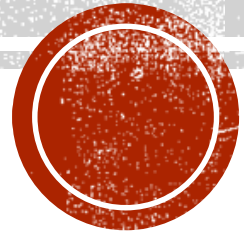
CS 101 LAB #7
SORTING AND SEARCHING ARRAYS



Mythili Vutukuru
IIT Bombay

Reference: "C How to Program", Deitel and Deitel, 8th Edition, Chapter 6

GOAL OF LAB #7

- In this lab, you will write programs to understand the following concepts:
  - Sorting arrays
  - Searching arrays
- Submit any two programs testing any two different concepts
- Please write code neatly, with proper indentations and comments

SORTING ARRAYS

- Write a program that generates an array of 1000 random integers between 0 and 100. Sort the array using bubble sort. Use the sorted array to compute the 10^{th}, 25^{th}, 50^{th}, 75^{th}, and 90^{th} percentile values in the set.
- Read up about the insertion sort algorithm online and implement it. Test it by sorting an array of 10 random integers in ascending order.

6.11 *(Selection Sort)* A *selection sort algorithm* for a one-dimensional array has the following steps:

a) The smallest value in the array is found.
b) It is swapped with the value in the first position of the array.
c) The above steps are repeated for the rest of the array starting at the second position and advancing each time.

Eventually the entire array is divided into two parts: the sub-array of items already sorted which is built up from left to right and is found at the beginning, and the sub-array of items remaining to be sorted, occupying the remainder of the array. Write a program that sorts an array of 10 integers using this algorithm.

SEARCHING ARRAYS

6.32 *(Linear Search)* Modify the program of Fig. 6.18 to use a recursive `linearSearch` function to perform the linear search of the array. The function should receive an integer array, the size of the array and the search key as arguments. If the search key is found, return the array index; otherwise, return –1.

6.33 *(Binary Search)* Modify the program of Fig. 6.19 to use a recursive `binarySearch` function to perform the binary search of the array. The function should receive an integer array, the starting index, the ending index and the search key as arguments. If the search key is found, return the array index; otherwise, return –1.