1E Find Patterns Forming Clumps in a String

Clump Finding Problem
Find patterns forming clumps in a string.

Input: A DNA string *Genome*, and integers k, L, and t.
Output: All distinct k-mers forming (L, t)-clumps in *Genome*.

```
aggtccTGCAATGCATGAAGCCTGCAtgtt
```

Formatting

Input: A DNA string *Genome* followed by space-separated integers k, L, and t.
Output: A space-separated list of strings containing all distinct k-mers forming (L, t)-clumps in *Genome*.

Constraints

- The length of *Genome* will be between 1 and 10^4.
- The integer k will be between 1 and 10^1.
- The integer L will be between 1 and 10^3.
- The integer t will be between 1 and 10^2.
- *Genome* will be a DNA string.

Test Cases

Case 1

Description: The sample dataset is not actually run on your code.

Input:

```
CGGACTCGACAGATGTGAAGAACGACAATGTGAAGACTCGACACGACAGAGTGAAGAGAAGAGGAAACATTGTAA
5 50 4
```

Output:

```
GAAGA CGACA
```

Case 2

Description: This dataset makes sure that your code only counts *k*-mers that fall *completely* within a given *L*-window.

Input:

```
CTAAAACGTCG
2 4 2
```

Output:

```
AA
```

Case 3

Description: This dataset checks if your code has an off-by-one error when checking *k*-mers within an *L*-window. Notice that, for each 1-mer (A, C, G, and T), there are 3 nucleotides between the first and second occurrence. In other words, each nucleotide occurs twice in a specific 5-window: once at the beginning of the 5-window, and once at the end: **A**CGT**A**CGT A**C**GTA**C**GT AC**G**TAC**G**T ACG**T**ACG**T**.

Input:

```
ACGTACGT
1 5 2
```

Output:

```
T C A G
```

Case 4

Description: This dataset checks if your code is correctly handling overlapping *k*-mers.

Input:

```
CCATATACC
3 5 2
```

Output:

```
ATA
```

Case 5

Description: A larger dataset of the same size as that provided by the randomized autograder.