4^{th} IIUC Inter-University Programming Contest, 2005

H Simple Minded Hashing

Input: standard input
Output: standard output

Problemsetter: Sohel Hafiz

All of you know a bit or two about hashing. It involves mapping an element into a numerical value using some mathematical function. In this problem we will consider a very 'simple minded hashing'. It involves assigning numerical value to the alphabets and summing these values of the characters.

For example, the string "acm" is mapped to 1 + 3 + 13 = 17. Unfortunately, this method does not give one-to-one mapping. The string "adl" also maps to 17 (1 + 4 + 12). This is called collision.

In this problem you will have to find the number of strings of length **L,** which maps to an integer **S,** using the above hash function. You have to consider strings that have only lowercase letters in strictly ascending order.

Suppose **L** = 3 and **S** = 10, there are 4 such strings.

1. abg
2. acf
3. ade
4. bce

agb also produces 10 but the letters are not strictly in ascending order.

bh also produces 10 but it has 2 letters.

Input

There will be several cases. Each case consists of 2 integers **L and S**. **(0 < L, S < 10000)**. Input is terminated with 2 zeros.

Output

For each case, output Case #: where # is replaced by case number. Then output the result. Follow the sample for exact format. The result will fit in 32 bit signed integers.

Sample Input	Output for Sample Input
3 10 2 3 0 0	Case 1: 4 Case 2: 1

2796 - Concert Hall Scheduling

Asia - Aizu - 2003/2004

You are appointed director of a famous concert hall, to save it from bankruptcy. The hall is very popular, and receives many requests to use its two fine rooms, but unfortunately the previous director was not very efficient, and it has been losing money for many years. The two rooms are of the same size and arrangement. Therefore, each applicant wishing to hold a concert asks for a room without specifying which. Each room can be used for only one concert per day.

In order to make more money, you have decided to abandon the previous fixed price policy, and rather let applicants specify the price they are ready to pay. Each application shall specify a period $[i, j]$ and an asking price w, where i and j are respectively the first and last days of the period ($1 \leq i \leq j \leq 365$), and w is a positive integer in yen, indicating the amount the applicant is willing to pay to use a room for the whole period.

You have received applications for the next year, and you should now choose the applications you will accept. Each application should be either accepted for its whole period or completely rejected. Each concert should use the same room during the whole applied period.

Considering the dire economic situation of the concert hall, artistic quality is to be ignored, and you should just try to maximize the total income for the whole year by accepting the most profitable applications.

Input

The input has multiple data sets, each starting with a line consisting of a single integer n, the number of applications in the data set. Then, it is followed by n lines, each of which represents one application with a period $[i, j]$ and an asking price w yen in the following format.

i j w

A line containing a single zero indicates the end of the input.

The maximum number of applications in a data set is one thousand, and the maximum asking price is one million yen.

Output

For each data set, print a single line containing an integer, the maximum total income in yen for the data set.

Sample Input

```
4
1 2 10
2 3 10
3 3 10
1 3 10
6
1 20 1000
3 25 10000
```

```
5 15 5000
22 300 5500
10 295 9000
7 7 6000
8
32 251 2261
123 281 1339
211 235 5641
162 217 7273
22 139 7851
194 198 9190
119 274 878
122 173 8640
0
```

Sample Output

```
30
25500
38595
```

Aizu 2003-2004

2352 - Sequence Alignment

Asia - Dhaka - 2001/2002

Consider a problem of finding the best possible solution of a new kind of right alignment of two strings. As an example, here is the best such right alignment for two strings `AADDEFGGHC` and `ADCDEGH`:

```
AAD-DEFGGHC


 ADCDE--GH-
```

Each vertical matching position scores 2, and each continuous run of gaps scores -1. Thus the total score of an alignment is twice the number of matching positions, less the number of continuous runs of gaps. In the given example, six positions (`A`, `D`, `D`, `E`, `G`, `H`) match, and three runs of gaps are introduced. Thus the score of this alignment is $2 \times 6 + (-1) \times 3 = 9$. Note that we do not penalize for misalignment at the left, as we only consider the problem of finding the best right alignment.

You are to write a program that finds the score of the best right alignment for any two given strings.

Input

The first line of the input file contains an integer N ($1 \leq N \leq 10,000$) indicating the number of test cases to follow. Each test case consists of two lines each of which contains a string of length at most 50. The strings are composed entirely of alpha-numeric characters.

Output

For each test case in the input print a line containing the score of the best possible right alignment of the two given strings.

