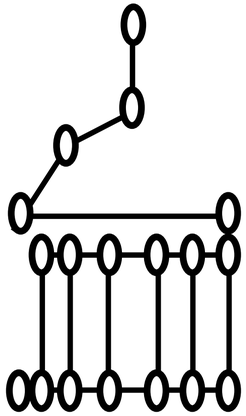
Lecture: Strings

ENGR 2730:Computers in Engineering



Five-Step Problem-Solving Methodology

1. State the problem clearly.
2. Describe the input and output.
3. Work hand examples.
4. Develop a solution/algorithm.
5. Test your solution.

Characters

- char
  - A variable of char type, as in

    char myChar;

    can store a single character like the letter m.
- character literal
  - A character literal is surrounded with single quotes, as in

    myChar = 'm';

ASCII TABLE

Decimal	Hex	Char	Decimal	Hex	Char	Decimal	Hex	Char	Decimal	Hex	Char
0	0	[NULL]	32	20	[SPACE]	64	40	@	96	60	`
1	1	[START OF HEADING]	33	21	!	65	41	A	97	61	a
2	2	[START OF TEXT]	34	22	"	66	42	B	98	62	b
3	3	[END OF TEXT]	35	23	#	67	43	C	99	63	c
4	4	[END OF TRANSMISSION]	36	24	$	68	44	D	100	64	d
5	5	[ENQUIRY]	37	25	%	69	45	E	101	65	e
6	6	[ACKNOWLEDGE]	38	26	&	70	46	F	102	66	f
7	7	[BELL]	39	27	'	71	47	G	103	67	g
8	8	[BACKSPACE]	40	28	(	72	48	H	104	68	h
9	9	[HORIZONTAL TAB]	41	29	)	73	49	I	105	69	i
10	A	[LINE FEED]	42	2A	*	74	4A	J	106	6A	j
11	B	[VERTICAL TAB]	43	2B	+	75	4B	K	107	6B	k
12	C	[FORM FEED]	44	2C	,	76	4C	L	108	6C	l
13	D	[CARRIAGE RETURN]	45	2D	-	77	4D	M	109	6D	m
14	E	[SHIFT OUT]	46	2E	.	78	4E	N	110	6E	n
15	F	[SHIFT IN]	47	2F	/	79	4F	O	111	6F	o
16	10	[DATA LINK ESCAPE]	48	30	0	80	50	P	112	70	p
17	11	[DEVICE CONTROL 1]	49	31	1	81	51	Q	113	71	q
18	12	[DEVICE CONTROL 2]	50	32	2	82	52	R	114	72	r
19	13	[DEVICE CONTROL 3]	51	33	3	83	53	S	115	73	s
20	14	[DEVICE CONTROL 4]	52	34	4	84	54	T	116	74	t
21	15	[NEGATIVE ACKNOWLEDGE]	53	35	5	85	55	U	117	75	u
22	16	[SYNCHRONOUS IDLE]	54	36	6	86	56	V	118	76	v
23	17	[ENG OF TRANS. BLOCK]	55	37	7	87	57	W	119	77	w
24	18	[CANCEL]	56	38	8	88	58	X	120	78	x
25	19	[END OF MEDIUM]	57	39	9	89	59	Y	121	79	y
26	1A	[SUBSTITUTE]	58	3A	:	90	5A	Z	122	7A	z
27	1B	[ESCAPE]	59	3B	;	91	5B	[	123	7B	{
28	1C	[FILE SEPARATOR]	60	3C	<	92	5C	\	124	7C	|
29	1D	[GROUP SEPARATOR]	61	3D	=	93	5D	]	125	7D	}
30	1E	[RECORD SEPARATOR]	62	3E	>	94	5E	^	126	7E	~
31	1F	[UNIT SEPARATOR]	63	3F	?	95	5F	_	127	7F	[DEL]

Characters

- whitespace character
  - A whitespace character is a character used to represent horizontal and vertical spaces in text, and includes spaces, tabs, and newline characters.
- escape sequence
  - Escape sequence: A two-character sequence starting with \ that represents a special character.

Escape sequence	Char
\n	newline
\t	tab
\'	single quote
\"	double quote
\\	backslash

Question

- What is the ASCII code for the letter A?
  - Answer: 65
- What is the ASCII code for the letter A without looking at the ASCII Table?
  - Answer: 'A'
- **In your code, I want you to use 'A' instead of 65.**

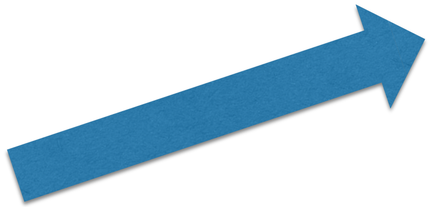
Strings and String Literals

- There are two types of strings
  - A C-string is implemented as a null-terminated array of type char.
    Example:

    char buffer[] = "blue";

    Same As

    char buffer[5] = {'b', 'l', 'u', 'e', '\0'};



Address	Value
1000	'b'
1001	'l'
1002	'u'
1003	'e'
1004	'\0'

  - A C++ string object.
- string literal
  - A string literal surrounds a character sequence with double quotes, as in "Hello World".

