MILESTONE 3

PLAGIARISM CHECKER TOOL

* **PSUEDOCODE CORRECTNESS AND COMPLEXITY (Rabin-Karp)**

1. Assume the text is length **n** and the length of the word is **m**. The best- and average-case running time of Rabin-Karp is **O(m+n)**because the rolling hash step takes **O(n)** time and once the algorithm finds a potential match, it must verify each letter to make sure that the match is true and not a result of a hashing collision and therefore must check each of the m letters in the word.
2. The worst-case running time of Rabin-Karp is **O(nm)**. This would occur with an extremely awful hash function that resulted in a false positive at each step. Since whenever the algorithm thinks it found a match, it must verify each of the m letters in the word, if there is a collision at each step, m letters will be checked n times resulting in a running time of **O(nm)**. This can be avoided with a good choice of hash function.

If n=m then it takes Ω(n)to verify that T=P. The best-case running time of *any* string-matching algorithm is Ω(m), since this is how long it takes to verify a match. Perhaps we are disregarding the time it takes to compute the hashes.

Ignoring all of this, let us suppose that we are considering an algorithm whose running time is Θ(n−m+1), where m ≤ n are two parameters. As the example makes clear, this is *not* the same as Θ(n), since if m is close to n then n−m+1 is much smaller than n. Generally speaking, an asymptotic expression depending on two parameters cannot be reduced to an asymptotic expression depending on only one of them.

In practice, often m is itself a known function of n, and this can be used to reduce Θ(n−m+1) to an asymptotic expression depending on a single parameter. For example, if m=n/2 then Θ(n−m+1) = Θ(n), whereas if m=n, then Θ(n−m+1) = Θ (1).