Characters and Strings in C

Friday, September 25, 2020 11:11 AM

Topic

How can a computer represent and manipulate more complex data like text?

Char

Defn: Variable that represents single glyph

Storage: As integers via ASCII conversion (8 bit unsigned)

```
char a = 'A';   // actually 65
char b = '\n';  // actually 10
```

(note: can bit flip upper/lowercase w/ 32's bit)

Strings

Defn: Set of characters

Storage: Array of characters w/ null terminator ('\0') at the end (i.e., char* or char[])

ex. "Hello"

idx	0	1	2	3	4	5
char	'H'	'e'	'l'	'l'	'o'	'\0'

strlen — calculates length, does not include null terminator

this is O(N) operation!

Functions

in <string.h>

all expect valid string input (i.e., '\0' at end)

Comparison. Standard method str1 $\gtrless$ str2 compares memory addrs!

Instead, use strcmp()

```
strcmp(str1, str2) == 0  // identical
                   < 0   // str1 before str2
                   > 0   // str2 before str1
```

Copying. str2 = str1 copies memory addrs!

Instead, use strcpy() or strncpy()

```
strcpy(char* dest, char* src)
strncpy(char* dest, char* src, int extent)  // does not copy '\0' if not in extent
```

Need to make sure there exists space in dest to avoid buffer overflow

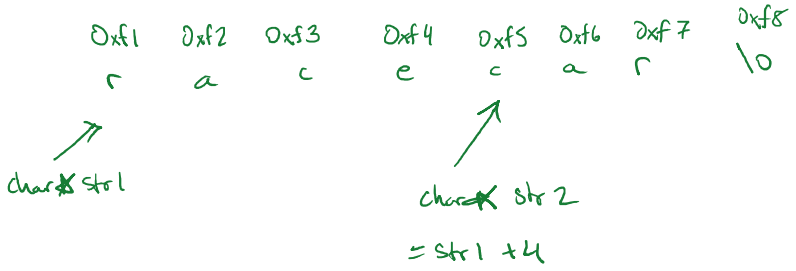
Concatenate str1 + str2 adds mem. addresses!

Instead, use strcat() or strncat()

```
strcat(char* dest, char* src);
strncat(char* dest, char* src, int extent);
// both ensure null terminators
```

Substrings. Use pointer arithmetic for end substr

ex.



// note: modifying 1 modifies both

strncpy for begin substr

ex.
```
char str2[5];
strncpy(str2, str1, 4); // "race"
str2[4] = '\0';
```

Combine for middle substr

ex.
```
char str3[4];
strncpy(str3, str1+1, 3);
str3[3] = '\0';
```