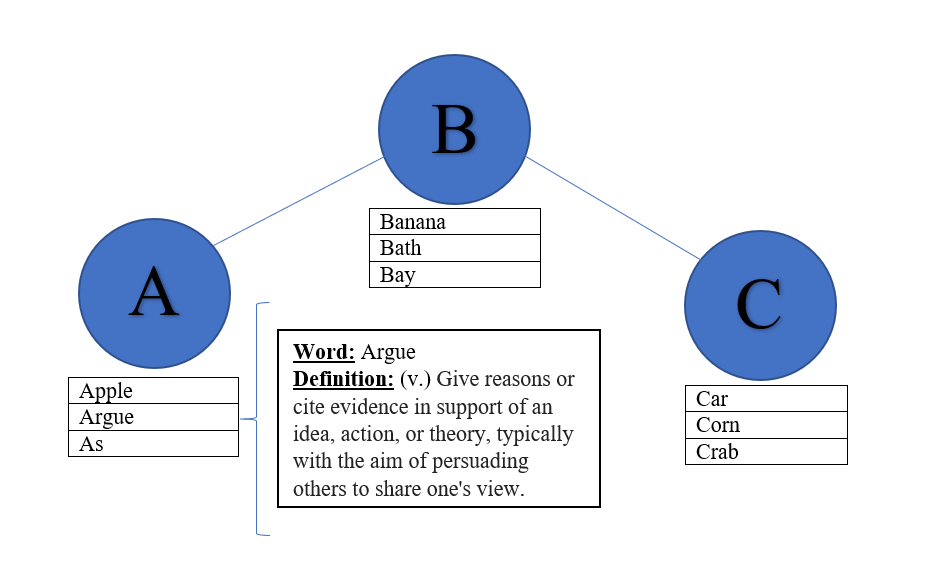
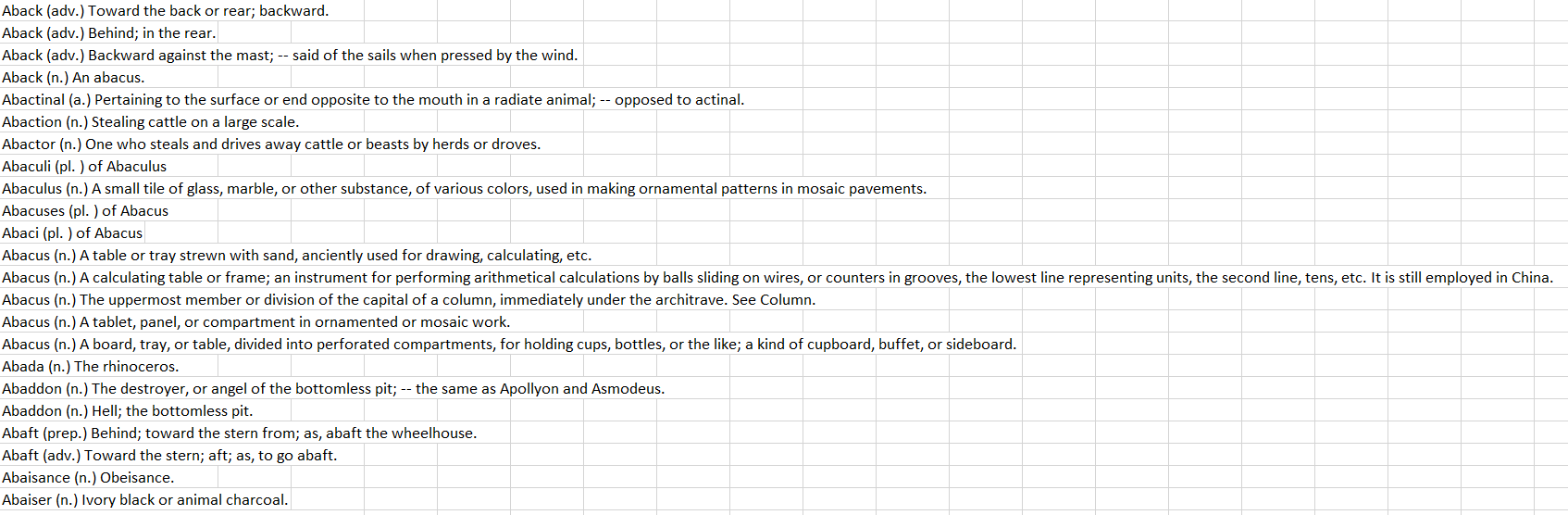
Trevor Hall  
CSCI 2270 – Zagrodski  
TA – Amit Kiran

