**Documentation**

**Github Link**:

<https://github.com/raresica1234/LFTC/tree/master/Lab2>

**Hash Function**

Computes the hash of a Value which can be either an identifier, a string constant, boolean constant or integer constant. If it’s a constant that isn’t a string, then it is converted into a string and then it returns the hash code for that. To compute the hash code of a string I just sum the letters. The hash then becomes the computed hash modulo capacity of the list.

**Add Function:**

The add function first calls checkSizeAndGrow function, then it increases the size. After that it gets the position which is just calling the hash function. And then adds it to the list at that position. The collisions are solved by using a list at the place where the object is supposed to be added. That way when the object is added, it just appends to an already existing list. However this also means that to sample the SymbolTable we need to have a Pair as a position.

**CheckSizeAndGrow Function:**

The function divides the size by the capacity and compares it with a predefined alpha to see if the hash table is supposed to grow or not. If it’s supposed to grow then it saves a copy of the current list, recreates the list, and rehashes everything. The rehashing Is simply done by readding all of the elements again in the list.

**Find Function**:

The find function works by first hashing the token, then looping through all of the elements in the list at that position until the specified token is found and then returning the position as a pair which specifies the hash position, and then the position in said collision list.