

## 9H Pattern Matching with the Suffix Array

---

### Multiple Pattern Matching with the Suffix Array

Use the suffix array of a string to find all occurrences of a collection of *Patterns*.

**Input:** A string *Text* and a collection *Patterns* containing (shorter) strings.

**Output:** All starting positions in *Text* where a string from *Patterns* appears as a substring.

```
7 $
1 ANANAS$
3 ANAS$
5 AS$      ANA: 1 3
0 BANANAS$ BAN: 0
2 NANAS$
4 NAS$
6 S$
```

---

### Formatting

**Input:** A string *Text* and a space-separated list of strings *Patterns*

**Output:** A newline-separated list of strings from *Patterns*. Each *Pattern* in *Patterns* is followed by a colon (":") and a space-separated list of starting indices in *Text* where *Pattern* appears as a substring.

### Constraints

- The length of *Text* will be between 1 and  $10^4$ .
- The number of patterns in the string-set *Patterns* will be between 1 and  $10^1$ .
- The length of any one pattern in *Patterns* will be between 1 and  $10^4$ .

## Test Cases

### Case 1

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

#### Input:

```
AATCGGGTTCAATCGGGT
ATCG GGGT
```

#### Output:

```
ATCG: 1 11
GGGT: 4 15
```

#### Figure:

Burrows-Wheeler Matrix	Suffix Array
\$AATCGGGTTCAATCGGGT	19
AATCGGGGT\$AATCGGGTTC	10
AATCGGGTTCAATCGGGGT\$	0
ATCGGGGT\$AATCGGGTTCA	11
ATCGGGTTCAATCGGGGT\$A	1
CAATCGGGGT\$AATCGGGTT	9
CGGGGT\$AATCGGGTTCAAT	13
CGGGTTCAATCGGGGT\$AAT	3
GGGGT\$AATCGGGTTCAATC	14
GGGT\$AATCGGGTTCAATCG	15
GGGTTCAATCGGGGT\$AATC	4
GGT\$AATCGGGTTCAATCGG	16
GGTTCAATCGGGGT\$AATCG	5
GT\$AATCGGGTTCAATCGGG	17
GTTCAATCGGGGT\$AATCGG	6
T\$AATCGGGTTCAATCGGGG	18
TCAATCGGGGT\$AATCGGGT	8
TCGGGGT\$AATCGGGTTCAA	12
TCGGGTTCAATCGGGGT\$AA	2
TTCAATCGGGGT\$AATCGGG	7

The complete Burrows-Wheeler matrix shown above can be inferred using only the Burrows-Wheeler transform of this string and the Last-To-First property of the Burrows-Wheeler transform. We then search for our query strings, ATCG and GGGT, as prefixes in the rows of our matrix. Finally, we can use the suffix array of the database string to locate the positions of query matches.

## Case 2

---

**Description:** There are no matches in *Text* to any pattern in *Patterns*.

**Input:**

```
ATATATATAT
GT AGCT TAA AAT AATAT
```

**Output:**

```
GT:
AGCT:
TAA:
AAT:
AATAT:
```

## Case 3

---

**Description:** *Text* contains overlapping occurrences of *Patterns*.

**Input:**

```
bananas
ana as
```

**Output:**

```
ana:  1 3
as:   5
```

## Case 4

---

**Description:** Large regions of *Text* being a single character or short tandem repeat (STR).

**Input:**

```
AAACAA
AA
```

**Output:**

```
AA: 0 1 4
```

### Case 5

---

**Description:** *Text* is palindromic or has substrings that are palindromic.

**Input:**

GAGCAT

GA AG

**Output:**

GA: 0

AG: 1

### Case 6

---

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