**EECS2040 Data Structure Hw #6 (Chapter 7 Sorting, Chapter 8 Hashing)**

**due date 6/27/2021**

**Part 2 Coding**

You should submit:

(a) All your source codes (C++ file).

(b) Show the execution trace of your program.

1. (50%) Sorting:

Write a C++ program to perform 5 different sorting , insertion sort, median-of-three quick sort, iterative merge sort, recursive merge sort, and heap sort, on lists of characters, integer, floating point numbers, and C++ strings.

1. You need to write the 5 sorting function templates (refer to example programs in textbook or pptx)
2. Randomly generate a list of 20 characters as an input unsorted list.
3. Randomly generate a list of 20 integers as an input unsorted list.
4. Randomly generate a list of 20 floats as an input unsorted list.
5. Randomly generate a list of 20 string objects as an input unsorted list.

Show your results using the above 3 lists in your program.

1. (50%) Hashing:

Write a C++ program to implement two simple symbol tables (dictionaries) using hash table with linear probing for collision and hash table with chaining. For simplicity,

1. Consider storing only the key (need not consider the (key, value) pair) in the symbol tables.
2. Furthermore, the key is a variable-length character array with the first character of the key is an alphabet, e.g., abc, abcde, b, bye, cool,…
3. Consider a simple hash function using only the first character of key to hash, so h(abcde) = h(abc), h(b) = h(bye),.., etc. Therefore, collision can happen frequently.
4. The initial hash table size can be set to 26 since we have 26 alphabets which are the hashed keys.

Create 2 symbol table classes for linear probing and chaining, respectively. Both must implement at least the following functions:

Constructor,

Insert(key)

Search(key)

You may add other functions needed in your program.

Your main function may contains code like:

SymbolTable1 d1;

Setup at least 10 key objects

Insert those 10 keys into d1.

Display d1

Demo the search function of d1 (try at least 5 keys)

SymbolTable2 d2;

Setup at least 10 key objects

Insert those 10 keys into d1.

Display d2

Demo the search function of d2(try at least 5 keys)