Course: Programming Fundamental - ENSF 337

Lab #: 4

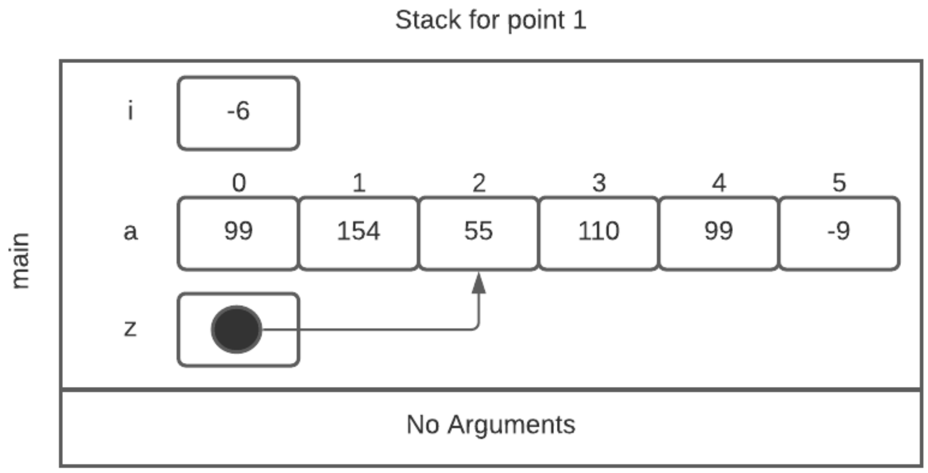
Instructor: Khedr

Student Name: Aleksander Berezowski

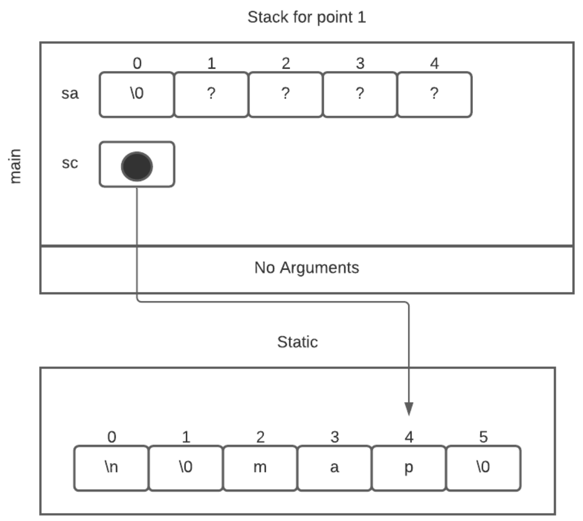
Lab Section: B04

Date submitted: October 20th

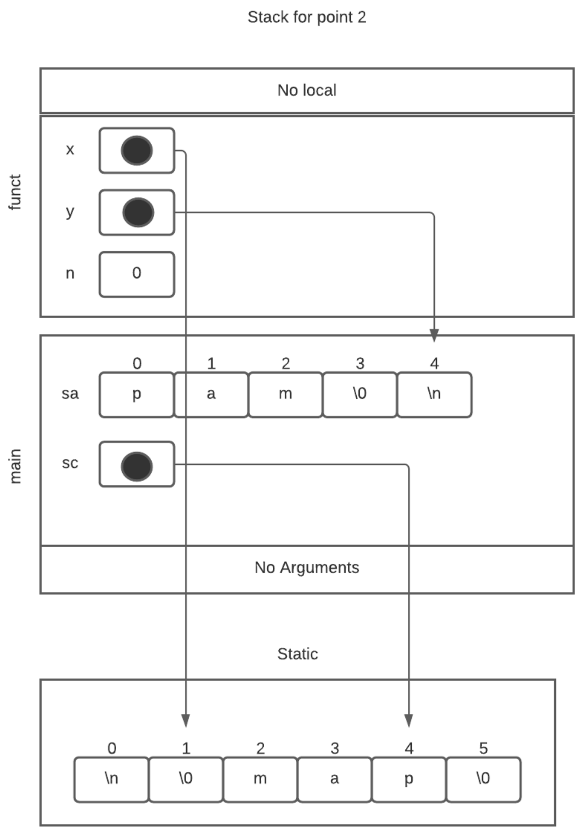
