Communicating images over the internet is a costly process, thanks to the limits in available bandwidth. This problem is resolved to an extent by employing point of interest image coding. To be more specific, the point of interest image coding assumes that, some part of the image is most important, encodes this region with high quality & the other regions of the image with low quality, thus reducing file size.

You are to implement a very simple case of point of interest image coding. Assume an image file is represented by a grid of letters from '`A`' to '`Z`', each letter representing one pixel. All pixels represented by the same letter represents a same region in the image whereas two different letters stand for two different regions. The region with maximum number of letters is to be considered as the most important region. If there are more than one region tied for the maximum, treat all of them as most important. Every single letter in the most important region(s) is encoded as M byte data while each non-important region letter is encoded as N bytes. You are to determine the image file size. You can assume the image file consists of image data only i.e. no image header, title, meta information etc.

Input

The input file starts with X, the number of test cases ($1 \leq X \leq 50$). X cases follows. Each test case begins with 4 integers in a line, R ($1 \leq R \leq 20$), C ($1 \leq C \leq 20$), M ($1 \leq M \leq 10$) & N ($1 \leq N \leq 10$) where R & C are the number of rows & columns in the pixel grid respectively. M is the number of bytes for each important pixel & N is the number of bytes for each non-important pixel. The following R lines describe the pixel grid. Each of these lines contain C letters between '`A`' & '`Z`' (inclusive).

Output

For each test case, print a line in the format '`Case` x`:` y' where x is the case number & y is the number of bytes in the image file.

Explanation of sample I / O:

The image contains 10 regions namely '`A`', '`B`', '`C`', '`D`', '`E`', '`F`', '`G`', '`Z`', '`I`' & '`P`'. Region '`A`' has the highest frequency of 6. So, this is the most important region. We encode 6 letters of this region with 2 bytes each & each of the remaining 14 letters with 1 byte. That makes the total file size = 6 * 2 + 14 * 1 = 26 bytes.

Sample Input

```
1
5 4 2 1
ABCD
ABCA
EFAC
BCAG
AZIP
```

Sample Output

```
Case 1: 26
```