C++
Programming Bootcamp 2

COSC2802

TOPIC 3

C strings / Char arrays

Array of Char

The most basic form of a string is just an array of characters.

```
char string[LENGTH] = "Hello world!";
std::cout << string << std::endl;
```

What happens if LENGTH is longer than the string?

We end up with junk data at the end, so when we print it out, all kinds of stuff could come out.

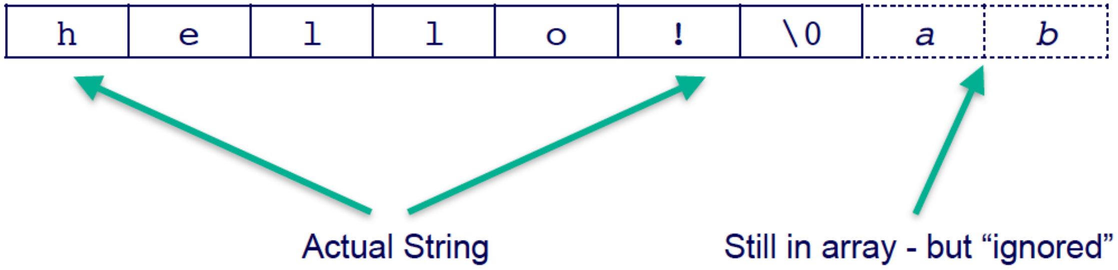
How do we know where the end of our string is then?

String termination character

- The null ‘\0’ termination character denotes the “end” of a string

h	e	l	l	o	!	\0

- If the ‘\0’ appears in the middle of a “string” it actually terminates earlier
  - The rest of the array is ignored



Constant strings

- Anything that is enclosed in double-quote “” are a ***constant string***
  - The compiler generates:
    - A character array
    - Of the exact length required
    - Guaranteed to end in a ‘\0’
  - Constant strings cannot be modified
  - Can be assigned or copied to ***mutable*** strings

Note the difference:

- IMMUTABLE: constant string either with “xyz” or const std::string
- MUTABLE: std::string objects

Lexical Comparison of Strings

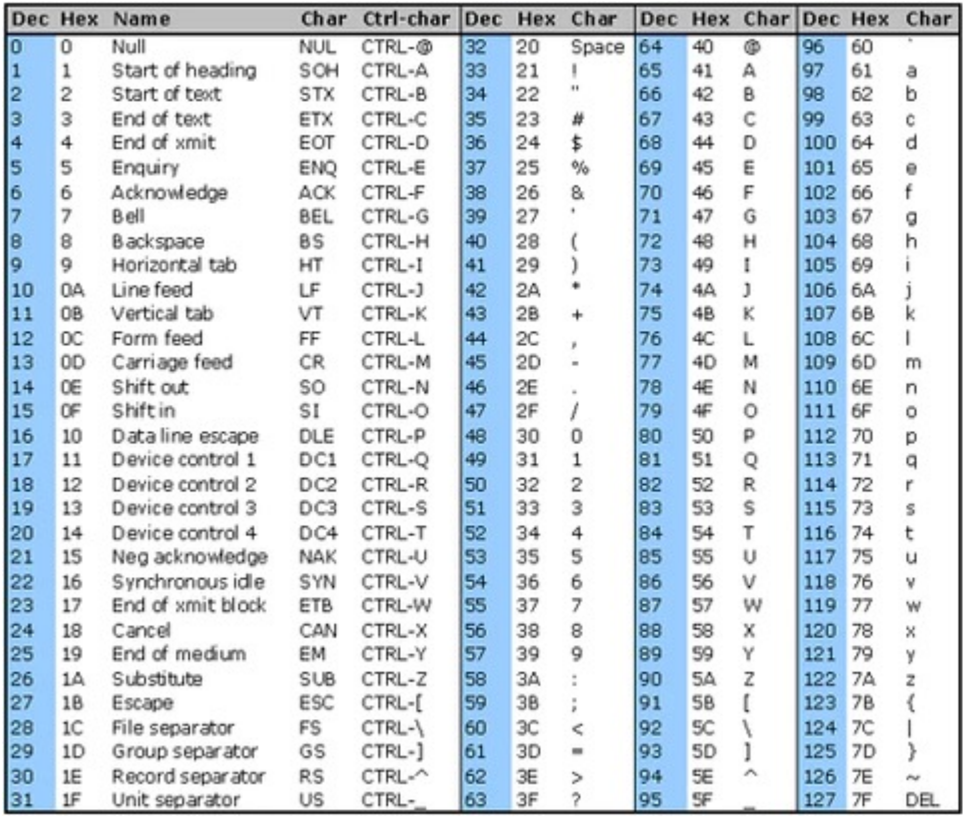
- Two string can be compared for ***lexical ordering***.
  - That is not necessarily the order which the two strings appear in a dictionary
  - The comparison uses ASCII (or unicode) values
- Each character of the string is compared one-at-a-time, in order until two characters differ
  - The character with the small ASCII value is ***lexicographically first***
  - Otherwise, the smaller string comes first

