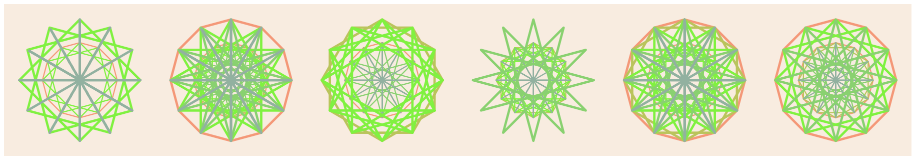
# Exercise

Professor Leon Tabak
Cornell College

22 May 2022



Using the C programming language, write a program that...

- Creates two strings.
- Prints the lengths of the two strings.
- Determines the correct alphabetical ordering of the two strings.
- Uses the two strings to build a larger string.

Begin by studying this program. Build your own program by imitating this example.

```
#include <stdio.h>
#include <stdlib.h>
#include <string.h>

int main( int argc, char** argv ) {
    // Create 2 strings
    //
    // These are the names of the 2 longest
    // rivers in the United States.
    //
    // They define the eastern and western borders
    // of Iowa.
    char mississippi[] = "Mississippi";
    char missouri[] = "Missouri";

    printf( "\n" );

    printf( "River on the eastern border = %s\n", mississippi );
    printf( "River on the western border = %s\n", missouri );

    printf( "\n"  );

    // Which river's name is longer?
    // Count the number of letters in the 2 strings.
    // 'strlen' means 'string length'
    int mississippiLength = strlen( mississippi );
    int missouriLength = strlen( missouri );

    printf( "number of letters in 'Mississippi' = %2d\n",
        mississippiLength );

    printf( "number of letters in 'Missouri' = %2d\n",
        missouriLength );

    printf( "\n\n" );

    // Count letters in 'Mississippi' in a different way.
    int count = 0;
    while( mississippi[count] != '\0' ) {
        count++;
    } // while

    printf( "number of letters in 'Mississippi' = %2d\n",
        count );

    // Count letters in 'Missouri' in a different way.
    count = 0;
    while( missouri[count] != '\0' ) {
        count++;
    } // while

    printf( "number of letters in 'Missouri' = %2d\n",
        count );

    // Here is one way to create copies
    // of the 2 strings.
    // 'strcpy' means 'string copy'
    char a[strlen(mississippi)];
    strcpy( a, mississippi );

    char b[strlen(missouri)];
    strcpy( b, missouri );


    // Here is another way to create copies
    // of the 2 strings.
    /*
    char* a = (char*) calloc( 80, sizeof(char) );
    strcpy( a, mississippi );

    char* b = (char*) calloc( 80, sizeof(char) );
    strcpy( b, missouri );
    */

    // Compare the 2 strings.
    // Which word will be listed first
    // in a dictionary?
    //
    // 'strcmp' means 'string compare'
    // The strcmp() function returns an integer
    // that is negative, zero, or positive.
    //   * negative means correct order is first string, then second
    //   * positive means correct order is second string, then first
    //   * zero means two strings are equal (order does not matter)
    int code = strcmp( a, b );
    if( code < 0 ) {
        printf( "\n%s < %s\n", a, b );
    } // if
    else if( code > 0 ) {
        printf( "\n%s > %s\n", a, b );
    } // else if
    else {
        printf( "\n%s == %s\n", a, b );
    } // else

    // If we create new variables with the
    // calloc() function, then we should free
    // the memory that we allocated.
    /*
    free( a );
    free( b );
    */

    // Join strings.
    // 'strcat' means 'string concatenation'
    // 'concatenation' means to join one after the other
    char sentence[80];
    strcat( sentence, "The " );
    strcat( sentence, mississippi );
    strcat( sentence, " and " );
    strcat( sentence, missouri );
    strcat( sentence, " rivers define Iowa's borders." );

    printf( "\n%s\n", sentence );

    exit(0);
} // main( int, char** )
```