Exercise A


Stack for point 1
main
i
-6
a
0
1
2
3
4
5
99
154
55
110
99
-9
z
No Arguments

Exercise B

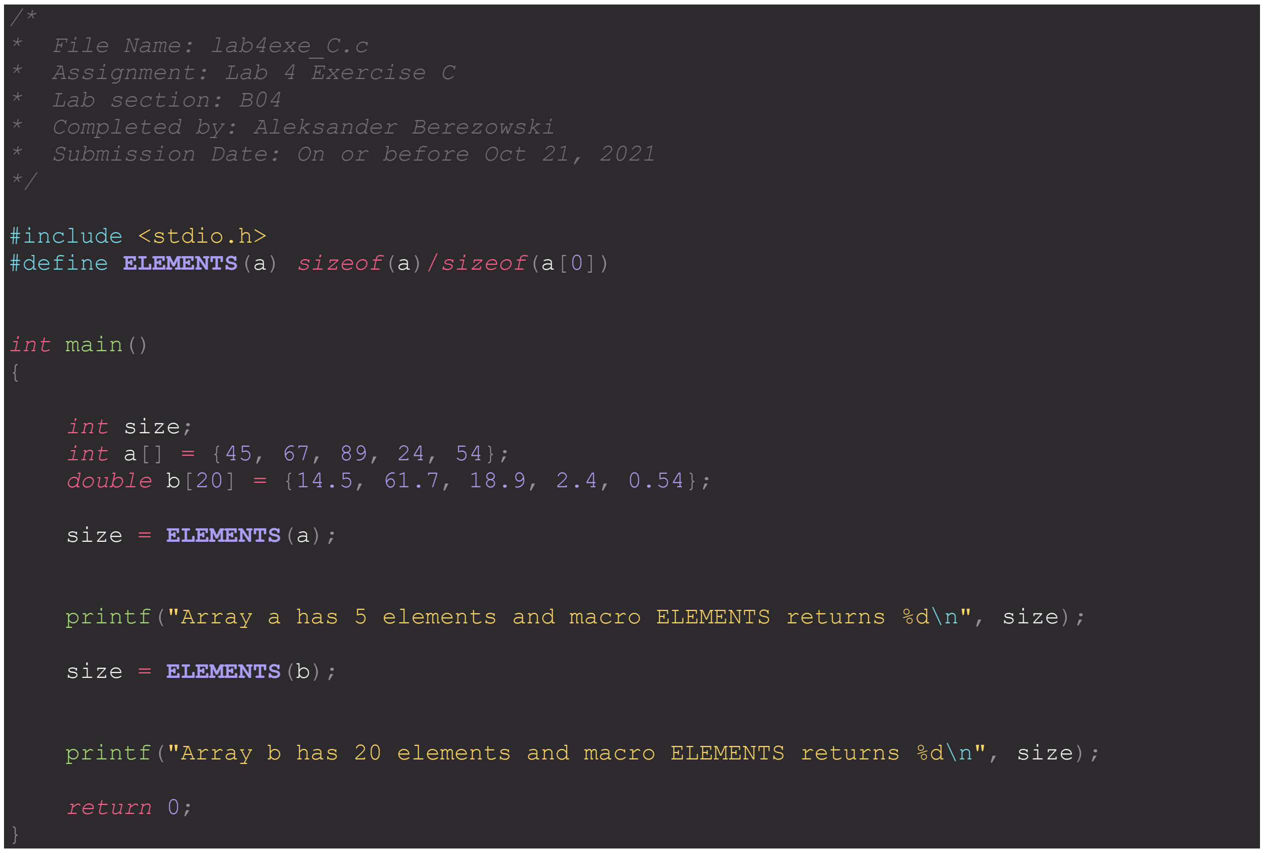


Stack for point 1
0
1
2
3
4
sa
\0
?
?
?
?
sc
main
No Arguments
Static
0
1
2
3
4
5
\n
\0
m
a
p
\0

