1D Find All Occurrences of a Pattern in a String

Pattern Matching Problem
Find all occurrences of a Pattern in a string.

Input: DNA strings *Pattern* and *Genome*.
Output: All starting positions in *Genome* where *Pattern* appears as a substring.

```
AGAGATCAGA
AGAGA  AGA
0 2    7
```

Formatting

Input: DNA strings *Pattern* and *Genome*.
Output: A space-separated list of integers representing each starting position in *Genome* where *Pattern* appears as a substring.

Constraints

- The length of *Pattern* will be between 1 and 10^1.
- The length of *Genome* will be between 1 and 10^4.
- *Pattern* and *Genome* will be DNA strings.

Test Cases

Case 1

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

Input:

```
GCAT
AAGCATAAGCATAA
```

Output:

```
2 8
```

Case 2

Description: This dataset checks if your code is written correctly but is also taking into account reverse complements, which we are not yet doing. Even though the reverse complement of `ACAC` (which is `GTGT`) occurs in *Genome*, we only want to count occurrences of `ACAC` specifically, which only occurs at index `4`.

Input:

```
ACAC
TTTTACACTTTTTTGTGTAAAAA
```

Output:

```
4
```

Case 3

Description: This dataset checks for off-by-one errors at the beginning of *Genome*. Notice that `AAA` occurs at the very beginning of *Genome*, so if you were to miss the first *k*-mer of *Genome*, your code would output the following: `7 12 16`.

Input:

```
AAA
AAAGAGTAAATTAAATAAACTGA
```

Output:

```
0 7 12 16
```

Case 4

Description: This dataset checks for off-by-one errors at the end of *Genome*. Notice that `TTT` occurs at the very end of *Genome*, so if you were to miss the last *k*-mer of *Genome*, your code would output the following: `2 8 13`.

Input:

```
TTT
GCTTTGCCTTTGCTTTGCTTT
```

Output:

```
2 8 13 18
```

Case 5

Description: This test dataset checks if your code correctly handles cases where instances of *Pattern* overlap in *Genome*. In this case, if you did not count overlaps, you would only find the first and last instances of `ATA` (**`ATA`**`TATA` and `ATAT`**`ATA`**). However, there is indeed a third occurrence, where the other two overlap (`AT`**`ATA`**`TA`).

Input:

```
ATA
ATATATA
```

Output:

```
0 2 4
```

Case 6

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