string 1:	h	e	l	l	o	!
compare (<)	<	<	<	<		
string 2:	h	e	l	p		
string 3:	h	e	l	l		

Compare using strcmp (see zybooks 7.14.2):

int strcmp(str1, str2); // Returns 0 if str1 and str2 are equal, non-zero if differ.

ASCII Chart



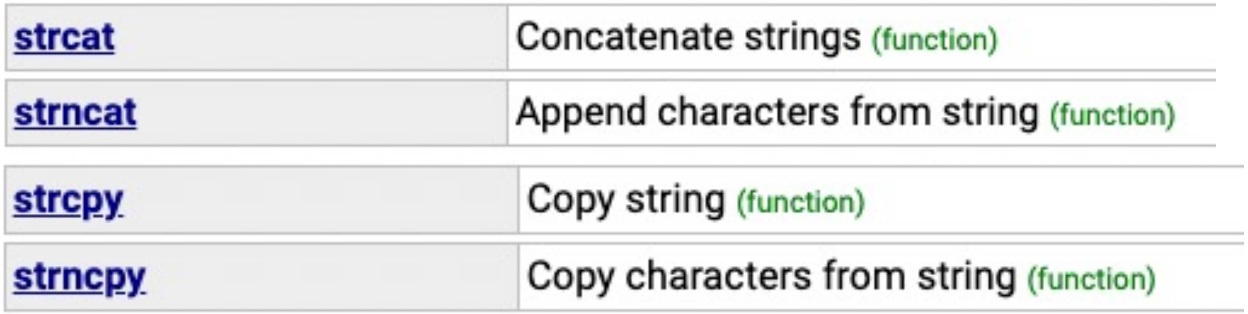
Dec	Hex	Name	Char	Ctrl-char	Dec	Hex	Char	Dec	Hex	Char	Dec	Hex	Char
0	0	Null	NUL	CTRL-@	32	20	Space	64	40	@	96	60	`
1	1	Start of heading	SOH	CTRL-A	33	21	!	65	41	A	97	61	a
2	2	Start of text	STX	CTRL-B	34	22	"	66	42	B	98	62	b
3	3	End of text	ETX	CTRL-C	35	23	#	67	43	C	99	63	c
4	4	End of xmit	EOT	CTRL-D	36	24	$	68	44	D	100	64	d
5	5	Enquiry	ENQ	CTRL-E	37	25	%	69	45	E	101	65	e
6	6	Acknowledge	ACK	CTRL-F	38	26	&	70	46	F	102	66	f
7	7	Bell	BEL	CTRL-G	39	27	'	71	47	G	103	67	g
8	8	Backspace	BS	CTRL-H	40	28	(	72	48	H	104	68	h
9	9	Horizontal tab	HT	CTRL-I	41	29	)	73	49	I	105	69	i
10	0A	Line feed	LF	CTRL-J	42	2A	*	74	4A	J	106	6A	j
11	0B	Vertical tab	VT	CTRL-K	43	2B	+	75	4B	K	107	6B	k
12	0C	Form feed	FF	CTRL-L	44	2C	,	76	4C	L	108	6C	l
13	0D	Carriage feed	CR	CTRL-M	45	2D	-	77	4D	M	109	6D	m
14	0E	Shift out	SO	CTRL-N	46	2E	.	78	4E	N	110	6E	n
15	0F	Shift in	SI	CTRL-O	47	2F	/	79	4F	O	111	6F	o
16	10	Data line escape	DLE	CTRL-P	48	30	0	80	50	P	112	70	p
17	11	Device control 1	DC1	CTRL-Q	49	31	1	81	51	Q	113	71	q
18	12	Device control 2	DC2	CTRL-R	50	32	2	82	52	R	114	72	r
19	13	Device control 3	DC3	CTRL-S	51	33	3	83	53	S	115	73	s
20	14	Device control 4	DC4	CTRL-T	52	34	4	84	54	T	116	74	t
21	15	Neg acknowledge	NAK	CTRL-U	53	35	5	85	55	U	117	75	u
22	16	Synchronous idle	SYN	CTRL-V	54	36	6	86	56	V	118	76	v
23	17	End of xmit block	ETB	CTRL-W	55	37	7	87	57	W	119	77	w
24	18	Cancel	CAN	CTRL-X	56	38	8	88	58	X	120	78	x
25	19	End of medium	EM	CTRL-Y	57	39	9	89	59	Y	121	79	y
26	1A	Substitute	SUB	CTRL-Z	58	3A	:	90	5A	Z	122	7A	z
27	1B	Escape	ESC	CTRL-[	59	3B	;	91	5B	[	123	7B	{
28	1C	File separator	FS	CTRL-\	60	3C	<	92	5C	\	124	7C	|
29	1D	Group separator	GS	CTRL-]	61	3D	=	93	5D	]	125	7D	}
30	1E	Record separator	RS	CTRL-^	62	3E	>	94	5E	^	126	7E	~
31	1F	Unit separator	US	CTRL-_	63	3F	?	95	5F	_	127	7F	DEL

