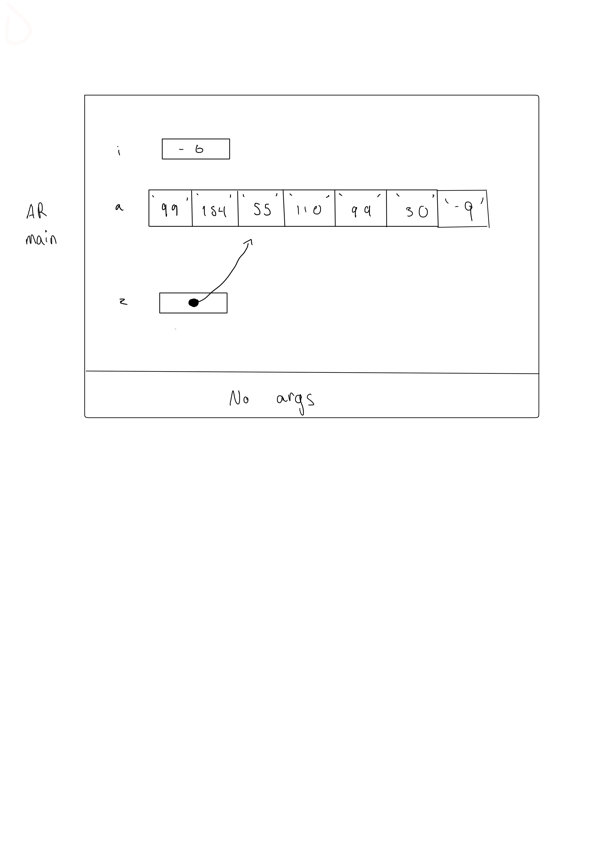
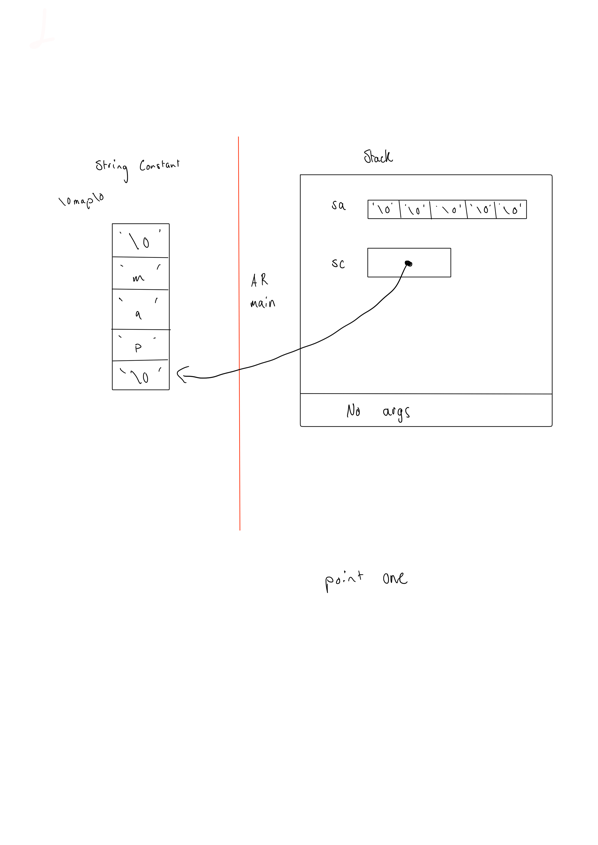
**Course**: Programming Fundamental – ENSF 337  
**Lab #**: Lab 4  
**Instructor:** M. Moussavi  
**Student Name:** Carl Soriano  
**Lab Section:** B01  
**Date submitted:** Oct 5, 2022

Exercise A



Exercise B

Diagram

Description automatically generated

Exercise C

// lab2exC.c

// ENSF 337 Lab 4 Exercise C

//

#include <stdio.h>

#define ELEMENTS(x) (sizeof(x)/sizeof(x[0]))