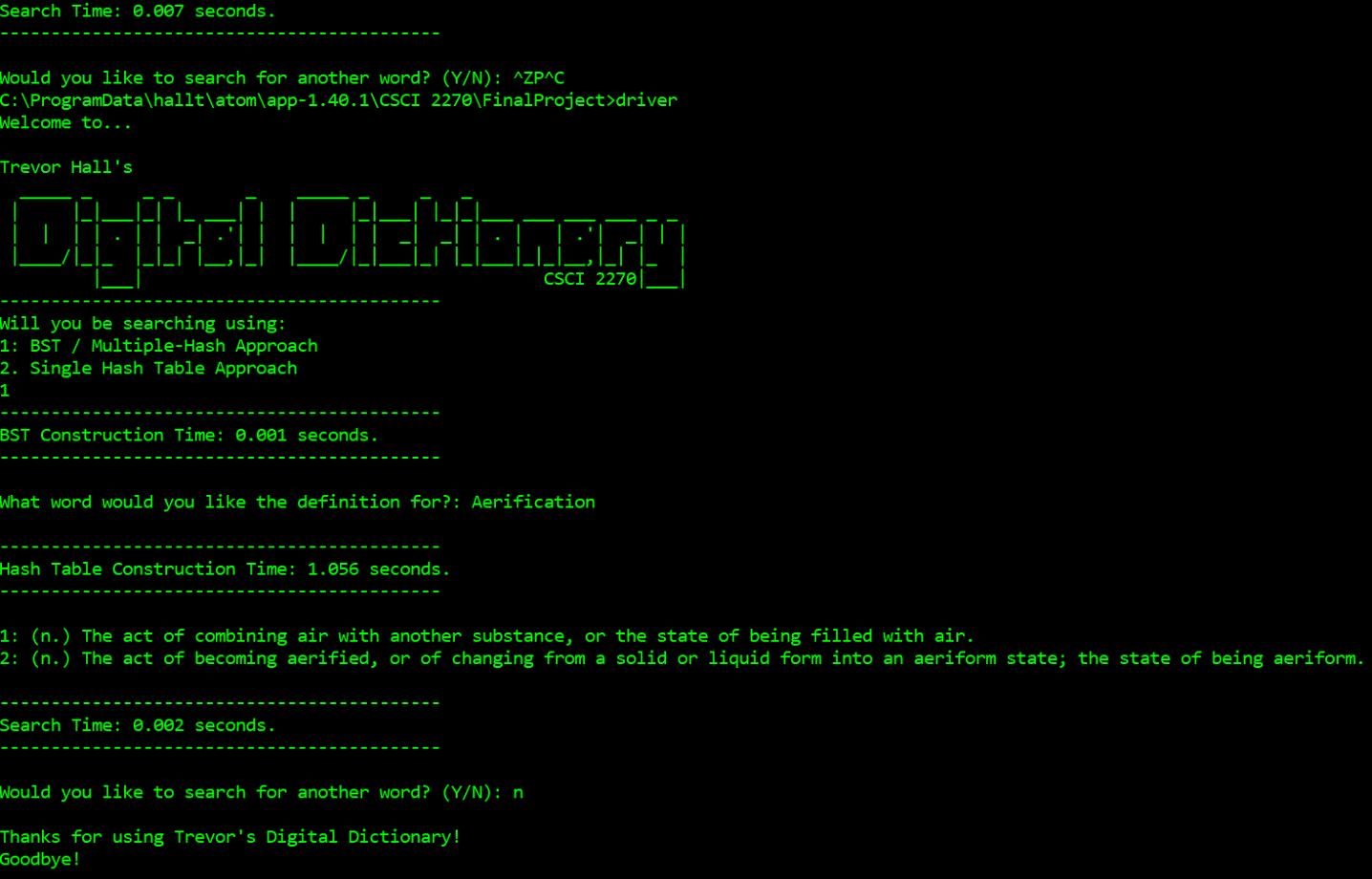
**Final Project – Dictionary**

For the CSCI 2270 Final Project, we were assigned to use data structures to solve a complicated problem. However, rather than to design a game or a system based on parameters I have established, I have chosen to design a project that would be practical for use in everyday life as a student: a Dictionary.

 **Data Structures:** While envisioning the practicality of a digital dictionary, I began to recognize that an extensive time requirement for building the dictionary may be impractical, as I find that dictionaries are rarely used to search more than a few words at a time. Therefore, the construction of a structure such as a single large hash table may prove difficult to justify for the purpose of the program. As such, I attempted to design a system which may minimize build time but also maintain ease and speed of search. As per a suggestion by my TA, I also added an option for the creation of a single large hash table and the ability to record times of operations such as build, search, etc. for comparison to my original methodology. The data structures I have chosen to use are a Binary Search Tree (for organization and compartmentalization of data), hash tables (for ease of search and defining of words) and linked list (chaining strategy for collision management). The exact utilization of each data structure will be more thoroughly explained in the methodology section.  
  
**Methodology:** My idea revolved around using a binary search tree to organize the data into components that can be more easily built and searched through. The BST will consist of 26 nodes, each one corresponding to a letter of the alphabet. Each node in the binary search tree holds information crucial for building a hash table of words beginning with only that letter (and int representing table size, string for name of the .csv file to pull data for that letter from, etc.). For instance, if the user looks to define “Data”, the BST will be searched to find the node representing the letter “D”, and thus will be able to construct a hash table that is – theoretically – 1/26 of the size of a hash table containing all words. As this reduces the build time, the dictionary becomes much faster and more practical.

 **Data:** In order to construct a dictionary containing every word in the English language and their definitions, there will obviously be a need for massive amounts of data. As such, I was able to compile 26 individual .csv files containing the words and definitions for each letter of the alphabet. However, the words and their definitions were not separated into individual cells. I have thus designed a function to separate the words and their definitions into separate strings, each being held by “word” structs for ease of search and access.

**Results:** As a result of the creation of the project, I now have access to a fast, efficient way of accessing words and their definitions. I will now include images containing an average output of the dictionary using each methodology.

****

****