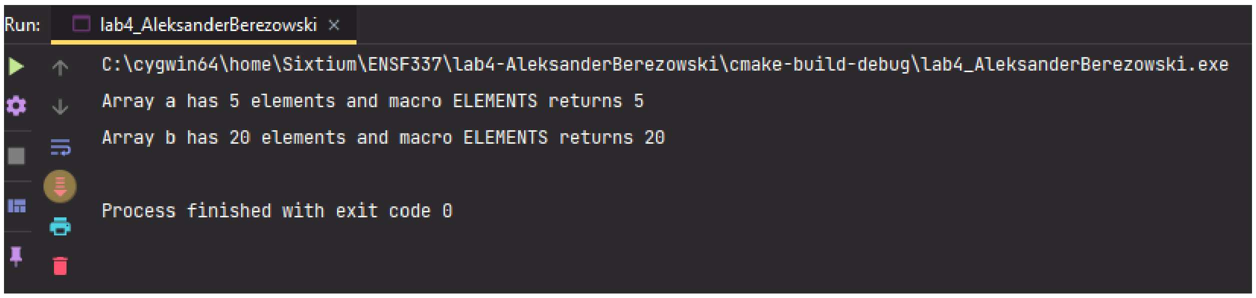


Stack for point 2
No local
x
y
n
0
funct
0
1
2
3
4
sa
p
a
m
\0
\n
sc
main
No Arguments
Static
0
1
2
3
4
5
\n
\0
m
a
p
\0

