Node:
    Person's SSN
    Person's name
    Pointer to next value
SLL():
    Set size to 0
    Set headPtr to null.
~SLL():
    Traverse from front to back of the linkedlist
    Delete each node
SLL::getHeadPtr():
    Return the headPtr
SLL::insert():
    Set the headptr
    Find the last node in the list
    Create a new node
    Append the new node to the end of the list
SLL::search():
    If item1 is found,
        Return the pointer to the node
    Return null
SLL::erase():
    If empty:
        Return false
    Traverse the list until the item is found
    Remove the first node, middle node, then last node

SLL::display():
    Traverse the list
    Print each value to the console

HashTable() :
    Initialize instance variables: tableSize to 3 and numNodes to 0.
    Create dynamic array of type SLL to store data of size tableSize
HashTable(size):
    Initialize tableSize with parameter size
    Set numNodes to 0
    Create dynamic array of type SLL to store data of size tableSize
HashTable()::find():
    Search item in the table
    If item is found
        Return true
    Otherwise
        Return false

HashTable()::insert():

Insert (item1, item2) to the table

Use item1 as the key

If inserted

return true

Otherwise, return false

HashTable::erase():

Delete the pair whose key value is item

If deleted, return true

Otherwise, return false

HashTable::getSize():

Return the number of nodes in the table