Programming Assignment #3

Due at 11:59pm, Sunday, April 10

All the codes should be written in c or c++ for linux and commented appropriately for major steps/functions

Code that does not compile will not be graded and get a 0 automatically

The codes should be submitted as a single zipped file through Blackboard

Implement Horspool's String Matching Algorithm using C or C++. (100 pts)

ALGORITHM *HorspoolMatching*$(P[0..m-1], T[0..n-1])$

```
//Implements Horspool's algorithm for string matching
//Input: Pattern P[0..m − 1] and text T[0..n − 1]
//Output: The index of the left end of the first matching substring
//        or −1 if there are no matches
ShiftTable(P[0..m − 1])    //generate Table of shifts
i ← m − 1                  //position of the pattern's right end
while i ≤ n − 1 do
    k ← 0                  //number of matched characters
    while k ≤ m − 1 and P[m − 1 − k] = T[i − k] do
        k ← k + 1
    if k = m
        return i − m + 1
    else i ← i+Table[T[i]]
return −1
```

Requirements:

- Your code should be able to read an input ASCII file named 'input.txt', where the first line contains the string of pattern, and the second line contains the string of text
- You can assume that all characters in both the pattern and text are belong to 26 letters of English alphabet plus the space
- Your code will produce an output ASCII file named 'output.txt', which contains the index of the left end of the **first** matching substring or -1 if no match

- Your code should output the execution time for running string matching excluding time of input/output.
- A script file or readme file including the instructions to compile and run the code should be submitted together with the codes

Bonus question:

Following the steps in lecture notes, implement Boyer-Moore's String Matching Algorithm using C or C++. (50 pts)

Requirements:

- Your code should be able to read an input ASCII file named 'input.txt', where the first line contains the string of pattern, and the second line contains the string of text
- You can assume that all characters in both the pattern and text are belong to 26 letters of English alphabet plus the space
- Your code will produce an output ASCII file named 'output.txt', which contains the index of the left end of the **first** matching substring or -1 if no match
- Your code should output the execution time for running string matching excluding time of input/output.
- A script file or readme file including the instructions to compile and run the code should be submitted together with the codes