Exercise C



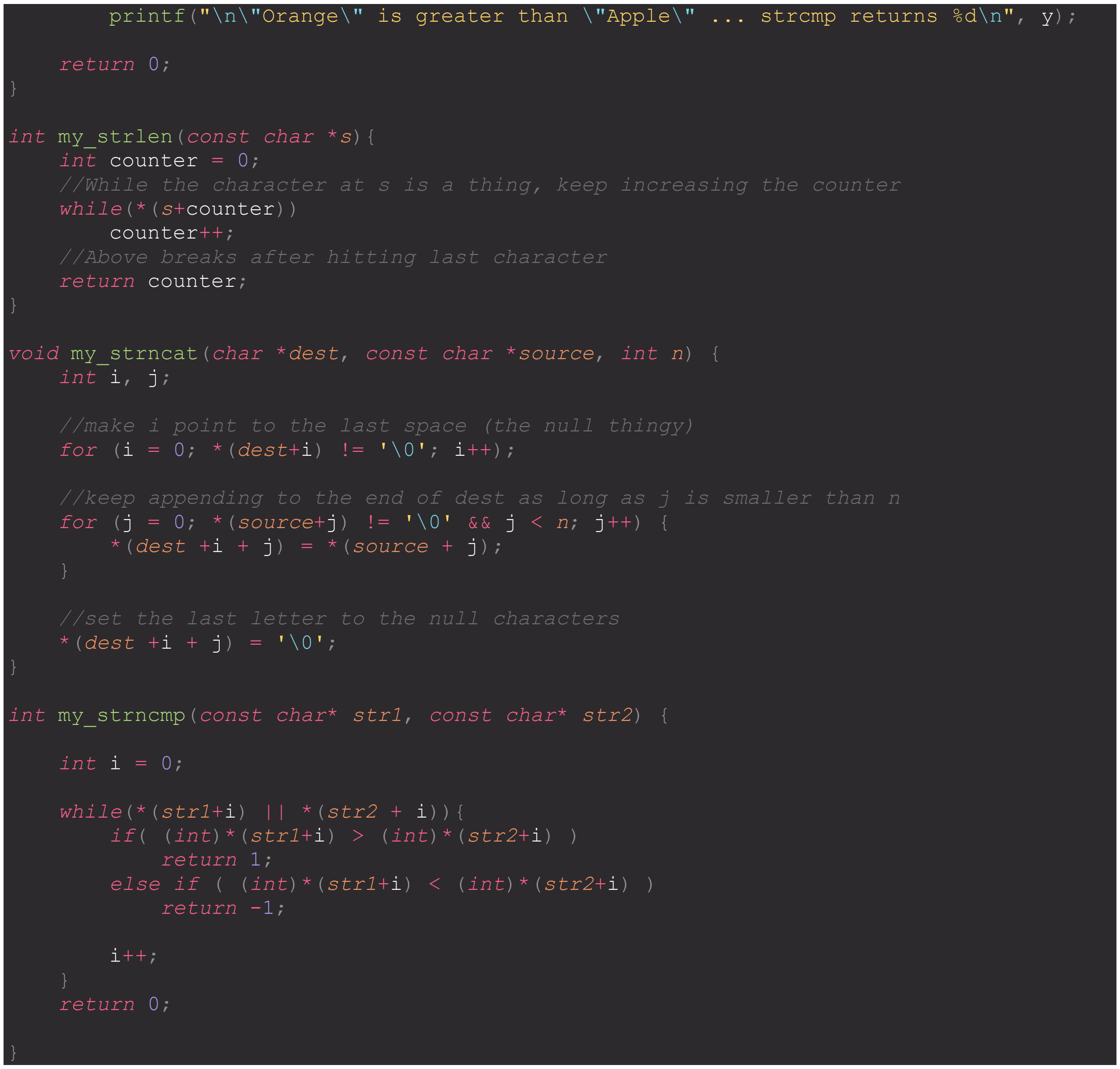
```c
/*
*  File Name: lab4exe_C.c
*  Assignment: Lab 4 Exercise C
*  Lab section: B04
*  Completed by: Aleksander Berezowski
*  Submission Date: On or before Oct 21, 2021
*/

#include <stdio.h>
#define ELEMENTS(a)  sizeof(a)/sizeof(a[0])


int main()
{

    int size;
    int a[] = {45, 67, 89, 24, 54};
    double b[20] = {14.5, 61.7, 18.9, 2.4, 0.54};

    size = ELEMENTS(a);


    printf("Array a has 5 elements and macro ELEMENTS returns %d\n", size);

    size = ELEMENTS(b);


    printf("Array b has 20 elements and macro ELEMENTS returns %d\n", size);

    return 0;
}
```



