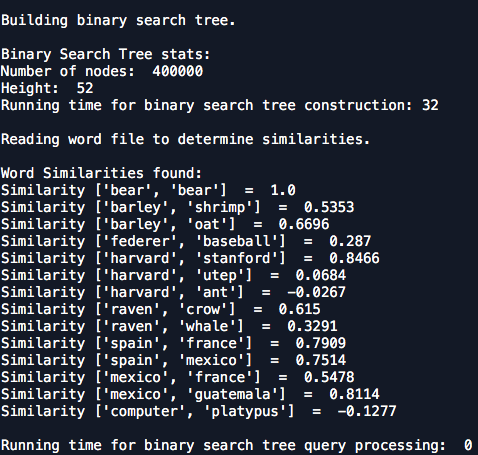
The Lab 5 consisted of doing a program that will implement a binary search tree or a hash table that will compare words and let the user know if they are similar from a scale of 1 to -1. The program will use word embeddings included in glove.6B.50d.txt. This will be used when comparing words in a different file that will be named wordlist.txt. The program will use the classes for hash tables or from the binary search tree depending on what the user chooses.

Starting with the binary search tree implementation, under bstmode() the running time for it consists of 2n which will have a runtime of O(n). This is without considering the methods it calls to make the comparisons for the word embeddings, only reading the files. Insert() which will insert the word and its embeddings uses recursion which is based on T(n)=2T(n-1)+1. This will give us a running time of O(n). NodeNum() which counts the number of nodes in the bst is the same with a recursive equation of T(n)=2T(n-1)+1 which it’s a running time of O(n). Height() has recursive equation of T(n)=2T(n/2)+1 which has a running time of O(log n). Find() also has the same recursive equation as NodeNum() which has s running time of O(log n).

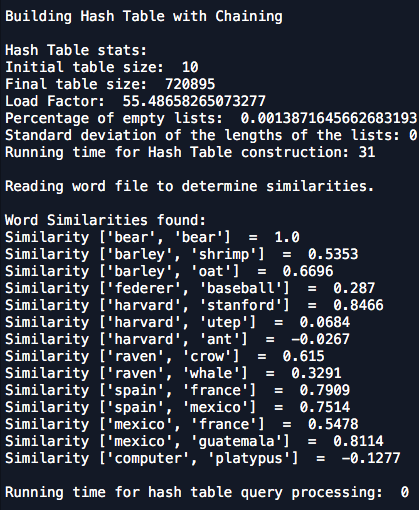
With the hash table implementation, under hashmode() the running time consists of the same for just reading the files with O(n). Now including the methods used for the hash table starting the EmptyListNum(H), its running time is O(n). For InsertC(H), the worst-case runtime is when the hash table needs to be doubled which will have a O(n^2). For FindC(H, line2[]), the running time is O(n).

I ended up learning a lot after completing this lab. I got a general idea of how Natural Language Processing use word embeddings. Additionally, implementing two different data structures to use the same data in the same way. In this case I found that the Hash Table was faster by 3 seconds during runtime. Additionally I found that coding this project using the Hash Table was easier than using the Binary Search Tree.

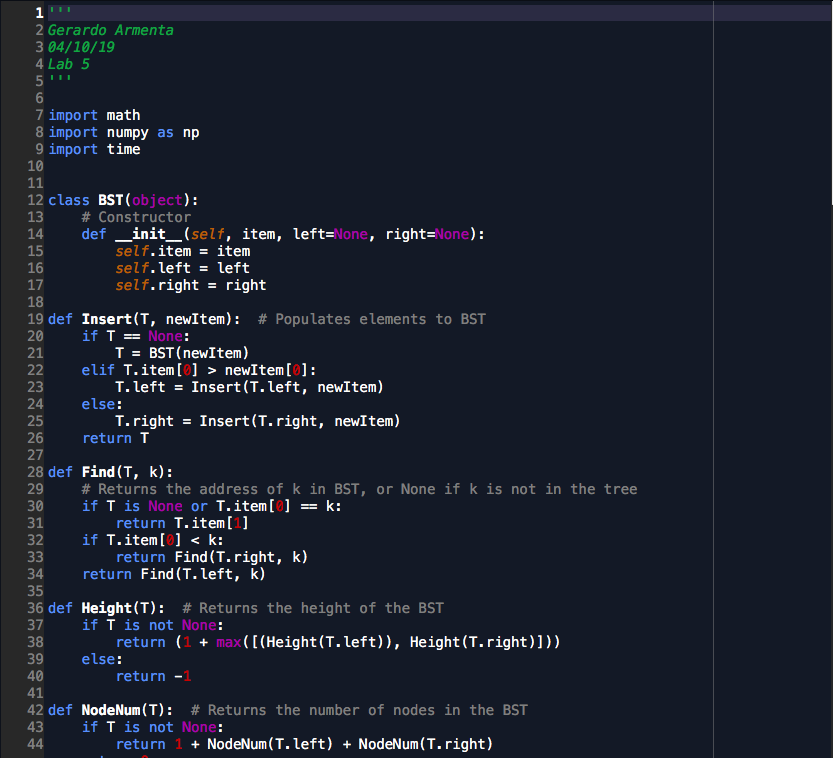
Screen Shot of the BST stats:

