CMPSC 412 – Lab – 4 & 5 (50 points)
Searching and Sorting

Due date: in 2 weeks

Note: attach screenshots of your program and results under each programming exercises. Please make sure that the screenshot is readable. Don't attach a very small screenshot image.

Lab Exercises:

1. Implement and use binary search to make a "guess-the-number" game: You think of a number between 1 and 10000, your program guesses the number and you reply whether your guessed number is higher or lower. Your program makes another guess and so on until it gets the right number. Calculate the actual memory size for each of the data structures, variable you use in your program. You can use __sizeof__() or sys.getsizeof().
   Example: print (a.__sizeof__())
   print (sys.getsizeof(a))

```
➜ Lab4N5 python main.py
Think of a number between 1 and 10,000
The number in your head is `2194` is this right? (Y/N) n
Is the number I guess lower or higher than your number?(lower, higher) lower
The number in your head is `3026` is this right? (Y/N) y
Total size of all the variable 188 bytes
```

2. Create a database with the following details for at least 20 students and store it as a text file:

   Student_ID first_name last_name email_id major

   Example:
   923329923 Jake Ryan jryan@psu.edu CMPAB_BS

- Write a program to read the data from the text file. Choose an appropriate data type and data structure (list, lists of list, dictionary) for storing the information in your program.

- Write a function which takes a parameter and sorts the entire list of students and displays all the details of all students. Your function should sort the text file using a) student id and b) first name, and save it in a different text file. Implement the sorting using selection sort, insertion sort, bubble sort, and merge sort. Print out how much cpu time it took to sort the data for each sorting algorithm. You can import a library to calculate the time. Calculate the actual memory size for your program.

- Table-1: Tabulate your recorded time for all the four sorting algorithms i.e., selection sort, insertion sort, bubble sort, and merge sort. Sorting according to a) student id and b) first name.

- Table-2: Tabulate the memory size for all the four sorting algorithms i.e., selection sort, insertion sort, bubble sort and merge sort. Sorting according to a) student id and b) first name.

- Write a conclusion paragraph about what you understood about the searching and sorting algorithms.

```
➜ Lab4N5 python main.py
Id Sort:
    bubble_time = 3.672399998322362e-05 seconds | Memory Peaks: 17314 Bytes
    insertion_time = 1.924599973790464e-05 seconds | Memory Peaks: 18942 Bytes
    selection_time = 3.788800040638307e-05 seconds | Memory Peaks: 20722 Bytes
    merge_time = 6.90900001245609e-05 seconds | Memory Peaks: 24726 Bytes
------------------------------
First name Sort:
    bubble_time = 7.363900022028247e-05 seconds | Memory Peaks: 24351 Bytes
    insertion_time = 3.7257999792927876e-05 seconds | Memory Peaks: 25131 Bytes
    selection_time = 6.667600018772646e-05 seconds | Memory Peaks: 25943 Bytes
    merge_time = 7.621600025231601e-05 seconds | Memory Peaks: 28275 Bytes
```

Conclusion:

For ID sort, all of it's result take less time than first name sort, this is because of how I implemented the data structure, I uses nested dictionary with the structure of

```
'id' : {
        'name': name,
        etc...
        }
```

because of this for first name sort I have to access the 2nd layer of the dictionary to sort therefore it take more times.

Because of the small sample size (20) merge time look like the most inefficient time wise, which is true for small sample size such as this situation.

For memory size it make sense that merge sort requires more memories, because merge sort is recursive which make it call itself which would requires some memory overhead, it also returns and creates multiple lists during it's call so that also contribute to it's higher memory usage compare to other algorithm.