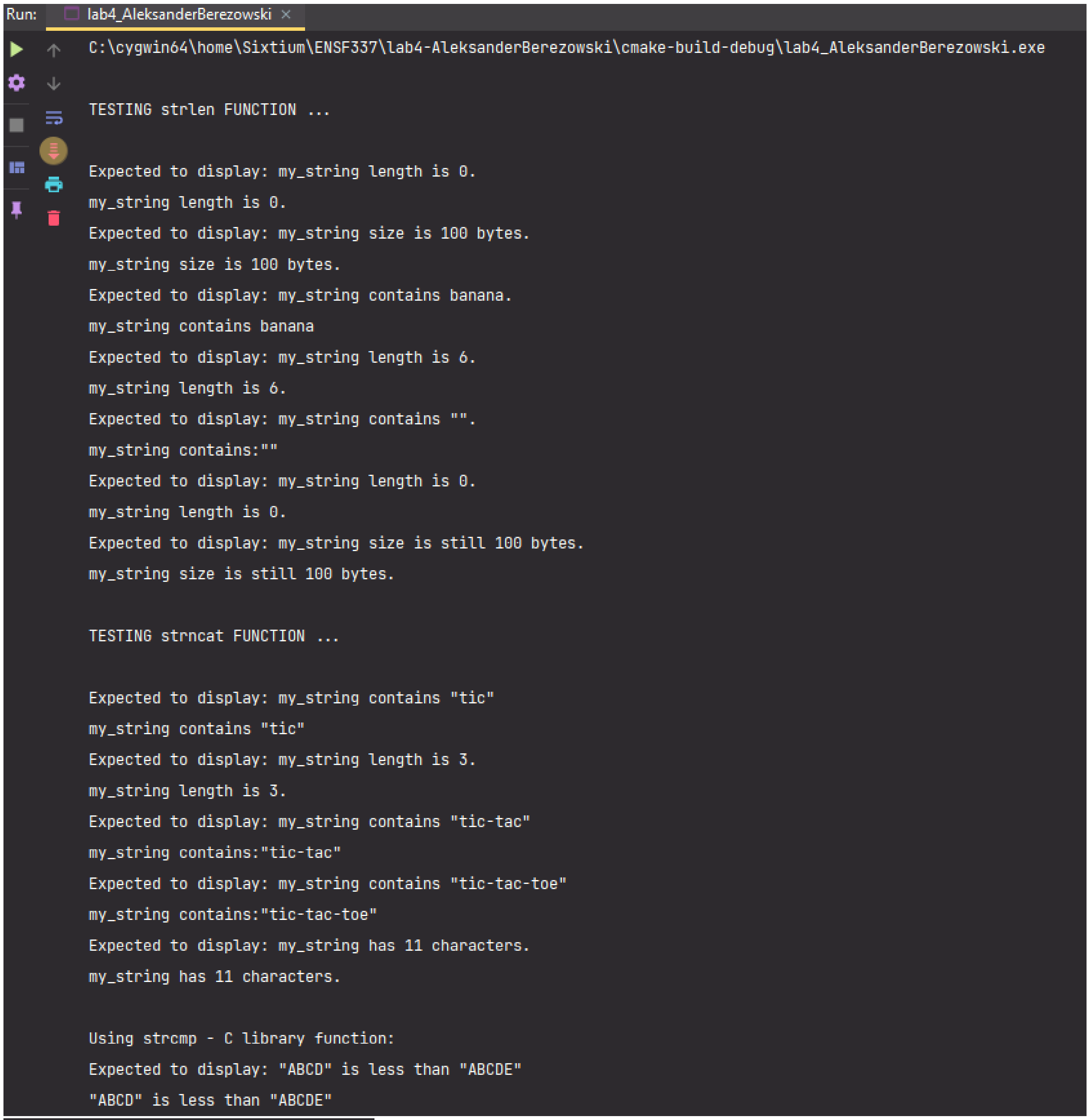
```
C:\cygwin64\home\Sixtium\ENSF337\lab4-AleksanderBerezowski\cmake-build-debug\lab4_AleksanderBerezowski.exe
Array a has 5 elements and macro ELEMENTS returns 5
Array b has 20 elements and macro ELEMENTS returns 20

Process finished with exit code 0
```

