CS 321 Data Structures
Summary of Searching and Sorting Algorithms

- **Search**

Searching Algorithm	Worst-case runtime	Average/expected-case runtime
Linear search (for unsorted data)	$\Theta(n)$	$\Theta(n)$
Binary search (for sorted data)	$\Theta(\lg n)$	$\Theta(\lg n)$

- **Notes on sorting**

1. Insertion sort, Merge sort, Heap sort, and Quick sort are based on using comparisons whereas the Counting sort, Radix sort, and Bucket sort use properties of the keys being sorted as decsribed below:
   - Counting sort assumes that the n input numbers belong to the set $0, 1, 2, \ldots, k$
   - Radix sort assumes there are n numbers, each number has d digits, and each digit can take up to k possible values.
   - Bucket sort assumes that the n numbers are uniformly distributed in the half-open interval $[0, 1)$.
2. A sorting algorithm is **in-place** if at most a constant number of elements of the input array are ever stored outside the array – which helps with space efficiency. Note that recursion requires space for the call stack, but we are not considering that in this case as it is usually a small amount of space.
3. A sorting algorithm is **stable** if two equal elements retain their original order in the sorted output. Any sorting algorithm can be made to be stable at some extra cost in space but Insertion sort, Merge sort, Counting sort, Radix sort, and Bucket sort are stable in their basic form. Heapsort, Quicksort, and Selection sort are not stable but can be made stable at extra cost in space.
   - Insertion Sort: stable (if no "=" sign in comparison)
   - Selection Sort: stable (if no "=" sign in comparison)
   - Merge Sort: stable (if the "=" sign is in the comparison)
   - Heap Sort: not stable (exchange $A[1] \longrightarrow A[n]$)
   - Quick Sort: not stable.
     Ex: input: $< 5, 5', 5'', 3, 4 >$ and the output is $< 3, 4, 5'', 5, 5' >$.
   - Bucket Sort: is stable if the subsorting algortihm used to sort buckets is stable.

   Problem 3(b) on the sample exam shows how to make any sorting algorithm stable by sing extra space.

- **Summary of sorting algorithms**

Sorting Algorithm	Worst-case runtime	Average/expected-case runtime	in-place	stable
Insertion sort	$\Theta(n^2)$	$\Theta(n^2)$	yes	yes
Selection sort	$\Theta(n^2)$	$\Theta(n^2)$	yes	yes
Merge sort	$\Theta(n \lg n)$	$\Theta(n \lg n)$	no	yes
Heapsort	$\Theta(n \lg n)$	(not studied)	yes	no
Quicksort	$\Theta(n^2)$	$\Theta(n \lg n)$ (expected)	yes	no
Counting sort	$\Theta(n + k)$	$\Theta(n + k)$	no	yes
Radix sort	$\Theta(d(n + k))$	$\Theta(d(n + k))$	no	yes
Bucket sort	$\Theta(n^2)$	$\Theta(n)$	no	yes