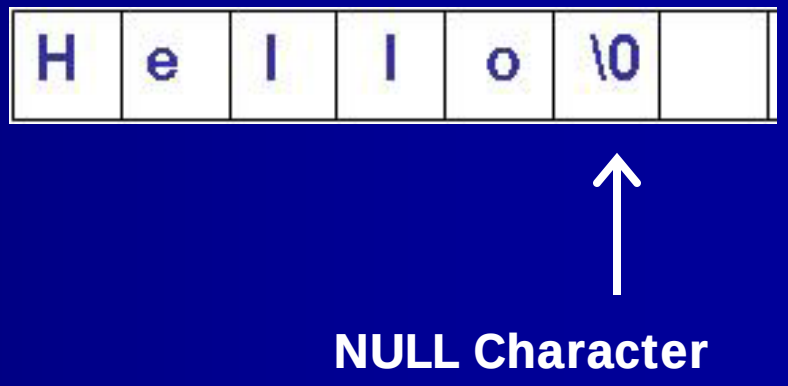
Strings and <string.h>

ESE 116, October 30 Fall 2007

What is a C String?

- A null – terminated character array (char*)
- Not a special data type or object, just a char*, uniquely formatted
- Not even remotely related to the Java String, which is a Object



How do you make a C String?

1. Declare a char array that you know is (max size of your text) + 1
2. To construct from characters, add characters to the array, then end with a '\0'
3. Any data after the '\0' is in the array, but not considered part of the string.
4. The above string is just "Hello". The spots after the '\0' are available, but not used

Other ways to make a C String

1. char test[] = “hello”;
2. char* test = “hello” (read only)
3. char test[100] = “hello”
4. char test[] = {‘h’, ‘e’, ‘l’, ‘l’, ‘o’, ‘\0’}

Examples 1, 2, and 3 add the NULL character to test[] automatically

Why are C Strings useful?

- The NULL termination means we can find the length of the string using a for loop
- As long as we NULL terminate every string we use and allocate big enough char arrays, we can deal with variable sized text.
- Opens up a new world of functional possibilities

The string.h library

- We are not the first people to find out that NULL termination is a good idea.
- The **string.h** library is an set of C functions devoted to NULL terminated char arrays
- All the functions in string.h accept pointers to C strings as arguments.
- They operate on them or give us the information we need to parse / control them

The big string.h functions

- Some of the most important string.h functions:
- int strlen(char* string)
  - Returns the string length (to ‘\0’)
  - NOT THE ARRAY LENGTH
- char* strcpy(char* dest, char* src):
  - Loops through src, copies to dest. Also returns dest

The big string.h functions ctd

- int strcmp(char* a, char* b)
- Compares the first character in a and b
- If equal, moves on to the next character.
- For first characters that are not equal,
  - returns a positive number if ascii value of a[0] is > ascii value of b[0]
  - returns a negative number if ascii value of a[0] is < ascii value of b[0]
- If all chars are equal, returns 0

Other string.h Functions

- char* strcat(char* dest, char* src)
  - Concatenates src onto dest
  - Dest must be large enough to hold src
- char* strncmp(char* a, char* b, size_t n)
  - Compare a and b for n characters
- char* strrchr(char* str, char c)
  - Locate last occurrence of c in str, return pointer
- Google string.h for the complete list of available functions

Bottom Line

- When in Rome, do as the Romans do
- When in C, use C strings to hold text
  - Extremely convenient
  - List of functions that can manipulate them easily
  - It is the standard
- Don’t forget: Array size must be (max string length + 1) (why?)
- When possible, use string.h functions instead of your own for loops