Consider a grid of size $1 \times N$. Each cell of the grid has the following properties

- Cell C of the grid has a value of V_C $(1 \leq C \leq N)$
- The value of each cell is a positive integer less than 26
- Some of the cells are *special* and they are represented with the character '`X`'
- Cell C has a weight of 2^{V_C} (two to the power of cell value)
- The special cells have weights of 1

You will be given the values of these N cells and your job will be to divide these into K segments so that

- Each segment contains at least one cell
- There is at least one *special* cell in each segment

The weight of a segment is equal to the product of the weights of the cells it contains. You have to form segments in such a way so that ratio

$$(Highest\ weight\ of\ all\ the\ segments)/(Lowest\ weight\ of\ all\ the\ segments)$$

is minimized.

In case there are multiple answers with the same lowest ratio, you have to make sure the number of cells in the first segment is maximized. If there is still a tie, then make sure the number of cells in the second segment is maximized and so on.

Example:

```
N = 5 and K = 2
Cell values = \{1 2 X 3 X \}
Cell weights = \{2 4 1 8 1\}
Optimal segmentation = (2 4 1)(8 1)
Weights of segments = (8)(8)
Ratio = 1
Final Result = (1 2 X)(3 X)
```

Input

The first line of input is an integer T $(T \leq 200)$ that indicates the number of test cases. Each case starts with two integers N $(1 < N < 31)$ and K $(1 < K < 16)$. The meaning of N and K are mentioned above. The next line contains N integers where the I-th integer gives the value of V_I. The integers that are special will be represented by '`X`'.

Output

For each case, output the case number first. If there is no way to divide the N cells into K segments, meeting the constraints above, then print '`not possible!`'. If there is a way but the ratio is greater than 10^{15} then print '`overflow`'. If the ratio is not greater than 10^{15} then output the ratio first followed by the segmentations. Each segment is enclosed by brackets. Look at the output for detailed format.

Sample Input

```
4
5 2
1 2 X 3 X
6 3
X X 2 3 4 5
10 3
X X X 25 25 25 25 25 25 25
10 3
4 X 3 1 X 3 X X 3 2
```

Sample Output

```
Case 1: 1 (1 2 X)(3 X)
Case 2: not possible!
Case 3: overflow
Case 4: 8 (4 X 3)(1 X 3 X)(X 3 2)
```