Sample Input

```
2
AADDEFGGHC
ADCDEGH
ERTTHCBYNQC
BEARTBCHQYN
```

Sample Output

```
9
8
```

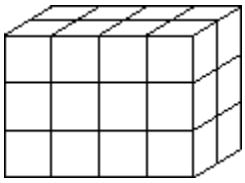
Dhaka 2001-2002

Garbage Heap

Time limit: ? seconds
Memory limit: 64 megabytes

Farmer John has a heap of garbage formed in a rectangular parallelepiped.

It consists of $A \times B \times C$ garbage pieces each of which has a value. The value of a piece may be 0, if the piece is neither profitable nor harmful, and may be negative which means that the piece is not just unprofitable, but even harmful (for environment).



The farmer thinks that he has too much harmful garbage, so he wants to decrease the heap size, leaving a rectangular nonempty parallelepiped of smaller size cut of the original heap to maximize the sum of the values of the garbage pieces in it. You have to find the optimal parallelepiped value. (Actually, if any smaller parallelepiped has value less than the original one, the farmer will leave the original parallelepiped).

Input

The first line of the input contains the number of the test cases, which is at most 15. The descriptions of the test cases follow. The first line of a test case description contains three integers A, B, and C ($1 \le A, B, C \le 20$). The next lines contain $A \cdot B \cdot C$ numbers, which are the values of garbage pieces. Each number does not exceed 2^{31} by absolute value. If we introduce coordinates in the parallelepiped such that the cell in one corner is (1,1,1) and the cell in the opposite corner is (A,B,C), then the values are listed in the order

$$(1,1,1),(1,1,2),\ldots,(1,1,C),$$
$$(1,2,1),\ldots,(1,2,C),\ldots,(1,B,C),$$
$$(2,1,1),\ldots,(2,B,C),\ldots,(A,B,C).$$

The test cases are separated by blank lines.

Output

For each test case in the input, output a single integer denoting the maximal value of the new garbage heap. Print a blank line between test cases.

Examples

Input	Output
1 2 2 2 -1 2 0 -3 -2 -1 1 5	6

UVa Online Judge

12547 RNA Secondary Structure

RNA, which stands for Ribonucleic Acid, is one of the major macromolecules that are essential for all known forms of life. It is made up of a long chain of components called nucleotides. Each component is made up of one of 4 bases and are represented using `A`, `C`, `G` or `U`. The primary structure of RNA is a sequence of these characters. The secondary structure of RNA refers to the base pairing interactions between different components. More specifically, base `A` can pair up with base `U` and base `C` can pair up with base `G`. The stability of the RNA secondary structure depends on the total number of base pairs that can be formed. The final structure is the one that contains the maximum number of base pairs.

Let's represent the primary structure as a string consisting of characters from the set (`ACGU`). The rules of secondary structure formation are as follows:

1. Any base `A` can form a pair with any base `U`
2. Any base `C` can form a pair with any base `G`
3. Each base can be part of at most one pair.
4. Let's assume $w < x$, $y < z$ and $w < y$. If base at index w forms a pair with base at index x and base at index y forms a pair with base at index z, then one of the following two conditions must be true:

$$\begin{aligned} y &> x \\ z &< x \end{aligned}$$

5. There can be at most K pairs between `C` and `G`.

You will be given the primary structure of the RNA of a certain species and your job is to figure out the total number of base pairings in the final secondary structure based on the constraints mentioned above.

You will be given the primary structure in a compressed format that uses run-length encoding. In this type of data compression, consecutive characters having the same value is replaced with a single character followed by its frequency. For example, "`AAAACCGAAUUG`" will be represented using "`A4C2G1A2U2G1`". That means the primary structure will be given in the format $< c_1 f_1 c_2 f_2 c_3 f_3 \ldots c_n f_n >$, where c_i is from the set (`ACGU`) and f_i is a positive integer.

The species that we are dealing with have the following properties:

1. $f_1 + f_2 + f_3 + \ldots + f_n \leq 10050$
2. $f_1 \leq 5000$
3. $f_n \leq 5000$
4. $f_2 + f_3 + f_4 + \ldots + f_{n-1} \leq 50$

Input

The first line of input is an integer T ($T \leq 200$) that indicates the number of test cases. Each case contains two lines. The first line is the primary structure given in run-length encoded format. The second line gives you the value of K ($0 \leq K \leq 20$), that gives an upper limit on the number of $C - G$ base pairs that can be in the final secondary structure.

