COP 3223H: Introduction to C Programming

Fall 2023

University of Central Florida

Dr. Kevin Moran

Week 11- Class III: Strings Part III

Administrivia

- SPA 3 now due on Mon. - Python script coming today
- SPA 4 and LPA 2have been released, are be due on November, 10th, and November 17th respectively.
- Exam grades will be released today!
- Mid-Semester Feedback Survey will be posted today.
  - Please complete to count as a quiz grade.

Today's Agenda

1. More on String Library Functions
2. Intro to Structs

Review

`fgets()`

- `fgets()` is similar to gets(), but with extra syntax.
- `fgets()` meets the possible that gets() raises.
- `fgets()` takes three arguments
  - Array
  - String Length Limit
  - File to read from (stdin which is standard input)
- `fputs()` works like **`puts()`**, except that it doesn't automatically append a newline

chomp()

```c
void chomp(char word[]){
    if(word[strlen(word) -1] == '\n')
        word[strlen(word)-1] = '\0';
}
```

The String library

- Strings has a library devoted to strings.
- The library contains a series of functions that can manipulate or access certain content about strings.
- All functions associated with strings are stored in the string header file (`string.h`)
- Since they are stored in separate header file, make sure to include it!!

```
#include<string.h>
```

The String library

Function	Stack Space
`strcpy()`	Makes a copy of source, a string, in the character array accessed by dest:
`strncpy()`	Makes a copy of up to n characters from source in dest: strncpy(dest, source, 5) stores the first five characters of the source and does NOT add a null character.
`strcat()`	Appends source to the end of dest: strcat(dest, source)
`strncat()`	Appends up to n characters of source to end of dest, adding the null character if necessary.
`strcmp()`	Compares s1 and s2 alphabetically. Returns a negative value if s1 should precede s2, a zero if strings are equal, and a positive value if s2 should precede s1 in an alphabetized list. strcmp(s1,s2)
`strncmp()`	Compares the first n characters in s1 and s2 returning positive, zero, and negative values like strcmp.
`strlen()`	Returns the number of characters in s, not counting the terminating null. strlen(s)
`strtok()`	Breaks the parameter string into tokens finding groups of characters separated by any of the delimiter characters. Each group is separated with '\0'.
`strchr()`	Returns a pointer to the first location of a character located in the string. Null is returned if character is not found.
`strpbrk()`	Return a pointer to the first location in the strings that holds any character found in another string.
`strchr()`	Returns a pointer to the last occurrence of a character in the string. Null is returned if character not found.
`strstr()`	Returns a pointer to the first occurrence of string s2 in string s1. Null is returned if character not found.

`strlen`

- The user defined function you just saw on the previous slide is already implemented in the `string.h` file.
  - Function takes one parameter which is the address of the string!
- It uses the same idea from our own custom function.
  - Start at the first address passed in the function.
  - Iterate through the string and count each character until the first null character is found.
  - Return the counter value.
  - IMPORTANT! The value return tells what index contains the null character!

strlen Example

```c
char word[100] = "Mondays";
int len = strlen(word);
```

len = 7;

`strlen` Example 2

```c
char word[100] = "Mondays";
word[3] = "\0";
int len = strlen(word);
```

len = 3;

`strlen` Example 3

```c
char word[100] = "Mondays";
word[2] = "\0";
int len = strlen(&word[3]);
```

len = 4;

strcpy

- The string library provides a function that allows you to completely copy the contents of a string into another including the null character (‘`\0`’).
- This is a very common task to do in many problems.
    - Hence why it even exists!
- The function takes two parameters.
    - The first parameter is the destination string (an address)
        - Where the contents copied need to be stored in memory
    - The second parameter is the source string (an address)
        - Where the contents that are needed to be copied are stored in memory
        - Important: You can also place a string literal has the source. This is the proper

strcpy Example

```
char string1[8];
char string2[8] = "Cakes";

strcpy(string1, string2);
strcpy(string1, "Cookies");
```

Something to Avoid

- Since we have learned that arrays are simply pointers, you might try to some sort statement like this…

```
char *string = "Hello!";
```

- This is a igning a pointer t to view it as a reac

- Now tha y to write the follov ment.

This code will crash!!

```
strcpy(string, "hi");
```

Substrings and `strncpy`

- Substrings are a fragment of a longer string
- strncpy is the function to use to generate substrings of a string
  - The function takes 3 parameters
    - The first parameter is the destination (address)
    - The second parameter is the source (address)
    - The third parameter is the number of characters (integer)
- Examples
  - String called “Andrew”
  - Substring of this is “And”
  - Substring of this is “drew”
  - “Adw” is NOT a substring!!

`strcat`

- Concatenation is taking two strings and joining them together as one string. It basically appending one string to the end of the another one.
- Example: “Progr” concatenated with “amming” would be “Programming”
- `strcat` and strncat are the string functions that handle concatenation
    - strcat appends an entire string (simply copies the entire source string)
        - First argument is the destination string (address)
        - Second argument is the source string (address)
    - `strncat` appends the first n characters of a string (handles the null character properly)
        - First argument is the destination string (address)
        - Second argument is the source string (address)
        - Third argument is the number of characters (integer)

strcat

```c
char string5[8] = "Vanilla";
char string6[8] = "Cookie";

strcat(string5, string6);

printf("string5 = %s\n", string5);
printf("string6 = %s\n", string6);
```

`strcmp`

- Programmers can compare strings to determine if they are a match.
- The string library has two comparison functions we can use to properly compare all the characters of each string.
- `strcmp`
    - First parameter is the first string
    - Second parameter is the second parameter
- `strncmp`
    - First parameter is the first string
    - Second parameter is the second parameter
    - Third parameter is the first n characters to compare
- Resulting Value Meaning
    - Negative number if first string comes first
    - Zero if both strings are EXACTLY the same
- Positive number if the second string comes first

Understanding String Comparison Results

- The strcmp function loops through each string simultaneously and computes the difference of each ASCII value. If the result is 0 (meaning the characters are the same), the function will traverse the strings until 1 of 2 things can happen.
    - A nonzero value is computed.
    - Reached the end of one of strings (not all strings compared are the same size)

strcmp

```c
char string7[8] = "red";
char string8[8] = "blue";

int result = strcmp(string7, string8);

printf("result = %d\n", result);
```

Acknowledgements

Slides adapted from Dr. Andrew Steinberg's COP 3223H course