Day15 A

Young W. Lim

2017-12-09 Sat

Outline

Based on

"C How to Program",
Paul Deitel and Harvey Deitel

A. Character Handling Functions

isdigit	is the character a digit?
isalpha	is the character an alphabet?
isalnum	is the character an alphabet or a number?
isxdigit	is the character a hexa digit?
islower	is the character a lowercase letter?
isupper	is the character a uppercase letter?
tolower	convert to a lowercase letter
toupper	convert to a uppercase letter
isspace	is the character a white space?
iscntrl	is the character a control character?
ispunct	is the character a punctuation character?
isprint	is the character a printable character?
isgraph	is the character a grphic character?

B. String Conversion Functions

strtod	string to double
strtol	string to long
strtoul	string to unsigned long

C. Standard Input/Output Library Functions

fgets	get a string
putchar	put a character
getchar	get a character
sprintf	printf to a string
sscantf	scanf from a string

D. String Manipulation Functions

strcpy	string copy
stcncpy	string copy only n characters
strcat	string concatenate
strncat	string concatenate only n characters

E. String Compare Functions

strcmp	string compare
strncmp	string compare only n characters

F. String Search Functions

strchr	search a character in the string (1st occurrence)
strrchr	search a character in the string (last occurrence)
strpbrk	the break point with the characters in the string
strspn	the length of the span starting from s1 consisting of only characters in s2
strcspn	the length of the span starting from s1 consisting of characters{not in s2
strstr	search a substring in the string
strtok	break the string by the token (delimeter)

strchr, strrchr : r reverse

stspn, strcspn : c complementry strspn

strpbrk : string pointer break

G. Memory Functions

memcpy	copy memory elements
memmove	move memory elements
memchr	character in memory elements
memset	set memory elements with the character

- memmove
  - using a temporary buffer while copying
  - can handle overlapped src and dst memories
  - a little bit slower than memcpy

H. Other String Functions

strerror
strlen

int isspace (int c);[1]

- to check whether c is a *white-space character
- for the "C" locale, white-space characters are any of:

' '	(0x20)	space (SPC)
'\t'	(0x09)	horizontal tab (TAB)
'\n'	(0x0a)	newline (LF)
'\v'	(0x0b)	vertical tab (VT)
'\f'	(0x0c)	feed (FF)
'\r'	(0x0d)	carriage return (CR)

[1] http://www.cplusplus.com/reference/cctype/isspace/

int iscntrl (int c);[2]

- to check whether c is a control character.
- a control character
  - does not occupy a printing position on a display
  - the opposite of a printable character (isprint)
- for the standard ASCII character set (used by the "C" locale)
  - between ASCII codes 0x00 (NUL) and 0x1f (US),
  - 0x7f (DEL)

[2] http://www.cplusplus.com/reference/cctype/iscntrl/

int isprint (int c);[3]

- to check whether c is a printable character.
- a printable character is a character
  - occupies a printing position on a display
  - the opposite of a control character (iscntrl).
- for the standard ASCII character set (used by the "C" locale),
  - greater than 0x1f (US),
  - except 0x7f (DEL).
- space (0x20)
  - isprint(' ') returns true
  - isgraph(' ') returns false

[3] http://www.cplusplus.com/reference/cctype/isprint/

int isgraph (int c);[4]

- to check whether c is a character with graphical representation
- the characters with graphical representation
  - all printable characters except the space character (' ').

[4] http://www.cplusplus.com/reference/cctype/isgraph/

int ispunct (int c);[5, 6]

- to check if the given character is a punctuation character as classified by the current C locale.
- the default C locale classifies the punctuation characters
  - `#$%&'()*+,-./:;<=>?@[\]^_‘{|}~`
- the standard "C" locale considers punctuation characters
  - all graphic characters (as in isgraph)
  - that are not alphanumeric (as in isalnum).

[5] http://www.cplusplus.com/reference/cctype/ispunct/
[6] http://en.cppreference.com/w/cpp/string/byte/ispunct

CR, NL, FF Examples [7]

- printf("aaabbb\rccc\n\n"); ..... \r : 0x0D
  - cccbbb
- printf("aaabbb\nccc\n\n"); ..... \n : 0x0A
  - aaabbb
  - ccc
- printf("aaabbb\fccc\n\n"); ..... \f : 0x0C
  - aaabbb
  - ccc

```
00000000: 6161 6162 6262 0d63 6363 0a0a 6161 6162  aaabbb.ccc..aaab
00000010: 6262 0a63 6363 0a0a 6161 6162 6262 0c63  bb.ccc..aaabbb.c
00000020: 6363 0a0a                                cc..
```

[7] https://stackoverflow.com/questions/3091524/what-are-carriage-return-linefeed-and-form-feed

CR, NL, FF Symbols [7]

		Original	Current
CR	0x0D	Carriage Return	Carriage Return
LF	0x0A	Line Feed	New Line
FF	0x0C	Form Feed	Section Separator

CR, NL, FF Old Definitions [7]

CR	0x0D	return to the beginning of the current line
LF	0x0A	advance downward to the next line
FF	0x0C	advance downward to the next page

- On old dot-matrix printers
  - advancing to the next line involved two operations:
  - moving the printer head back to the beginning of the horizontal scan range (CR)
  - advancing the paper by one line to print on (LF)

CR, NL, FF Current Usages [7]

CR	0x0D	Carriage Return return to the beginning of the current line
LF	0x0A	New Line terminates lines LF, NL, CRLF, (CRNL : x)
FF	0x0C	Section Separator advance downward to the next page

- Caret Notation
  - CR : `^M` : `/r`
  - LF : `^J` : `/n`
  - FF : `^L` : `/f`

CR, NL, FF on Windows and Unix-likes [7]

Windows	CRLF	0x0D0A	/r/n
Unix-likes	NL	0x0A	/n
New Macs	NL	0x0A	/n
Old Macs	CR	0x0D	/r