Surabhi Ravishankar
Email: surabhiravishankar2016@u.northwestern.edu
EECS 395 – Data Structures and Data Management

PROJECT-3 REPORT

Execution:

```
$ make
$ ./spellcheck Program3Fall2014input
```

Design and Implementation:

The goal of this project was to implement a spellchecker. I implemented 4 algorithms to perform the spell check although only 3 were required.

Design of the Hash Table:

The Hash Table was designed in store all the words in the dictionary. Each slot in the hash table consists of a key value pair. Where the key is the index and value is the word that is being stored.

The add function of the HashTable reads the word from the file one by one and adds it to the Dictionary Hash Table. I have used an Array Implementation throughout the program.

Double Hashing was used as a collision resolution strategy. The hash function used was the one given in the Problem Statement.

The lookup() function iterates though the entire hash table and returns true if the word is present.

Design of the Spell Check Algorithms:

I used the built-in substr() function of the string library class extensively in order to implement the spellcheck algorithms.

CheckInsert()
Check Insert works by inserting a letter (a - z) between each of the letters of the word. At each iteration it performs a lookup in the dictionary and return if the word is present. The algorithm is given in the comments.

CheckReplace()
Check Replace works by replacing each character in the word with a letter from a - z between each of the letters of the word. At each iteration it performs a lookup in the dictionary and returns if the word is present. The algorithm is given in the comments.

CheckSwap()
CheckSwap works by replacing adjacent characters of the input word and returns when the word is found in the dictionary. The algorithm is given in the comments.

CheckDelete()
CheckDelete works by deleting each letter in the word and checking it against the dictionary hash table. The algorithm is given in the comments.