# CSE 1321L: Programming and Problem Solving I Lab

## Lab 9

## Searching and Sorting

What students will learn:

- How to implement searching and sorting algorithms

Overview: If you were given a list of names and asked if your name was on the list, how would you know? How you choose to search or sort data may be difficult for you to explain to a computer. For example, for searching, you could start at the top and work your way down to the bottom of a list. If the list is small, that is very manageable. However, if the list contains all the students at KSU or all the people in Georgia, it may take a little while and be inefficient. If this list is randomly ordered, it may make sense for you to first sort the names in alphabetical order. How you choose to sort and search your lists will ultimately depend on the application situation and time efficiency.

### Searching

There are many ways to search through data, but this lab focuses on 2, linear search and binary search.

a) Linear search:
   1. Start at the first element/cell of a list
   2. Check if its value matches or does not match the target value
   3. If it does match, stop. If it does not match move on to next element/cell
   4. Repeat from step 2 until the end of the list

b) Binary search:
   1. Sort the list – this is a requirement for binary search
   2. Find the middle element/cell of the list (list length / 2)
   3. Compare target value with the value in the middle element/cell
   4. If the value is less than the value in middle element/cell
      - Search the lesser (left) half and no longer consider the upper (right) half
   5. Else
      - Search the upper (right) half and no longer consider the lesser (left) half
   6. Repeat from step 2 until you find the target

You might think that linear search is faster. However, this is not always the case. If the target is the first item in the data set then yes, but you should think about in an average case. On average, linear search is much slower compared to binary search. We often use terms such as O (n) (read as big “O” of “n”). This accounts for the average time complexity of a searching or sorting algorithm.

### Sorting

Just as there is more than one way to search, there are many ways to sort data. The efficiency also changes dramatically depending on the algorithm used, and the correct sorting method will change based on the application. Remember, there is never a BEST overall sorting algorithm, but rather a BEST sorting algorithm for this application.

For this lab, please reference slides found on the FYE website for specifics with how to these searching and sorting functions work:
https://ccse.kennesaw.edu/fye/1321_labs_assignments.php

Lab9A: Warmup.
Write a program that uses an array (of size 10) to demonstrate how to use the linear search algorithm to search through a user generated list of numbers for a specific value. Assume that the numbers in the list will be integers from between -100 and +100. You should store the numbers in a 1D array. The logic and print statements declaring whether the target is or is not in the set of numbers should also be located in your main method.

Tip: Make sure to check out the slides in the link above for the algorithm in your language.

Remember, the file name should be Lab9A.
The user input is indicated in bold.

Sample output #1

```
Please enter 10 numbers:
Integer 1: 15
Integer 2: 12
Integer 3: 89
Integer 4: -14
Integer 5: 11
Integer 6: -99
Integer 7: 1
Integer 8: 42
Integer 9: 27
Integer 10: 2

What is the target number: 42

The target is in the set.
```

Sample output #2

```
Please enter 10 numbers:
Integer 1: 15
Integer 2: 12
Integer 3: 89
Integer 4: -14
Integer 5: 11
Integer 6: -99
Integer 7: 1
Integer 8: 42
Integer 9: 27
Integer 10: 2

What is the target number: -3

The target is not in the set.
```

Lab9B: Binary Bubbles.
Binary search is a very fast searching algorithm, however it requires a set of numbers to be sorted first. For this lab, create an array full of 11 integers which the user will generate. Like in the previous lab, assume that the values will be between -100 and +100. Then, using the sorting algorithm called BubbleSort, put the array in the correct order (from lowest to highest number). After this, please print the array to the screen. Finally, search the array for the target value using Binary Search.

The BinarySearch code will implement the algorithm described in the lab slides. During this, you should print out a few key values which help Binary Search function. For example, this algorithm focuses on a low, mid, and high which correspond to the indices in the array the algorithm is currently considering and searching.
Printing these values during the search process will help with debugging and fixing any issues.

- BubbleSort sorts the array to prepare for the next step
- BinarySearch searches the now sorted array to determine if the target value is in the array or not

Remember, the file name should be Lab9B.
The user input is indicated in bold.

Sample output #1

```
Please enter 11 numbers:
Integer 1: 15
Integer 2: 12
Integer 3: 89
Integer 4: -14
Integer 5: 11
Integer 6: -99
Integer 7: 1
Integer 8: 42
Integer 9: 27
Integer 10: 2
Integer 11: 67
What is the target number: 42
The sorted set is: -99 -14 1 2 11 12 15 27 42 67 89
Low is 0

High is 10

Mid is 5

Searching

Low is 6

High is 10

Mid is 8

Searching

The target is in the set.
```

Sample output #2

```
Please enter 11 numbers:
Integer 1: 15
Integer 2: 12
Integer 3: 89
Integer 4: -14
Integer 5: 11
Integer 6: -99
Integer 7: 1
Integer 8: 42
Integer 9: 27
Integer 10: 2
Integer 11: 67
What is the target number: -5

The sorted set is: -99 -14 1 2 11 12 15 27 42 67 89

Low is 0

High is 10

Mid is 5

Searching

Low is 0

High is 4

Mid is 2

Searching

Low is 0

High is 1

Mid is 0

Searching

Low is 1

High is 1

Mid is 1

Searching

The target is not in the set.
```

Instructions:

- Programs must be working correctly.
- Programs must be saved in files with the correct file name.
- If working in Java or C#, class names must be correct.
- Programs (only .java, .cs or .cpp files) must be uploaded to Gradescope by due date.