# Major 2 Data Structures Checklist

## User Input Requirements

- The only user input for this assignment is what name the user wants to search for

## Programming / Functionality Requirements

- Create a structure and I called It NameNode instead of 'WordNode', it should have an array of char length 21 and a pointer to next
- Get input (names) from a file called names.txt
- In a loop put these names into a Hash Table that has 127 buckets and a very long linked list
- Each time you get a name pass it into a function called myHashFunction that gets the hash value from the djb2 hash function created by Dan Bernstein and remember to say where you got it from. Also you need to do the division method on that hash value to then return a value from 0-126 inclusive from the myHashFunction function.
- Make sure the lists are SORTED Linked lists, sorted in ascending order

- Once the user is done then set up a separate loop to search in the specific buckets.

-You must create a function called searchLinkedList that searches a sorted linked list for a name and returns a pointer to the node containing the name (if found) or NULL.

- This function must return NULL immediately after you pass the point in the linked list where the name would have been found if it was in the linked list.
- The function takes three parameters:
  - char * searchName (or you can use a string object): name to search for (entered by the user)
  - struct linkedList * linkedList: linked list to search (in your program, you can call the linked list node struct anything that makes sense)
  - int * comparisonCount: pointer to int filled in by the function with the count of strcmp comparisons done in searching the linked list

-Create a search function called searchForNameTwice. It returns nothing and will have the following parameters:

- char * searchName (or you can use a string object): name to search for (entered by the user)
- struct linkedList * linkedList: linked list to search

- struct linkedList * hashTable[]: hash table to search
- int comparisonCount[2]: array containing the count of strcmp comparisons done in searching the extremely-long sorted linked list (element 0) and in searching the hash table (element 1)

-It must call your linked list search function. It then displays one of the following messages once the search is done:

- "name was found in the list in number comparisons", where name is the name being searched for, list is either "linked list" or "hash table bucket" and number equals the number of times that strcmp was called
- "name was NOT found in the list in number comparisons"

-You will use this search function to search the hash table bucket and the sorted linked list.

- Don't worry about the grammatical inconsistency of possibly displaying "1 comparisons".
- Indent the message displayed by one TAB ('\t').

- Comment as normal
- Free the memory when done being used
- Do not use calloc()
- Do not have any other input or output

## Commenting

- Comment where you think necessary
- Function headers
- Main Header

## Style

- Make sure to go by my naming style
- Make sure to indent properly

- Make sure to space everything out properly and the same to make it look neat, I always do 4 spaces between functions and above main I usually use like 3

## File Naming

- Project is called dsA2
- Source file is called dsA2.cpp
- Checklist as well! Make sure the checklist is named “checklist.pdf” or check to see if it changed
- Zip file called dsA2.zip

## Things Required in all assignments

- Return 0 in main()
- Braces around single-line bodies for if, while and for… statements.
- No Global Variables
- Use Magic Numbers where applicable

## Submitting

- Clean solution!!!!
- Delete vs.
- Zip the folder
- Make sure the zip isn’t unnecessarily large