C++ string object

- C++ string

```
string name = "University of Iowa";
```

- The default value for a string is the empty string (i.e., "").
- Operators (e.g., +, =, ==) have been *overloaded* to make working with strings easier
- Numerous member functions on the string object:
  - name.length() or name.size() return the length of string name
  - name[i], name.at(i), or an iterator can be used for accessing an element of string name
  - many other functions exist

Input string from the user

- Can use cin directly for strings without spaces

```
string name;
cout << "Please enter your first name: ";
cin >> name;
cout << "Hello " << name << "!" << endl;
```

- Can use getline function to read from the standard input stream object cin the characters the user enters, up to, but not including, the newline

```
string completeName;
cout << "Please enter your first and last name: ";
cin.ignore(); // throw away remaining '\n'
getline(cin, completeName, '\n'); //read until newline is encountered
cout << "Hello " << completeName << "!" << endl;
```

C++ string object

```cpp
using namespace std;

int main() {
    string s1; // empty string
    string s2 = "Hello";
    string s3("world!");

    // assignment and concatenation example
    s1 = s2;    // assign one string to another
    s1 +=  " "; // concatenation with character string
    s1 += s3;   // concatenation with another string
    cout << "s1 = " << s1 << endl;
    cout << "s2 = " << s2 << endl;
    cout << "s3 = " << s3 << endl;

    // another option for concatenation
    string s4; // empty string
    s4 = s2 + " " + s3;
    cout << "s4 = " << s4 << endl;
```

```
s1 = Hello world!
s2 = Hello
s3 = world!
```

```
s4 = Hello world!
```

```
s1 is equal to s4
s2 equals "Hello"
```

```cpp
    // comparing strings
    if (s1 == s4) {
        cout << "s1 is equal to s4" << endl;
    }

    if (s2 == "Hello") {
        cout << "s2 equals \"Hello\"" << endl;
    }

    // reading a string (one word)
    string name;
    cout << "Please enter your first name: ";
    cin >> name;
    cout << "Hello " << name << "!" << endl;

    // reading a line
    string completeName;
    cout << "Please enter your first and last name: ";
    cin.ignore(); // throw away remaining '\n'
    getline(cin, completeName, '\n'); //read until newline
    cout << "Hello " << completeName << "!" << endl;

    return 0;
}
```

```
Please enter your first name: Mo
Hello Mo!
```

```
Please enter your first and last name: Mo Ro
Hello Mo Ro!
```

C++ string object

```
using namespace std;

int main() {
    string myString = "CIE classroom";
    cout << "myString = " << myString << endl;

    // iterate through all elements of the string: option 1
    for (size_t i = 0; i < myString.length(); i++)
    {
        cout << myString[i] << " ";
    }
    cout << endl;

    // iterate through all elements of the string: option 2
    for (size_t i = 0; i < myString.length(); i++)
    {
        cout << myString.at(i) << " ";
    }
    cout << endl;
```

```
myString = CIE classroom
C I E   c l a s s r o o m
C I E   c l a s s r o o m
class found at 4
After replacement, myString = CIE funroom
```

```
    // search/replace example
    string searchString = "class";
    size_t pos = myString.find(searchString);
    if (pos != string::npos)
    {
        cout << searchString << " found at "
             << pos << endl;
    }

    // replace class with fun
    size_t startPos = pos;
    size_t numberCharsToReplace = searchString.size();
    string replacementStr = "fun";
    myString.replace(startPos, numberCharsToReplace,
                     replacementStr);

    cout << "After replacement, myString = "
         << myString << endl;

    return 0;
}
```

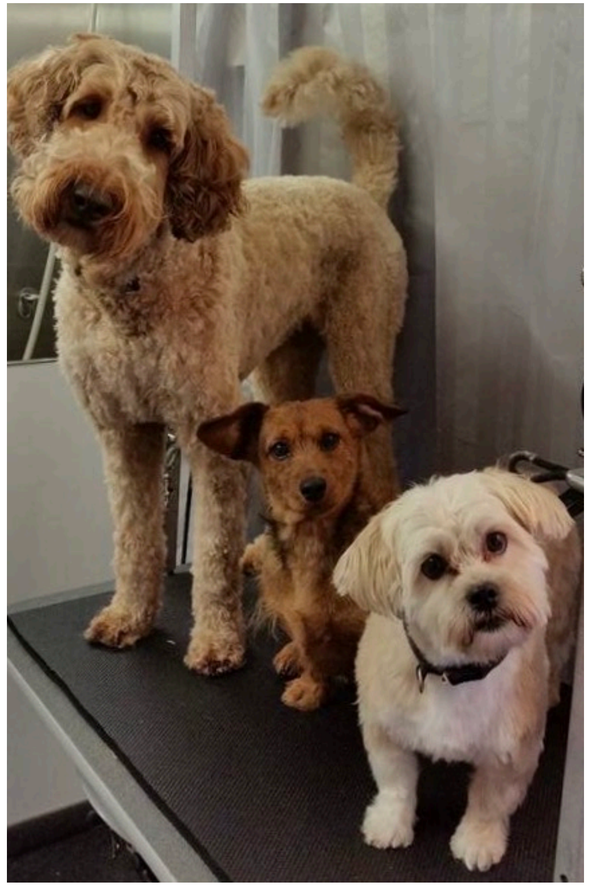
CQ: What is the output of the function below?

```
int main()
{
    string myDog1 = "Truman";
    string myDog2 = "Bess";

    string generalSentence = "dog and dog are very cute.";

    generalSentence.replace(generalSentence.find("dog"), 3, myDog1);
    generalSentence.replace(generalSentence.find("dog"), 3, myDog2);

    cout << generalSentence << endl;
}
```



A. Tru and Bes are very cute.

B. Bes and Tru are very cute.

C. Truman and Bess are very cute.

D. Bess and Truman are very cute.

E. Bess and Truman are very ugly.

In class assignment

1. Copy the lec07Strings directory from the Public directory to your Practice directory. This is a copy of the program in the previous notes
2. Load project CMakeLists.txt file
3. Run program.
4. Add code to the beginning of the program to read in a string.
5. Reverse the order of the characters in the string.
6. Print the reversed string.