Output

For each case, output the case number followed by the maximum number of base pairs that can be formed. Look at the samples for exact format.

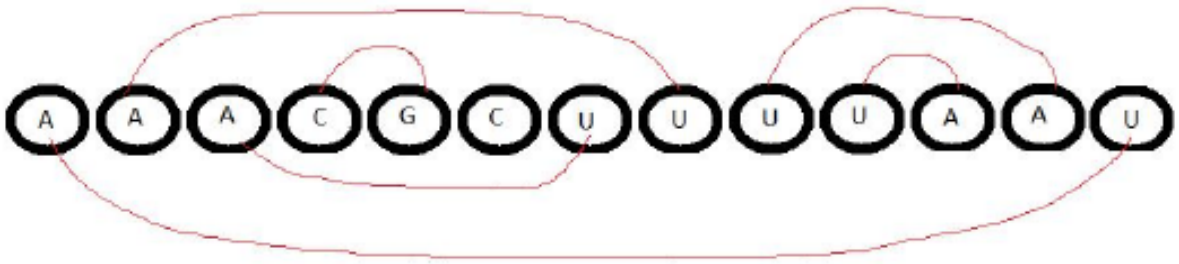
Sample Input

```
3
A3C1G1C1U4A2U1
1
A3C1G1C1U4A2U1
0
A100U200
2
```

Sample Output

```
Case 1: 6
Case 2: 5
Case 3: 100
```

`ILLUSTRATION:` One possible final secondary structure for case 1 is depicted below that shows the 6 base pairings.



Problem C
Best Coalitions

Envy Inc. is a joint stock company, that is, a company in which every stockholder legally owns shares of stock that account for some percentage of the total shares of stock of the company. Due to the global economic crisis, the management rules of Envy Inc. define a particular way for distributing last year's profit: if a stockholder owns more than half of the shares of stock, he/she wins the total profit. Nothing fancy so far in this wild world!

Nevertheless, there are situations in which no stockholder owns more than 50% of the shares of stock of the company. So, in order to gain some profit, stockholders are allowed to form *coalitions*, i.e., groups of stockholders. The participation of the coalition, share-wise, is equivalent to the sum of its stockholders' percentile participation. Hence, if a coalition has more than half of the shares of stock, its members win the totality of last years profit. Then, the members of the coalition receive a part of the profit proportional to their individual participation in the coalition.

For instance, let us assume there are 5 stockholders: A, B, C, D and E, owning 20%, 12%, 14%, 29% and 25% of the stock of the company, respectively. The stockholder E could form several winning coalitions. For example, if E were to form a coalition with A and B, he/she would get 43.86% of last year's profit. If E were to form a coalition with B and C instead, he/she would get 49.02% of last year's profit. On the other hand, E could not form a winning coalition with only A.

Your problem is, given a distribution of shares of stock of Envy Inc., and a stockholder, to determine the maximum percentage of the last year's profit that the given stockholder may win.

Input

The input consists of several test cases, each one defining a percentile distribution of shares of stock, and the index of a stockholder to determine his/her optimal participation. More precisely, each test case is defined by several input lines:

- the first line contains two integer values n ($1 \leq n \leq 100$) and x ($1 \leq x \leq n$), separated by a blank, representing the number of stockholders in Envy Inc. and the index of a stockholder to determine his/her optimal participation, respectively;
- each one of the following n lines has a single floating point value p_i, rounded to 2 decimal places, which represents the percentage of stock ownership of stockholder i ($1 \leq i \leq n$). The floating point delimiter is '.' (i.e. the dot). You can assume that $p_1 + \ldots + p_n = 100$.

The end of the input is indicated by $n = x = 0$, an artificial case that must be ignored.

The input must be read from standard input.

Output

For each given case, output a single line with the corresponding answer. The answer should be formatted and approximated to two decimal places. The floating point delimiter must be '.' (i.e. the dot). The rounding applies towards the *nearest neighbor* unless both neighbors are equidistant, in which case the result is rounded up (e.g. 78.312 is rounded to 78.31; 78.566 is rounded to 78.57; 78.345 is rounded to 78.35, etc.).

The output must be written to standard output.

Sample input	**Output for the sample input**
5 5 20.00 12.00 29.00 14.00 25.00 2 1 56.87 43.13 2 2 56.87 43.13 3 1 10.00 45.00 45.00 0 0	49.02 100.00 43.13 18.18