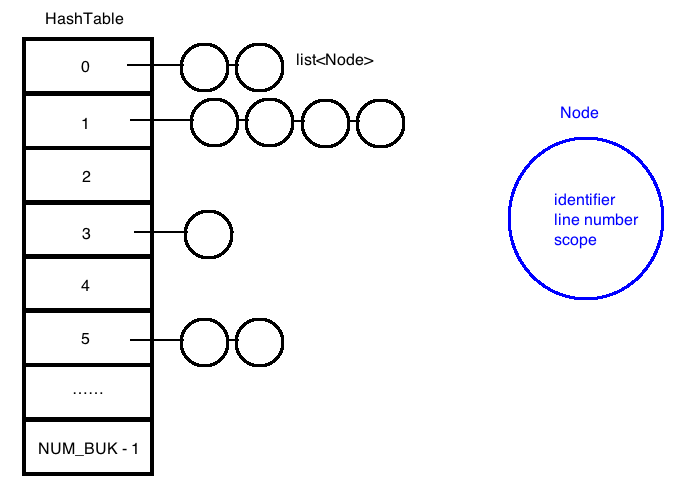
CS 32 Project4 Report

Yang Li 104138705



I introduced the struct node to save the identifier’s value, line number and the scope number in which it is declared.

I implemented the SymbolTableImpl with a HashTable of list<Node>.

I choose this type of container because it is very cheap in terms of time complexity.

Time complexity of functions:

enterScope: O(1) constant time. Independent of the number of items.

exitScope: O(N) since the number of buckets is constant, Big O is linear to the

number of Node in the list.

Declare: O(N) linear to the number of Node in the list

Find: O(N) linear to the number of Node in the list

Pseudocode

hashFunc:

Traverse throuth the characters in the string

add the Ascii number to total

exitScope:

If current scope greater than zero

Traverse through the hashtable array

if not empty

traverse through the list

if the scope of this Node is equal to the scope we are exiting

delete the Node

decrement the current scope

declare:

map the index of idenfier

traverse through the list in this index

if same identifier is already declared under current scope

return false

push the identifier at the front of the list

find:

map the index of idenfier

traverse through the list in this index

if same identifier is found under current scope

return line number

return -1

One notable problem I encountered when I wrote the code was deleting Nodes while looping through the list. At first I did not know how I should deal with the iterator pointing to the Node which is to be deleted. After I tried several tests, I found out that the iterator which used to point to the deleted Node is now pointing to the next Node(the one right after the deleted one). And I can still use this same iterator to continue looping through the list and visit every Node following the deleted one. In other words, I can loop through the list and delete Nodes without introduce a second iterator.