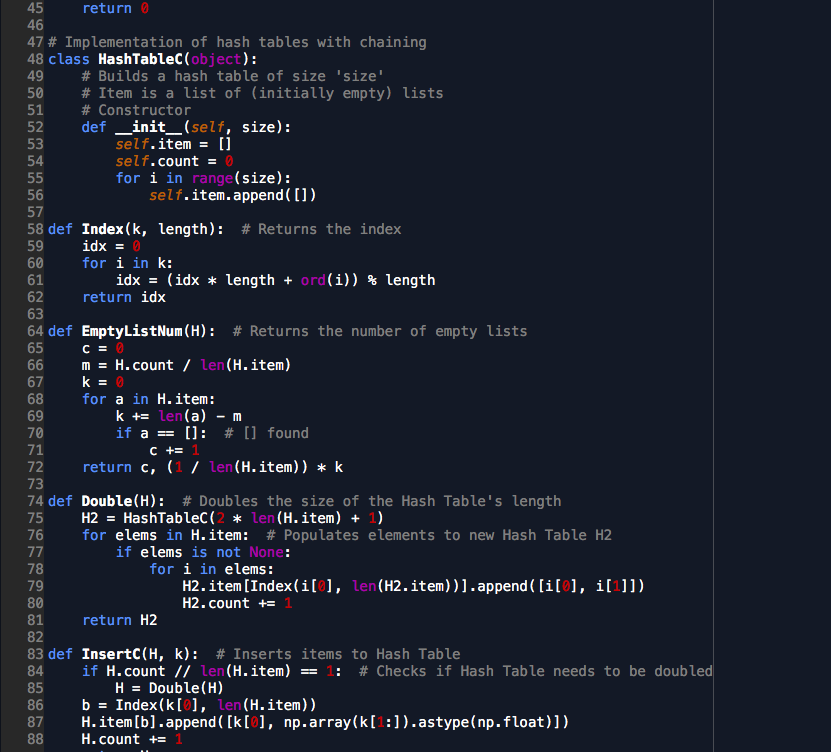


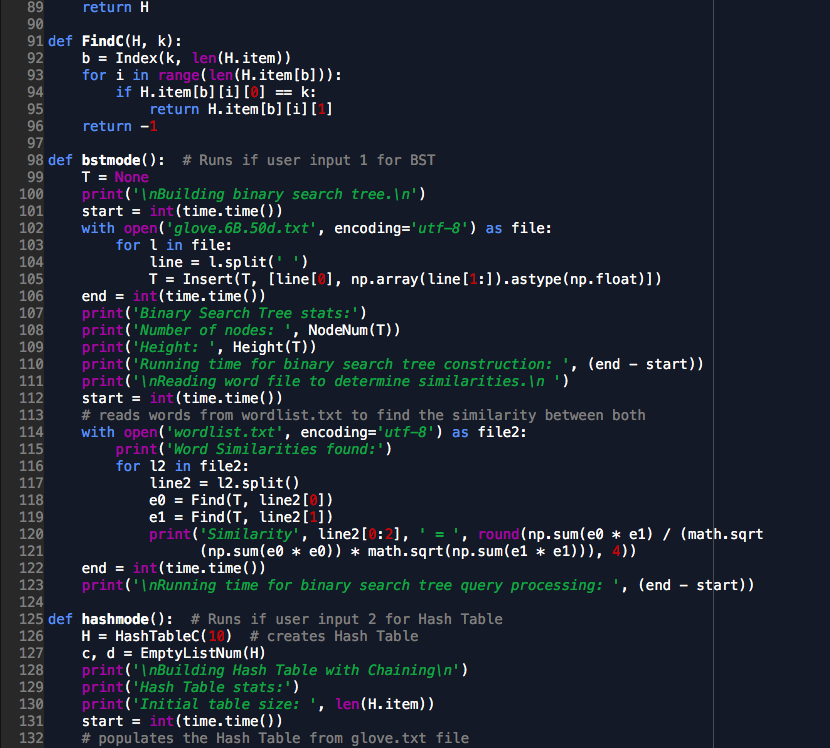
Screen Shot of the Hash Table stats:

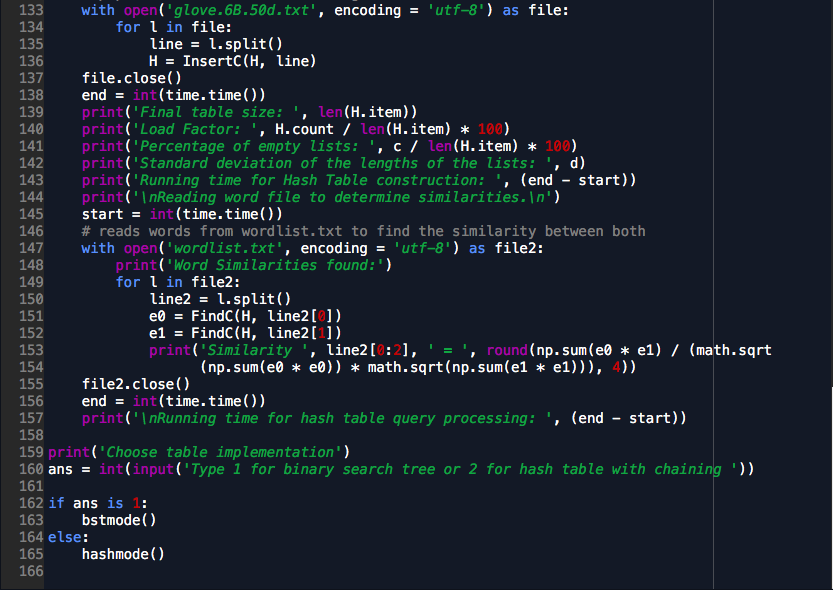


Appendix:









Academic dishonesty includes but is not limited to cheating, plagiarism and collusion. Cheating may involve copying from or providing information to another student, possessing unauthorized materials during a test, or falsifying data (for example program outputs) in laboratory reports. Plagiarism occurs when someone represents the work or ideas of another person as his/her own. Collusion involves collaborating with another person to commit an academically dishonest act. Professors are required to - and will - report academic dishonesty and any other violation of the Standards of Conduct to the Dean of Students.