Exercise D

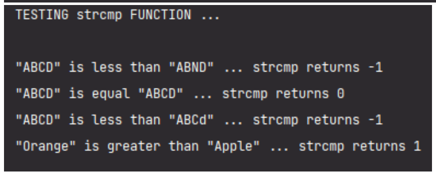
```c
/*
*  File Name: lab4exD.c
*  Assignment: Lab 4 Exercise D
*  Lab section: B04
*  Completed by: Aleksander Berezowski
*  Submission Date: On or before Oct 21, 2021
*/

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

int my_strlen(const char *s);
/*  Duplicates strlen from <string.h>, except return type is int.
 *  REQUIRES
 *     s points to the beginning of a string.
 *  PROMISES
 *     Returns the number of chars in the string, not including the
 *     terminating null.
 */

void my_strncat(char *dest, const char *source, int);
/*  Duplicates strncat from <string.h>, except return type is void.
 *   dest and source point to the beginning of two strings.
 *  PROMISES
 *     appends source to the end of dest. If length of source is more than n.
 *     Only copies the first n elements of source.
 */

int my_strncmp(const char* str1, const char* str2);
/*  Duplicates strcmp from <string.h>, except return type is int.
 *  REQUIRES
 *     str1 points to the beginning of a string, and str2 to the beginning of
 *     another string.
 *  PROMISES
 *     Returns 0 if str1 and str2 are idntical.
 *     Returns a negative number of str1 is less that str2.
 *     Return a psitive nubmer of str2 is less than str1.
 */

int main(void)
{
    char str1[7] = "banana";
    const char str2[] = "-tacit";
    const char* str3 = "-toe";

    char str5[] = "ticket";
    char my_string[100]="";
    int bytes;
    int length;
    int y;

    printf("\nTESTING strlen FUNCTION ... \n");

    /* using strlen function */
    length = (int) my_strlen(my_string);
    printf("\nExpected to display: my_string length is 0.");
    printf("\nmy_string length is %d.", length);

    /* using sizeof operator */
    bytes = sizeof (my_string);
    printf("\nExpected to display: my_string size is 100 bytes.");
    printf("\nmy_string size is %d bytes.", bytes);

    /* using strcpy C libarary function */
    strcpy(my_string, str1);
    printf("\nExpected to display: my_string contains banana.");
    printf("\nmy_string contains %s", my_string);

    length = (int) my_strlen(my_string);
    printf("\nExpected to display: my_string length is 6.");
    printf("\nmy_string length is %d.", length);

    my_string[0] = '\0';
    printf("\nExpected to display: my_string contains \"\".");
    printf("\nmy_string contains:\"%s\"", my_string);

    length = (int) my_strlen(my_string);
    printf("\nExpected to display: my_string length is 0.");
    printf("\nmy_string length is %d.", length);

    bytes = sizeof (my_string);
    printf("\nExpected to display: my_string size is still 100 bytes.");
    printf("\nmy_string size is still %d bytes.", bytes);

    printf("\n\nTESTING strncat FUNCTION ... \n");
    /* strncat append the first 3 characters of str5 to the end of my_string */
    my_strncat(my_string, str5, 3);
    printf("\nExpected to display: my_string contains \"tic\"");
    printf("\nmy_string contains \"%s\"", my_string);

    length = (int) my_strlen(my_string);
    printf("\nExpected to display: my_string length is 3.");
    printf("\nmy_string length is %d.", length);

    my_strncat(my_string, str2,  4);
    printf("\nExpected to display: my_string contains \"tic-tac\"");
    printf("\nmy_string contains:\"%s\"", my_string);

    /* strncat append ONLY up ot '\0' character from str3 -- not 6 characters */
    my_strncat(my_string, str3, 6);
    printf("\nExpected to display: my_string contains \"tic-tac-toe\"");
    printf("\nmy_string contains:\"%s\"", my_string);

    length = (int) my_strlen(my_string);
    printf("\nExpected to display: my_string has 11 characters.");
    printf("\nmy_string has %d characters.", length);

    printf("\n\nUsing strcmp - C library function: ");
    printf("\nExpected to display: \"ABCD\" is less than \"ABCDE\"");
    printf("\n\"ABCD\" is less than \"ABCDE\"", strcmp("ABCD", "ABCDE"));


    printf("\n\nTESTING strcmp FUNCTION ... \n");

    if((y = my_strncmp("ABCD", "ABND")) < 0)
        printf("\n\"ABCD\" is less than \"ABND\" ... strcmp returns %d", y);

    if((y = my_strncmp("ABCD", "ABCD")) == 0)
        printf("\n\"ABCD\" is equal \"ABCD\" ... strcmp returns %d", y);

    if((y = my_strncmp("ABCD", "ABCd")) < 0)
        printf("\n\"ABCD\" is less than \"ABCd\" ... strcmp returns %d", y);

    if((y = my_strncmp("Orange", "Apple")) > 0)
```



```c
        printf("\n\"Orange\" is greater than \"Apple\" ... strcmp returns %d\n", y);

    return 0;
}

int my_strlen(const char *s){
    int counter = 0;
    //While the character at s is a thing, keep increasing the counter
    while(*(s+counter))
        counter++;
    //Above breaks after hitting last character
    return counter;
}

void my_strncat(char *dest, const char *source, int n) {
    int i, j;

    //make i point to the last space (the null thingy)
    for (i = 0; *(dest+i) != '\0'; i++);

    //keep appending to the end of dest as long as j is smaller than n
    for (j = 0; *(source+j) != '\0' && j < n; j++) {
        *(dest +i + j) = *(source + j);
    }

    //set the last letter to the null characters
    *(dest +i + j) = '\0';
}

int my_strncmp(const char* str1, const char* str2) {

    int i = 0;

    while(*(str1+i) || *(str2 + i)){
        if( (int)*(str1+i) > (int)*(str2+i) )
            return 1;
        else if ( (int)*(str1+i) < (int)*(str2+i) )
            return -1;

        i++;
    }
    return 0;

}
```

