Tutorial 5B: C-Strings

Objectives: To practice with `C - Strings`

1. Given the declaration

```
char partOne[40]="BT137";
char partTwo[40]="WT105";
```

a. **Write an `if` statement that compares two strings and outputs content of the string with the greatest value.**

b. **Copies content of `partOne` into `partTwo`.**

c. **Checks that the `partTwo` array size is sufficient to store the content of `partOne` String.**

2. Write a function that pads a variable-length string with blanks to its maximum size. For example, if s10 is a ten-character array currently holding the string "screen", `blank_pad` would add three blanks (one of which would overwrite the null character) and finish the string with the null character. Be sure your function would work if no blank padding were necessary.

3. Write a program that takes a word less than 25 characters long and prints a statement like this:

```
fractal starts with the letter f
```

Have the program process words continually until it encounters a "word" beginning with the character `'9'`.

4. Given these declarations,

```
char socsec[12] = "123-45-6789";
char ssnshort[7], ssn1[4], ssn2[3], ssn3[5];
```

write statements to accomplish the following:
a. Store in `ssnshort` as much of `socsec` as will fit.
b. Store in `ssn1` the first three characters of `socsec`.
c. Store in `ssn2` the middle two-digit portion of `socsec`.
d. Store in `ssn3` the final four digits of `socsec`.

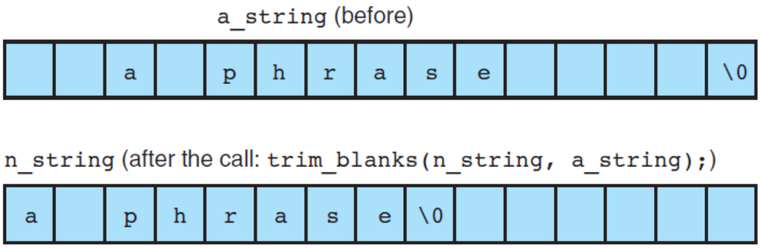
Be sure your statements store valid strings in each variable.

5. Given the string `pres` (value is "Adams, John Quincy"). There is an error in the last line of the following code fragment. What is the error? Why is it wrong? How would you correctly achieve the intent of this call?

```
    strcpy(tmp1, &pres[12]);
    strcat(tmp1, " ");
    strcat(tmp1, pres[7]);
```

6. **Write a function `bracket_by_len` that takes a word as an input argument and returns the word bracketed to indicate implicitly its length. Words less than five characters long are bracketed with << >> , words five to ten letters long are bracketed with (* *) , and words over ten characters long are bracketed with /+ +/ . Your function should require the calling function to provide as the first argument, space for the result, and as the third argument, the amount of space available.**

7. **Complete function `trim_blanks` whose purpose is to take a single string input parameter (`to_trim`) and return a copy of the string with leading and trailing blanks removed. Use `strncpy` in `trim_blanks` .**



`a_string` (before)

		a		p	h	r	a	s	e					\0

`n_string` (after the call: `trim_blanks(n_string, a_string);`)

a		p	h	r	a	s	e	\0						