Problem D
Extend Huffman Coding
Max no. of test cases: 10
Time limit: 2 seconds

Huffman coding is a well-known data compression method. Assume a source generates 4 different symbols $\{A, B, C, D\}$ with probability $\{0.4; 0.35; 0.2; 0.05\}$. A binary tree is generated from left to right (sorting by alphabetic order of symbols) taking the two least probable symbols and putting them together to form an internal node which has a probability that equals the sum of the two symbols. When there are equal probability nodes, the selection of the two least probable nodes always follow the alphabetic order. For example, if a node is formed by $\{B, C\}$ and a node is formed by $\{A, D\}$, node $\{A, D\}$ is at the left of node $\{B, C\}$ because $A < B$ and $A < C$, as in the alphabetic order.

The process is repeated until there is just one node with probability value of 1.

The tree can then be read downwards, from left to right, assigning different bits to different branches. The tree construction is illustrated in Fig. 1.

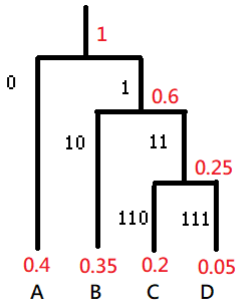

Figure 1: Huffman Binary Tree Construction

The final Huffman code is:

Symbol	*code*
A	0
B	10
C	110
D	111

So, "AAABC" will be encoded as 00010110.

The Huffman coding, however, can be extended by considering more symbol combinations. For example, symbol A's probability is 0.4, then AA's probabiliy is $0.4 * 0.4 = 0.16$. Symbol B's probability is 0.35 then the probability of AB is $0.4 * 0.35 = 0.14$. Let's call it a 2-symbol probability.

In Fig. 2, we show the beginning of how such a 2-symbol tree should be constructed based on the aforementioned alphabetic ordering. In this example, symbol 'CD' and 'DC' have equal probability. By the alphabetic ordering, 'CD' should be selected to merge with 'DD'.

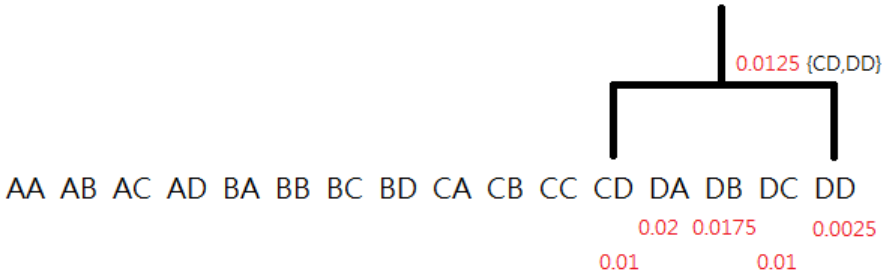

Figure 2: Huffman Binary Tree Construction

Given a set of symbols and their 1-symbol probabilities, please construct 2-symbol Huffman coding to encode a string. sorting by alphabetic order of symbols

Input File Format

There are more than one test cases in the input file. The first number N is the number of test cases. Each test case begins with $n(n <= 10)$ which is the number of symbols. It is followed by n lines of symbol (single character from [A-Z] or [a-z]) and its 1-symbol probability. The last line of the test case is a symbol string with length <= 36, which is an even number.

Output Format

For each test case, please print the length (int bits) of the encoded 0/1 string.

Sample Input

```
2
4
A 0.6
B 0.2
C 0.1
D 0.1
AAABDD
3
a 0.45
b 0.35
c 0.20
baac
```

Output for the Sample Input

```
11
7
```