**Spell Checker Report**

Firstly, I created the “hashTable” class to represent the hash table data structure. I used the hash table to store the dictionary of words. This class has different methods that initialize the hash table (\_\_init\_\_), hash a word to get an index (hashFunction), insert a word into the hash table (insert), and searches for a word in the hash table (search).

Secondly, I created a SpellChecker class which has the main focus of checking the spelling of the words in the dictionary. This class contains methods that initializes the spell checker with a hash table from a dictionary file(\_\_init\_\_), loads dictionary from the file into the hash table (loadDictionary), checks the spelling of a text (checkSpelling), suggests corrections for misspelled words (suggestCorrections), and adds new words to the dictionary (addToDictionary).

The main function prompts the user to enter the name of the dictionary file, if it’s a correct file name then it presents a menu that lets the user check spelling of words in file, add words, or quit the problem.

**Techniques Used**

I used a hash table to store the dictionary allows to lookup words and to insert words in the dictionary. I used hash function which assigns words to indexes in the hash table. It’s effective and minimizes collisions and ensures fast access to words. The spell checking algorithm effectively checks for misspelled words and suggest corrections.

**Time complexity**

The time complexity is O(no. of words \* average length of a word). It depends mainly on the number of words in the text and the length of each word.

**User Manual**

1. Download the main file and the dictionary file.
2. When run is pressed, type in the path to the dictionary file (“dictionaryFile.txt”).
3. List of options will be shown for you to choose from.
4. Follow the steps on the screen for whichever option you choose.
5. To exit, press 3.