Worksheet 11: Text Processing

The goal of this worksheet is to test the text processing algorithms covered in the course.

Download the file `PatternMatching.java` that was developed in the lectures.

1. Copy the PatternMatching class to create a new class called W3Q1.java and create a new static method with the signature given below and implement the KMP Search Algorithm given in the notes (you may need to create additional methods, but these should be declared private because they are part of the implementation of the KMP Search Algorithm.

   ```
   public static int kmpSearch(String text, String pattern) {…}
   ```

   Test your answer by creating main method that searches the text “the theme was their idea” for the following patterns:
   - the, there, idea

2. Copy the PatternMatching class to create a new class called W3Q1.java and create a new static method with the signature given below and implement the Boyer-Moore Search Algorithm given in the notes (you may need to create additional methods, but these should be declared private because they are part of the implementation of the Boyer-Moore Search Algorithm.

   ```
   public static int boyerMooreearch(String text, String pattern) {…}
   ```

   Again, test your answer by creating a class W3Q2.java that contains a main method that searches the text “the theme was their idea” for the following patterns:
   - the, there, idea

   HINT: Java provides a generic implementation of HashMaps (java.util.HashMap) and a Map interface (java.util.Map). Please use these for the implementation of the last occurrrence function.

3. Create a class called W3Q3.java and copy the search algorithms you implemented in Q1 and Q2. Instrument your search algorithms (add code) to counting the number of comparisons that take place and to print this value to the console when each algorithm terminates.

   Implement a main method that performs the tests carried out in Q1 and Q2 and compare the number of comparisons. Add a comment at the top of the class that identifies, for each test, which algorithm took the least number of comparisons.

   HINT: the comment should look something like this:

   ```
   /**
    * This is my answer…
    */
   public class W3Q3 { … }
   ```

4. Create a class called W3Q4.java and copy the search algorithms you implemented in Q1 and Q2. Modify the implementations of all the search algorithms to allow the provision of an offset for the text (i.e. so we can say – search for this pattern in the text starting at character i).

   ```
   public static int kmpSearch(String text, String pattern, int offset) {…}
   ```

   Test your implementations by writing a program that finds all the occurrences of “the” in “the cat sat on the mat” using each of the three search algorithms.