C-string library functions

cstring library:

- #include<cstring>

Functions include concatenation, copying, ….



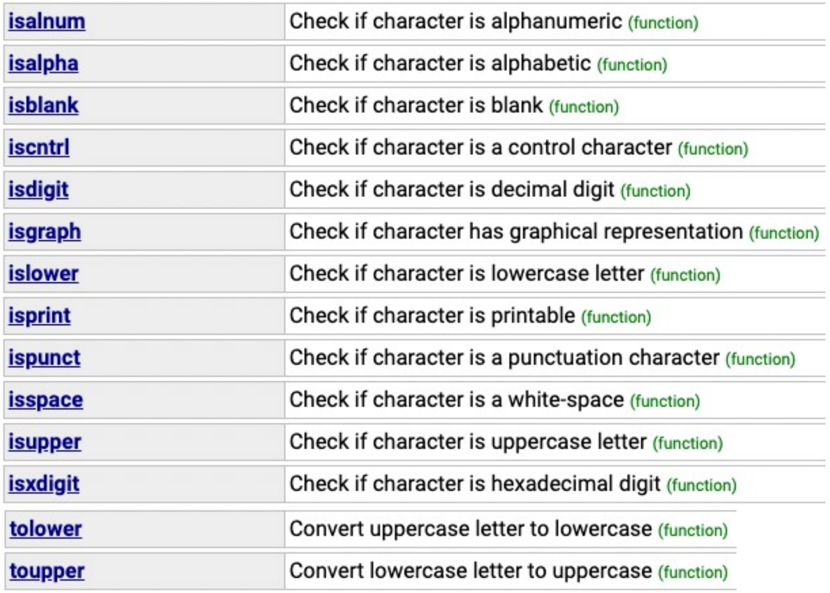
strcat	Concatenate strings (function)
strncat	Append characters from string (function)
strcpy	Copy string (function)
strncpy	Copy characters from string (function)

See: https://cplusplus.com/reference/cstring/

char library functions

cctype library: #include<cctype>

Functions include **char** checking and conversion ….



isalnum	Check if character is alphanumeric (function)
isalpha	Check if character is alphabetic (function)
isblank	Check if character is blank (function)
iscntrl	Check if character is a control character (function)
isdigit	Check if character is decimal digit (function)
isgraph	Check if character has graphical representation (function)
islower	Check if character is lowercase letter (function)
isprint	Check if character is printable (function)
ispunct	Check if character is a punctuation character (function)
isspace	Check if character is a white-space (function)
isupper	Check if character is uppercase letter (function)
isxdigit	Check if character is hexadecimal digit (function)
tolower	Convert uppercase letter to lowercase (function)
toupper	Convert lowercase letter to uppercase (function)

