CSCI 2170
C type String

C type string: a sequence of characters (in an array) terminated by a null character '\0'

```
const int SIZE=20;
char   lastname[SIZE], firstname[SIZE];

lastname = "Brown";
firstname= "Amy";
```

What will be the output of the following?

1. cout << "The person's name is :" << firstname << " " << lastname << endl;
2. cout << "The person's last and first initial is: " << lastname[0] << " " << firstname[0] << endl;

(1) Enter string interactively
(1a) read from keyboard character by character

```
int  index;
char   ch;

cout << "Please enter the last name"
<< endl;
cin.get(ch);
index=0;
```

```
while (ch != '\n')
{
        lastname[index]=ch;
        cin.get(ch);
        index++;
}
lastname[index] = '\0';
```

(1b) read characters from the keyboard up to the first white space

```
cout << "Please enter the lastname" << endl;
cin >> lastname;
```

(1c) read from keyboard the entire line of characters

```
cout << "please enter the last name" << endl;
gets(lastname);
```

(2) more examples
(2a) count the number of blank spaces entered on one line of text
(2b) count the number of lines of text entered through keyboard

(3) Output the string

```
cout << text << endl;     or      puts(text);     // need to #include  <cstdio>
```

- **Library functions associated with C type string**

(1) **strlen()** -- returns the number characters in a string before the first NULL character
usage: lengh = strlen(string1);

```
cout << strlen("How now brown cow");
```

(2) **strcpy()** – creates a copy of a string
usage: strcpy(string1, string2);
copy-to copy-from

```
const int SIZE=20;
char lastname[SIZE], newlastname[SIZE];

lastname = "Johnson";
strcpy(newlastname, lastname);  or strcpy(newlastname, "Johnson");
```

(3) **strcat()** – string concatenation, appending one string at the end of another string
usage strcat(string1, string2);

```
char part1[size] = "one two ";
char part2[size] = "buckle my shoe";
strcat(part1, part2);
```

(4) **strcmp()** – compares two strings by comparing the position of their characters in the underlying character set
usage: result = strcmp(string1, string2);
if string 1 and string 2 are identical, strcmp returns 0
if string1 lexicographically less than string 2, strcmp returns a negative value
if string 1 lexicographically greater than string 2, strcmp returns a positive value

```
strcmp("Tomato", "Apple")     → returns positive value
strcmp("one", "one");         → returns 0
strcmp("now", "She");         → returns positive value
strcmp("one", "two");         → returns negative value
```

(5) What is the output of the following program?

```
const int SIZE = 512;
int main()
{
    char workingStr[size];
    char str1[size]="I know an old lady";
    char str2[size]="who swallowed a fly";

    strcpy(workingStr, str1);
    if (!strcmp(workingStr, str1))
            cout << "Copy successful" << endl;
    else
            cout << "Error in copy" << endl;

    strcat(workingStr, " ");
    strcat(workingStr, str2);

    cout << "The current string is " << workingStr << endl;
    cout << "It has " <<  strlen(workingStr) << "characters" << endl;

    return 0;
}
```

(6) how to write your own versions of the string functions which can have the same effects as the ones provided by the C++ library? For example: Mystrlen()