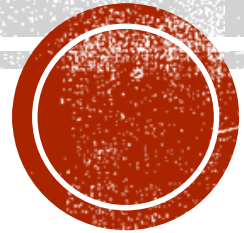
CS 101 LAB #8

POINTERS & STRINGS



Mythili Vutukuru

IIT Bombay

Reference: "C How to Program", Deitel and Deitel, 8th Edition, Chapter 7,8

GOAL OF LAB #8

- In this lab, you will write programs to understand the following concepts:
  - Accessing and manipulating variables using pointers
  - Accessing and manipulating strings/arrays using pointers
  - Writing recursive functions on strings
- Submit any two programs testing any two different concepts
- Please write code neatly, with proper indentations and comments

PRACTICE WITH POINTERS

- Write a program with a function to find the minimum and maximum of three distinct integers. The function should take pointers to the three integers, as well as pointers to min and max variables as arguments. The function must have a void return value.
- Write a program to print addresses of all elements of an array of size 10 integers
  - Are all elements of the array stored contiguously in memory? Can you confirm this fact from the addresses?
  - Are the addresses the same when you run the program multiple times on the same computer? What about when you run it on different computers?
- Write a program to compute the sum of all integers of an array by traversing the array using pointers.

STRING MANIPULATION USING POINTERS

8.6 *(Displaying Strings in Alternating Uppercase and Lowercase)* Write a program that inputs a line of text into `char` array `s[100]`. Output the line in alternate uppercase letters and lowercase letters.

8.10 *(Appending Part of a String)* Write a program that uses function `strncat` to append part of a string to another string. The program should input the strings, and the number of characters to be appended, then display the first string and its length after the second string was appended.

8.28 *(Write Your Own String-Copy and String-Concatenation Functions)* Write two versions of each of the string-copy and string-concatenation functions in Fig. 8.14. The first version should use array indexing, and the second should use pointers and pointer arithmetic.

8.29 *(Write Your Own String-Comparison Functions)* Write two versions of each string-comparison function in Fig. 8.17. The first version should use array indexing, and the second should use pointers and pointer arithmetic.

8.30 *(Write Your Own String-Length Function)* Write two versions of function `strlen` in Fig. 8.33. The first version should use array indexing, and the second should use pointers and pointer arithmetic.

SORTING AND STRINGS

- Write a program that reads several strings from the user, sorts them in alphabetical order, and prints the output to the screen. You can assume that all strings contain only lower case alphabets, so simply comparing the numerical values of the characters is enough to sort them alphabetically. You can use any sorting algorithm of your choice.

RECURSION AND STRINGS

6.31 *(Palindromes)* A palindrome is a string that's spelled the same way forward and backward. Some examples of palindromes are: "radar," "able was i ere i saw elba," and, if you ignore blanks, "a man a plan a canal panama." Write a recursive function `testPalindrome` that returns 1 if the string stored in the array is a palindrome and 0 otherwise. The function should ignore spaces and punctuation in the string.

6.36 *(Print a String Backward)* Write a recursive function `stringReverse` that takes a character array as an argument, prints it back to front and returns nothing. The function should stop processing and return when the terminating null character of the string is encountered.