Gabriel Arnell

Lab 6

Crafting a Compiler
8.1 (binary search trees and hash tables)
The two data structures most commonly used to implement symbol tables in production compilers are binary search trees and hash tables. What are the advantages and disadvantages of using each of these data structures for symbol tables?

Binary Tree:

Advantages: The binary tree is a well known and widely used algorithm that can have a good performance for setting and getting entries on average of O(log n) time. It is simple and efficient on average.

The disadvantage to the binary tree is that the average case of O(log n) time is not actually the average when it is compiling code is written by programmers because the identifier names are not chosen at random. This can throw the efficiency completely off.

Hash Table

Hash tables advantage lay in their performance which does not particularly require the programmer to be careful with their variable names like the binary tree because all of the inputs gets ran through a hash function which should remove any near similarities anyway.

A downside to using a hashtable is that if the hash's collision accidentally allows an overwrite, the compiler can spit out incorrect results. The hashtable also requires much more space to fit all the possible entries in than the binary tree.