deck:

   If there are no cards left in your discard pile (i.e., all $4n$ cards have been placed onto their slots), the game ends.

   Otherwise, take the discard pile and flip it upside down to produce a new deck. Note that the order of cards in the new deck is the same as the relative order of these cards in the original deck.

Problem specification

Obviously, Maggie will always win the game eventually, but sometimes it can take quite long.

You are given n and the initial order of cards in the deck. Compute how many times Maggie went through her deck of cards before winning the game. In other words, the answer you should compute is one plus the number of times she flipped the discard pile to get a new deck.

Input specification

The first line of the input file contains an integer t specifying the number of test cases. Each test case is preceded by a blank line.

Each test case consists of two lines. The first line contains the number n. The second line contains a list of $4n$ different cards – the initial deck from top to bottom. Each card is given in the format `Xy`, where `X` is one of the four suits and `y` is the value. E.g., `H47` is the 47 of hearts.

In the **easy subproblem S1** we have $t = 20$ and $n = 13$ in each test case. (Maggie is playing with the standard deck of 52 cards.)

In the **hard subproblem S2** we have $t = 12$ and $1 \leq n \leq 150,000$ in each test case. Because the input file size for subproblem S2 is about 30 MB, you cannot download it directly. Instead, we have provided a small Python 2 program that will generate the file `s2.in` when executed.

Output specification

For each test case output a single line with a single integer: the number of times Maggie has to go through her deck in order to win the game.

Note

In the real contest some large inputs will be provided in the same way as the input `s2.in` in this practice problem. Please make sure you are able to generate it.

Example

input	output
1 3 S2 H3 S1 D3 D1 H1 D2 C1 S3 H2 C2 C3	2

After going through her deck once, the topmost cards on the four slots will be the 1 of spades, the 2 of hearts, the 2 of diamonds, and the 3 of clubs. The remaining cards will now be in the discard pile. Afterwards, Maggie will flip the discard pile over and she will go through the remaining cards again (starting again with the 2 of spades). During this second pass Maggie will be able to place all the remaining cards.

Problem T: Town

You are standing in a town with infinitely many houses. Currently, the houses do not have any house numbers. You were given the task to fix this.

You have a box with plastic digits. For each i between 0 and 9, inclusive, there are d_i copies of the digit i in your box. You can number a house by sticking the appropriate digits to its wall. For example, on the house number 474 you will use two digits 4 and one digit 7.

You have decided that you will number the houses sequentially, starting from 1. How many houses can you number before you run out of digits?

Problem specification

You are given the counts $d_0, \ldots, d_9$ of the digits in your box. Find the largest x such that you are able to write the numbers 1 through x using your set of digits.

Input specification

The first line of the input file contains an integer t specifying the number of test cases. Each test case is preceded by a blank line. Each test case consists of a single line containing 10 nonnegative integers – the counts of digits 0 through 9.

In the **easy subproblem T1** the sum of all d_i will be at most 10^7.

In the **hard subproblem T2** the sum of all d_i will be at most 10^{16}.

Output specification

For each test case, output a single line with the answer to the test case.

Example

input	output
1 1 3 1 1 2 1 1 2 1 1	10

With the digits you have, you are able to build the numbers 1 through 10. Once you do so, you will be left with three digits: one 1, one 4, and one 7. This is not enough to construct the number 11.

Problem U: Unusual Game Show

You may have already heard about the famous game show host Monty Hall. Back in the day, his game show had confused many a bright mathematician.

This is how it all looked like: The contestant was shown three doors, labeled 1 through 3. There was a prize (e.g., a new car) behind one of the doors, and a goat behind each of the other two doors. The door hiding the prize was chosen uniformly at random. Monty knew which door contains the prize. The game consisted of three steps:

1. At the beginning of the game, the contestant was asked to choose one of the three doors for herself.
2. Once the choice was made, Monty would open one of the doors the contestant did not choose.

   Of course, Monty would never open the door with the prize. If the contestant chose the door with the prize, Monty would open either of the other two doors, chosen uniformly at random. In all other cases Monty would open the only door that was neither chosen by the contestant nor hiding the prize.
3. Then, Monty asked the contestant a very tricky question: "Do you want to *keep* the door you have, or do you want to *switch* to the other door?"

   Once the contestant made her final decision, Monty opened the door she chose to show whether she found the prize.

This game became very famous among mathematicians because the optimal strategy is very counter-intuitive. At the end of the game, the player gets to choose between two doors. One of them contains the prize, the other does not. Thus, on the surface it seems that the choice doesn't matter and that the probability of winning the prize is always 1/2. **This is not true.** It can be shown that the optimal strategy for the contestant is to *never keep, always switch to the other door*. With this strategy, the actual probability of winning the prize is 2/3.

Monty's game today

Monty Hall is still hosting the game show. However, there have been some changes:

- For financial reasons, the number of doors is now d ($d \geq 3$). There is one prize and $d - 1$ goats.
- Monty is very old. When performing a show, with probability p he is tired.

  If Monty isn't tired, he follows the above protocol. However, if he is tired, he wants to avoid unnecessary walking. Therefore, if he has a choice in step 2, he will always open **the door with the smallest number** among the doors he is allowed to open. (He is still not allowed to open the door with the prize nor the door currently chosen by the contestant.)
- In step 3, the contestant gets to choose whether she keeps the door she has or switches to *any other unopened door*. When switching, the contestant gets to choose which one of the other doors she now wants.
- After step 3, the game is over. The remaining $d - 1$ doors are opened and it is revealed whether the contestant won the prize.

Problem specification

You are given the number of doors d and the probability p. Find an optimal strategy for the contestant, and report her probability of winning the prize if she follows that strategy.

Input specification

The first line of the input file contains an integer t specifying the number of test cases. Each test case is preceded by a blank line. Each test case consists of a single line containing the numbers d and p. The value p is always a number from $[0, 1]$ with exactly 6 decimal places.

In the **easy subproblem U1** we have $t = 10$ and $d = 3$.

In the **hard subproblem U2** we have $t = 100$ and $3 \leq d \leq 100$.

Output specification

For each test case, output a single line with a single real number: the optimal probability of winning the prize. Output at least 10 decimal places. Answers within 10^{-9} of our answer will be accepted as correct.

Example

input	output
1 3 0.000000	0.66666666666666667

In this example we have 3 doors and Monty is never tired, so we are playing the original game.