//indexs x[0] it gives the size of the specific element IE int or double

**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;

}

Text

Description automatically generated

Exercise D

strncat(my\_string, str3, 6);

printf("\nExpected to display: my\_string contains \"tic-tac-toe\"");

printf("\nmy\_string contains:\"%s\"", my\_string);

length = (**int**) 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 returns %d", strcmp("ABCD", "ABCDE"));

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

**if**((y = strcmp("ABCD", "ABND")) < 0)

printf("\n\"ABCD\" is less than \"ABND\" ... strcmp returns %d", y);

**if**((y = strcmp("ABCD", "ABCD")) == 0)

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

**if**((y = strcmp("ABCD", "ABCd")) < 0)

printf("\n\"ABCD\" is less than \"ABCd\" ... strcmp returns %d", y);

**if**((y = strcmp("Orange", "Apple")) > 0)

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

**return** 0;

}

**int** my\_strlen(**const** **char** \*s)

{

**const** **char** \*length = s;

**while** (\*length != '\0')

length++;

**return** (**int**)(length - s);

}

**void** my\_strncat(**char** \*dest, **const** **char** \*source, **int** n)

{

**char** \*temp = dest;

**while**(\*dest){

dest++;

}

**while**(n && \*source){

n--;

\*dest = \*source;

dest++;

source++;

}

\*dest = '\0';

dest = temp;

}

**int** my\_strncmp(**const** **char**\* str1, **const** **char**\* str2)

{

**for**(**int** i = 0; ;i++)

{

**if**(\*str1 == '\0' && \*str2 == '\0')

{

**return** 0;

}

**int** ascii1 = (**int**)\*str1;

**int** ascii2 = (**int**)\*str2;

**if**(ascii1>ascii2)

**return** 1;

**if**(ascii2>ascii1)

**return** -1;

}

}

/\*

\* lab4exD.c

\*

\* ENSF 337 Lab 4 Exercise D

\*

\*/

#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** n);

/\* 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 \*/

strncat(my\_string, str5, 3);

printf("\nExpected to display: my\_string contains \"tic\"");

printf("\nmy\_string contains \"%s\"", my\_string);

length = (**int**) strlen(my\_string);

printf("\nExpected to display: my\_string length is 3.");

printf("\nmy\_string length is %d.", length);

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 \*/

strncat(my\_string, str3, 6);

printf("\nExpected to display: my\_string contains \"tic-tac-toe\"");

printf("\nmy\_string contains:\"%s\"", my\_string);

length = (**int**) 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 returns %d", strcmp("ABCD", "ABCDE"));

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

**if**((y = strcmp("ABCD", "ABND")) < 0)

printf("\n\"ABCD\" is less than \"ABND\" ... strcmp returns %d", y);

**if**((y = strcmp("ABCD", "ABCD")) == 0)

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

**if**((y = strcmp("ABCD", "ABCd")) < 0)

printf("\n\"ABCD\" is less than \"ABCd\" ... strcmp returns %d", y);

**if**((y = strcmp("Orange", "Apple")) > 0)

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

**return** 0;

}

**int** my\_strlen(**const** **char** \*s)

{

**const** **char** \*length = s;

**while** (\*length != '\0')

length++;

**return** (**int**)(length - s);

}

**void** my\_strncat(**char** \*dest, **const** **char** \*source, **int** n)

{

**char** \*temp = dest;

**while**(\*dest){

dest++;

}

**while**(n && \*source){

n--;

\*dest = \*source;

dest++;

source++;

}

\*dest = '\0';

dest = temp;

}

**int** my\_strncmp(**const** **char**\* str1, **const** **char**\* str2)

{

**for**(**int** i = 0; ;i++)

{

**if**(\*str1 == '\0' && \*str2 == '\0')

{

**return** 0;

}

**int** ascii1 = (**int**)\*str1;

**int** ascii2 = (**int**)\*str2;

**if**(ascii1>ascii2)

**return** 1;

**if**(ascii2>ascii1)

**return** -1;

}

}

Output Exercise D

Text

Description automatically generated

Exercise E

**Compiling error**

**Cant compile code**

**Submitted my files via D2L**