**Lab 2: Exercise 3**

## 1. Indicate the advantage(s) of using the hash function of fig. 5.4 of the course book (also used in **main.cpp**), instead of the hash function of the program in **test.cpp**.

1.1

The hash function in *main.cpp* has a more even distribution of the values. This is because the values are distributed seemingly random with the base 37. The value 37 is chosen since it is a prime number larger than the amount of characters in the alphabet (31 could have been used as well, but 37 is considered better by most programmers). Incorporating a prime number in the hashing introduces more “uniqueness” thanks to the nature of the primes.

1.2

The hash in test.cpp simply sums the integer value of each character in the word and creates a hash out of the sum. This implies that every permutation of the same string gets the same hash:

“hej”, “hje”, “jeh” etc.

The hash in main.cpp weighs each character differently. The character at place n gets multiplied by 37 n times, which means that different permutations of the same string receive different sums (i.e. different hashes). This is a huge advantage since the English language contains many anagrams (i.e. permutations of the same string).

## 2. What is the main difference between **std::map** and the hash table you have implemented in this lab?

The std::map stores its data ordered by key. This is so that one can iterate over a subset of the elements in the map based on their key. The HashTable implemented in this lab has no particular order other than the “organized chaos” created by the hash function.