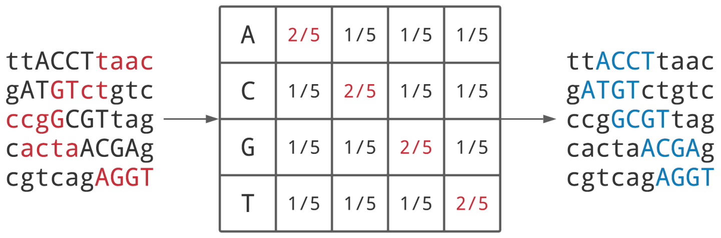
2F Implement RandomizedMotifSearch

Randomized Motif Search Problem
Implement RandomizedMotifSearch.

Input: A collection of strings *Dna*, and integers *k* and *t*.
Output: A collection of strings resulting from running RANDOMIZEDMOTIFSEARCH(*Dna*, *k*, *t*) 1000 times. Remember to use pseudocounts!



Formatting

Input: Space-separated integers *k* and *t*, followed by a newline-separated collection of strings *Dna*.
Output: A space-separated list of strings containing a collection of strings resulting from running RANDOMIZEDMOTIFSEARCH(*Dna*, *k*, *t*) 1000 times. Remember to use pseudocounts!

Constraints

- The integer *k* will be between 1 and 10^2.
- The integer *t* will be between 1 and 10^2.
- The number of strings in *Dna* will be between 1 and 10^2.
- The length of each string in *Dna* will be between 1 and 10^2.
- Each string in *Dna* will be a DNA string.

Test Cases

Case 1

Description: A small and hand-solvable dataset taken from the example problem on Stepik.

Input:
```
8 5
CGCCCCTCTCGGGGGTGTTCAGTAAACGGCCA GGGCGAGGTATGTGTAAGTGCCAAGGTGCCAG
TAGTACCGAGACCGAAAGAAGTATACAGGCGT TAGATCAAGTTTCAGGTGCACGTCGGTGAACC
AATCCACCAGCTCCACGTGCAATGTTGGCCTA
```

Output:
```
TCTCGGGG CCAAGGTG TACAGGCG TTCAGGTG TCCACGTG
```

Case 2

Description: This dataset checks if your code has an off-by-one error at the beginning of each sequence of *Dna*. Notice that the some of the motifs of the solution occur at the beginning of their respective sequences in *Dna*, so if your code did not check the first *k*-mer in each sequence of *Dna*, it would not find these sequences.

Input:
```
6 8
AATTGGCACATCATTATCGATAACGATTCGCCGCATTGCC
GGTTAACATCGAATAACTGACACCTGCTCTGGCACCGCTC
AATTGGCGGCGGTATAGCCAGATAGTGCCAATAATTTCCT
GGTTAATGGTGAAGTGTGGGTTATGGGGAAAGGCAGACTG
AATTGGACGGCAACTACGGTTACAACGCAGCAAGAATATT
GGTTAACTGTTGTTGCTAACACCGTTAAGCGACGGCAACT
AATTGGCCAACGTAGGCGCGGCTTGGCATCTCGGTGTGTG
GGTTAAAAGGCGCATCTTACTCTTTTCGCTTTCAAAAAAA
```

Output:
```
CGATAA GGTTAA GGTATA GGTTAA GGTTAC GGTTAA GGCCAA GGTTAA
```

Case 3

Description: This dataset checks if your code has an off-by-one error at the end of each sequence of *Dna*. Notice that the some of the motifs of the solution occur at the end of their respective sequences in *Dna*, so if your code did not check the last *k*-mer in each sequence of *Dna*, it would not find these sequences.

Input:
```
6 8
GCACATCATTAAACGATTCGCCGCATTGCCTCGATTAACC
TCATAACTGACACCTGCTCTGGCACCGCTCATCCAAGGCC
AAGCGGGTATAGCCAGATAGTGCCAATAATTTCCTTAACC
AGTCGGTGGTGAAGTGTGGGTTATGGGGAAAGGCAAGGCC
AACCGGACGGCAACTACGGTTACAACGCAGCAAGTTAACC
AGGCGTCTGTTGTTGCTAACACCGTTAAGCGACGAAGGCC
AAGCTTCCAACATCGTCTTGGCATCTCGGTGTGTTTAACC
AATTGAACATCTTACTCTTTTCGCTTTCAAAAAAAAGGCC
```

Output:
```
TTAACC ATAACT TTAACC TGAAGT TTAACC TTAAGC TTAACC TGAACA
```

Case 4

Description: A larger dataset of the same size as that provided by the randomized autograder. Check input/output folders for this dataset.