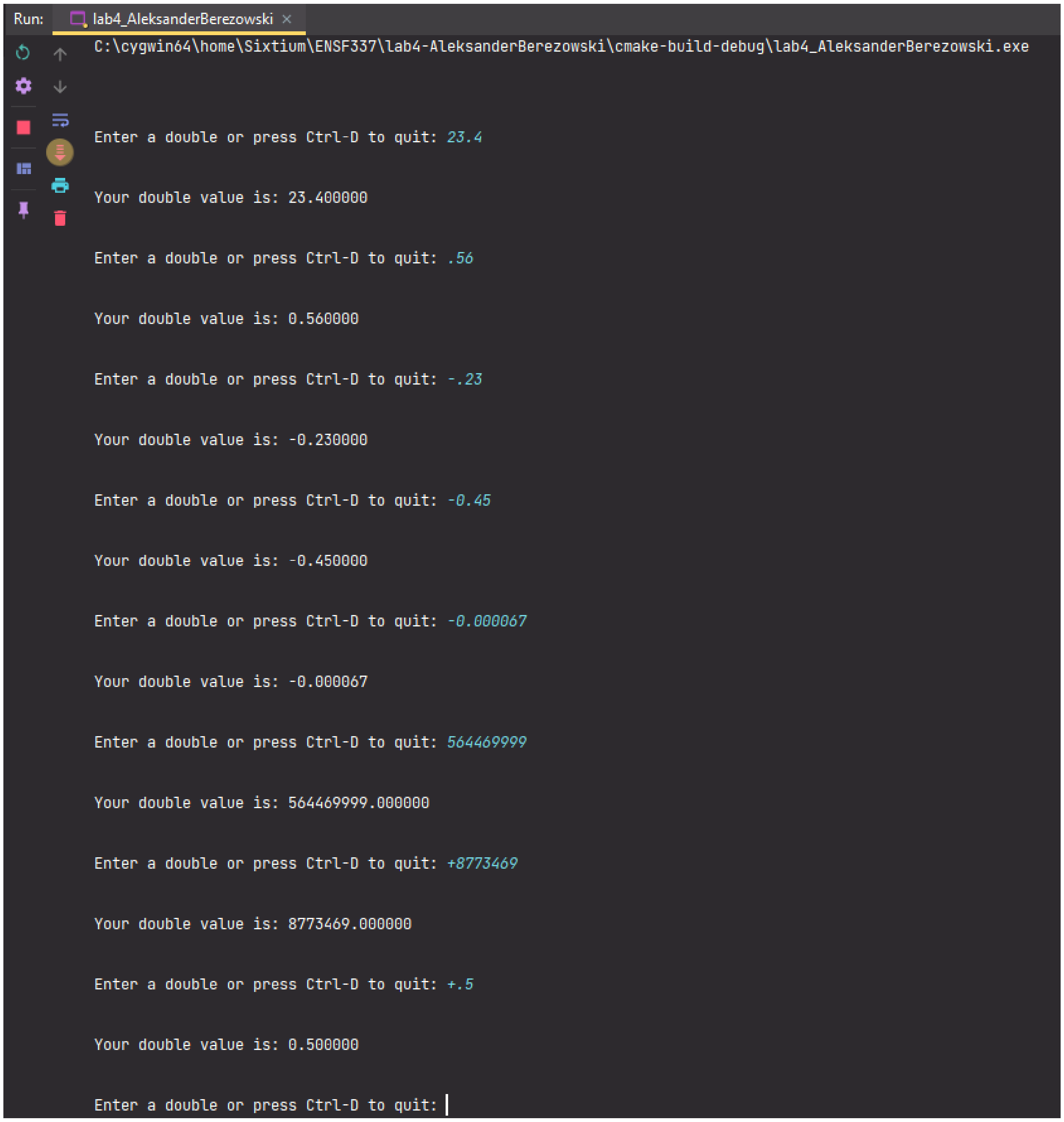


```
C:\cygwin64\home\Sixtium\ENSF337\lab4-AleksanderBerezowski\cmake-build-debug\lab4_AleksanderBerezowski.exe

TESTING strlen FUNCTION ...

Expected to display: my_string length is 0.
my_string length is 0.
Expected to display: my_string size is 100 bytes.
my_string size is 100 bytes.
Expected to display: my_string contains banana.
my_string contains banana
Expected to display: my_string length is 6.
my_string length is 6.
Expected to display: my_string contains "".
my_string contains:""
Expected to display: my_string length is 0.
my_string length is 0.
Expected to display: my_string size is still 100 bytes.
my_string size is still 100 bytes.

TESTING strncat FUNCTION ...

Expected to display: my_string contains "tic"
my_string contains "tic"
Expected to display: my_string length is 3.
my_string length is 3.
Expected to display: my_string contains "tic-tac"
my_string contains:"tic-tac"
Expected to display: my_string contains "tic-tac-toe"
my_string contains:"tic-tac-toe"
Expected to display: my_string has 11 characters.
my_string has 11 characters.

Using strcmp - C library function:
