String Matching - Exercise

1. Given the text T and a pattern P, implement the naive algorithm to find all valid shifts with which the given pattern P occurs in T.

   Function Prototype: `int* naiveStringMatcher(char *T, char *P);`

2. Given the text T and a pattern P, implement the KMP algorithm to find all valid shifts with which the given pattern P occurs in T.

   Function Prototype: `int* kmpStringMatcher(char *T, char *P);`

3. Consider that the text T and pattern P are strings of lowercase alphabets. Additionally, P can contain a special character “-” indicating any lowercase alphabet. Example: P = “ac-b” can indicate “acab” or “acbb” or “accb” or “acdb” or any string with “-” replaced by any alphabet from a-z. There can be zero or more occurrences of “-” in P. Implement an algorithm to find all valid shifts with which the given pattern P occurs in T.

   Function Prototype: `int* anyCharStringMatcher(char *T, char *P);`

4. Consider that the text T and pattern P are 2 dimensional arrays of size $n \times n$ and $m \times m$ respectively, where $m \leq n$. Implement a naïve algorithm to find all valid shifts with which the given pattern P occurs in T. Here, a shift refers to a pair of integers indicating the shift in rows and columns.

   Function Prototype: `int** naive2dArrayMatcher (int T[][],int P[][]);`