COSC 320 – 001

Analysis of Algorithms

2022/2023 Winter Term 2

Project Topic Number: 1

Milestone 1

Keyword Replacement for Text

Group Lead: Brandon Mack

Group Members: Aidan Elliott, Eddy Zhang

Problem Formulation. Formulate the problem as an algorithmic problem, including the mathematical notations. For example, given an input array A of length n, the algorithm should output A in sorted order.

Given a string A, the algorithm creates an array B by splitting the string by every space. The algorithm then iterates through B and looks up each value of B, i, into a Hashmap C. If i is a key of C, then i is replaced by the value found when searching I into the Hashmap. B is then outputted with all the abbreviations converted to their new words.

Pseudo-code.

```
ourAlgorithm(String sentence){
B = sentence split with " " as delimiter
C = preloaded hashmap
For i in length of B:
    If B[i] is a key of C:
        B[i] = C[B[i]]   #Replace B[i] with value found when looking up B[i] into hashmap
Return B
}
```

Algorithm Analysis. Include the analysis of the algorithm and proof of correctness and running time of your algorithm.

Loop Invariant:

For 0<= i < n*m, where m is the number of posts and n is the number of words in a post, all words between 0 and i(inclusive) are processed, which means that all the abbreviations are replaced with respective words.

Inductive Hypothesis:

The loop invariant holds true for all iterations.

Base case:

When i = 0(first word), if the word is not an abbreviation then it won't be replaced. If it is an abbreviation it will be replaced. The loop invariant holds true for i=0.

Inductive case:

Assume the loop invariant holds true for k iterations.
For i = k+1, all abbreviations between 0 and k+1(inclusive) have been replaced with full words.
For i = k+2, all abbreviations between 0 and k+2(inclusive) have been replaced with full words.

Therefore, we conclude that the loop invariant holds true for each iteration of the loop.
When i = n*m, all abbreviations are replaced.

Unexpected Cases/Difficulties. If you encountered any issues for design and implementation of your work, include it here. Also, mention the solutions you have considered to alleviate the issue.

- If a user enters an empty string, the system will return an empty string to the user, however the algorithm is capable of handling this itself.

- The original plan was to have the method take an array of words as the input. However, this requires to pre-process the data and converting it from a string. Our decision to instead take the raw string as an input limits any pre-processing of data and handles it within the algorithm.

- The correctness of the algorithm analysis depends on the correct implementation of the algorithm. The loop invariant might not hold true if there's a glitch in the code. Also, due to case sensitivity and users' typing behavior, the algorithm might not be able to accurately replace all acronyms in a processed string.