Expected to display: "ABCD" is less than "ABCDE"
"ABCD" is less than "ABCDE"
```



```
TESTING strcmp FUNCTION ...

"ABCD" is less than "ABND" ... strcmp returns -1
"ABCD" is equal "ABCD" ... strcmp returns 0
"ABCD" is less than "ABCd" ... strcmp returns -1
"Orange" is greater than "Apple" ... strcmp returns 1
```

Exercise E



Run: lab4_AleksanderBerezowski

```
C:\cygwin64\home\Sixtium\ENSF337\lab4-AleksanderBerezowski\cmake-build-debug\lab4_AleksanderBerezowski.exe


Enter a double or press Ctrl-D to quit: 23.4

Your double value is: 23.400000

Enter a double or press Ctrl-D to quit: .56

Your double value is: 0.560000

Enter a double or press Ctrl-D to quit: -.23

Your double value is: -0.230000

Enter a double or press Ctrl-D to quit: -0.45

Your double value is: -0.450000

Enter a double or press Ctrl-D to quit: -0.000067

Your double value is: -0.000067

Enter a double or press Ctrl-D to quit: 564469999

Your double value is: 564469999.000000

Enter a double or press Ctrl-D to quit: +8773469

Your double value is: 8773469.000000

Enter a double or press Ctrl-D to quit: +.5

Your double value is: 0.500000

Enter a double or press Ctrl-D to quit:
```