## 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:**

```
CGGACTCGACAGATGTGAAGAAATGTGAAGACTGAGTGAAGAGAAGAGGAAACACGACACGACATTGCGA...
...CATAATGTACGAATGTAATGTGCCTATGGC
5 75 4
```

**Output:**

```
CGACA GAAGA AATGT
```

### Case 2

**Description:** This dataset makes sure that your code only counts *k*-mers that fall *completely* within a given *L*-window. For example, take the 4-window starting at index 4 (`AAAACGTCGAAAAA`). One might think that the 2-mer `CG` occurs twice in this window since the first letter of the second occurrence happens at the very end of the window. However, since the second occurrence of `CG` does not fall entirely in this 4-window, it does not count. Thus, the only result is `AA`.

**Input:**

```
AAAACGTCGAAAAA
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:**

```
A C G T
```

**Case 4**

---

**Description:** This dataset checks if your code is correctly handling overlapping *k*-mers. For example, ATA forms a (5, 2)-clump in CCCATATACCC (CCC**ATATA**CCC and CCCAT**ATACC**C).

**Input:**

```
CCACGCGGTGTACGCTGCAAAAAGCCTTGCTGAATCAAATAAGGTTCCAGCACATCCTCAATGGTTTCAC...
...GTTCTTCGCCAATGGCTGCCGCCAGGTTATCCAGACCTACAGGTCCACCAAAGAACTTATCGATTAC...
...CGCCAGCAACAATTTGCGGTCCATATAATCGAAACCTTCAGCATCGACATTCAACATATCCAGCG
3 25 3
```

**Output:**

```
AAA CAG CAT GCC TTC
```

**Case 5**

---

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