See: https://cplusplus.com/reference/cctype/

NOTE:

C++ **string class** preferred because it eliminates many security problems and bugs that can be caused by manipulating C strings

However, understanding of C strings, pointers and built-in arrays is important:

- C-string processing required for command-line arguments
- Likely encounter C-strings in legacy C and C++ programs

We’ll revisit this later as pointer-based strings

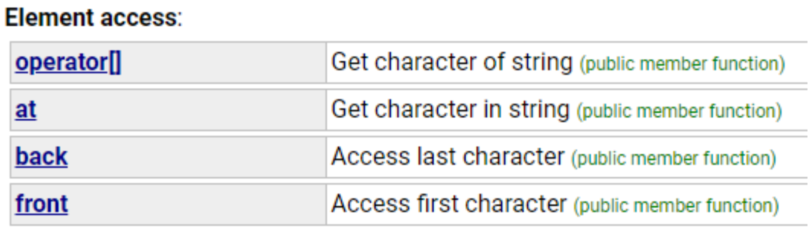
std::string class | Indexing

A part of the <string> STL package.

```
std::string str = "Hello world";

std::cout << str << std::endl;
str.push_back('!');
std::cout << str << std::endl;
```

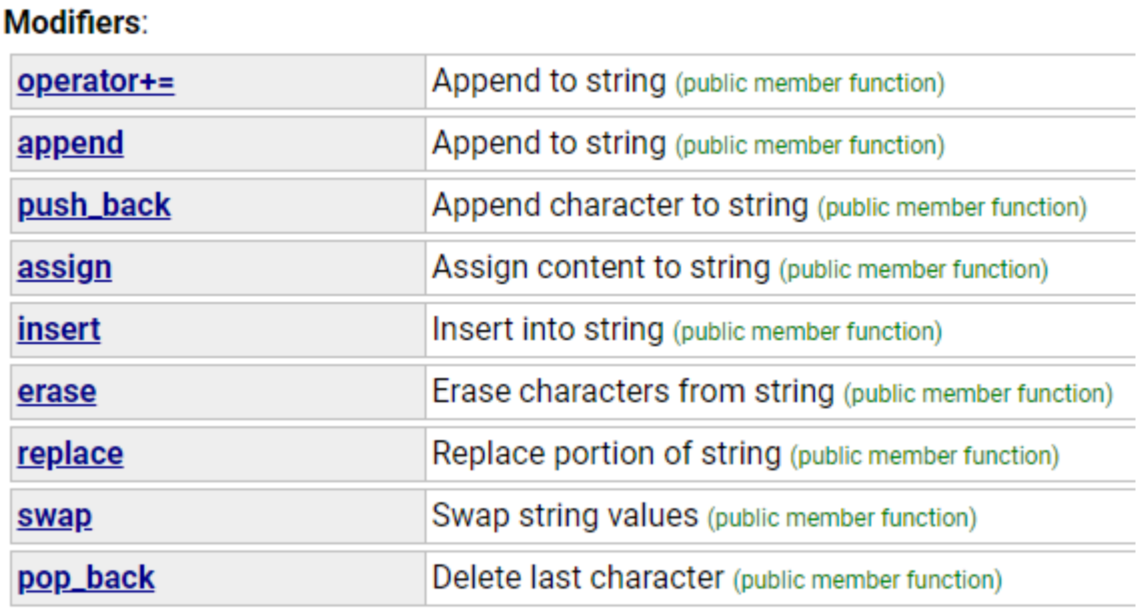
Provides a wrapper for C-Style strings. Kind of like std::array: it provides a lot of helper functions to let you manage the object.



Element access:

operator[]	Get character of string (public member function)
at	Get character in string (public member function)
back	Access last character (public member function)
front	Access first character (public member function)

std::string class | Methods



Modifiers:

operator+=	Append to string (public member function)
append	Append to string (public member function)
push_back	Append character to string (public member function)
assign	Assign content to string (public member function)
insert	Insert into string (public member function)
erase	Erase characters from string (public member function)
replace	Replace portion of string (public member function)
swap	Swap string values (public member function)
pop_back	Delete last character (public member function)