Lab Report

Course: ENSF 337 – Programming Fundamentals for Software and Computer

Lab #: 4

Instructor: Dr. Maan Khedr

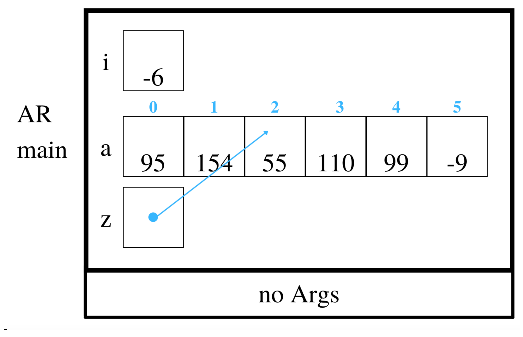
Student Name: Sadia Khandaker

Lab Section: B04

Submission Date: October 19, 2021

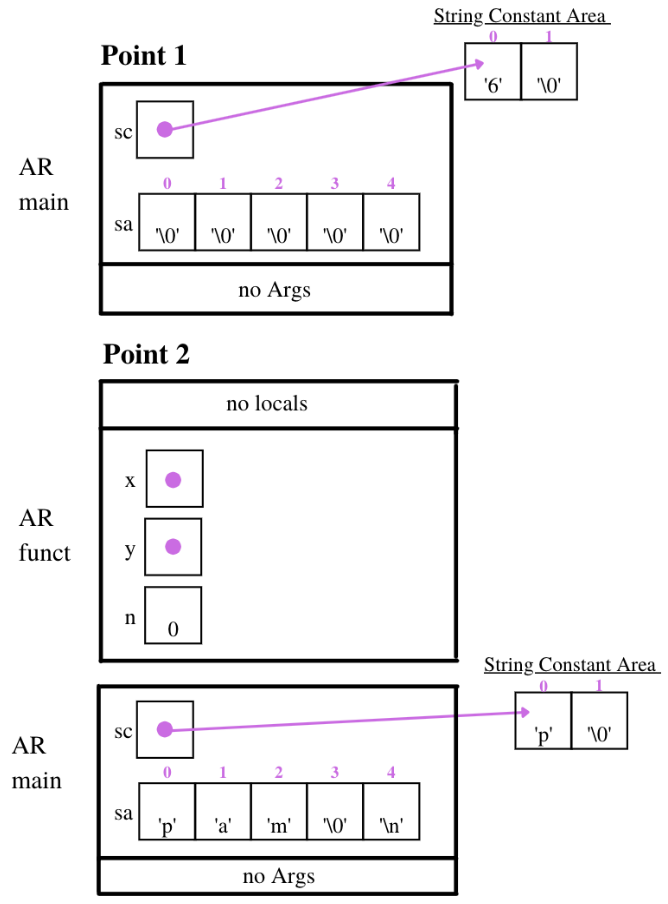
Exercise A: Pointer Arithmetic in C

AR Diagram at Point 1:



Exercise B: More Practice on Pointers and Pointer Arithmetic

AR Diagrams at Point 1 and Point 2:

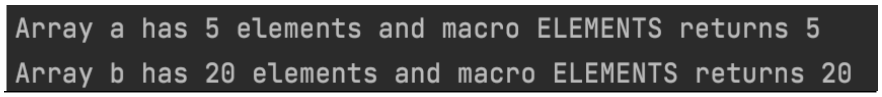


Exercise C: A Simple Macro

Program That Returns the Number of Elements in an Array:

```c
/*
 * File Name: lab4exe_C.c
 * Assignment: Lab 4 Exercise C
 * Lab Section: B04
 * Completed By: Sadia Khandaker
 * Submission Date: Oct 19, 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;
}
```

Output:



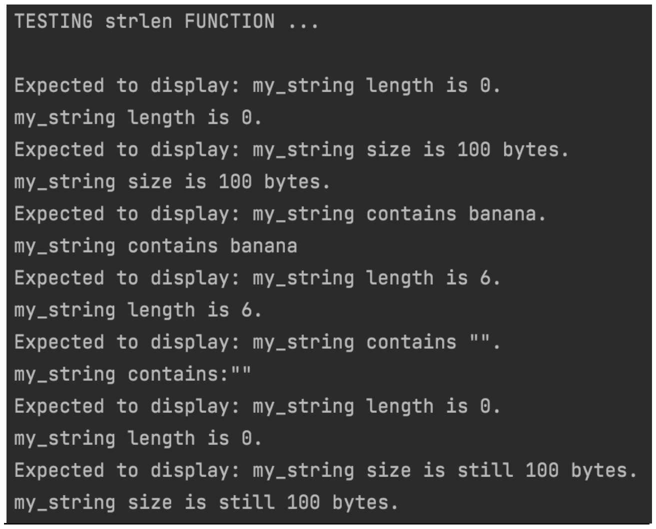
```
Array a has 5 elements and macro ELEMENTS returns 5
Array b has 20 elements and macro ELEMENTS returns 20
```

Exercise D: Duplicating Library Function, Using Pointer Arithmetic

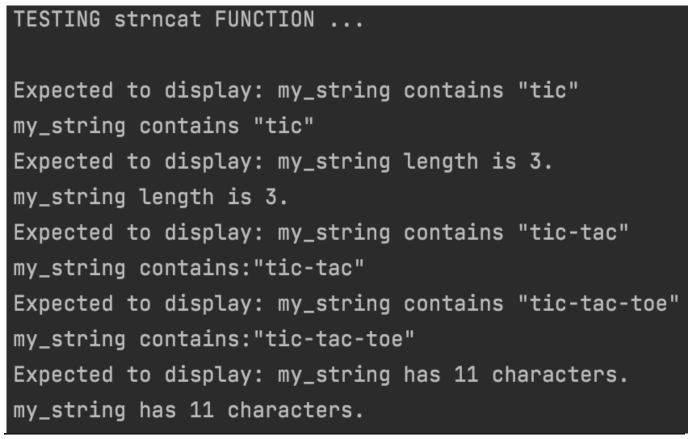
Program to Duplicate Library Function:

```c
/*
 * File Name: lab4exe_D.c
 * Assignment: Lab 4 Exercise D
 * Lab Section: B04
 * Completed By: Sadia Khandaker
 * Submission Date: Oct 19, 2021
*/

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

int my_strlen(const char *s);
void my_strncat(char *dest, const char *source, int);
int my_strncmp(const char* str1, const char* str2);

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)
        printf("\n\"Orange\" is greater than \"Apple\" ... strcmp returns
%d\n", y);
    return 0;
}

int my_strlen(const char *s) {
    int l=0;
    while(*s != 0)
    {
        s++;
        l++;
    }
    return l;
}

void my_strncat(char *dest, const char *source, int i) {
    int j=0;
    while(*dest) {
        dest++;
    }
    while(*source && j<i){
        *dest = *source;
        j++;
        dest++;
        source++;
    }
    *dest = '\0';
}

int my_strncmp(const char* str1, const char* str2) {
    int l1 = my_strlen(str1), l2 = my_strlen(str2), i = 0;
    while (i < l1 && i < l2) {
        if (*(str1 + i) < *(str2 + i))
            return -1;
        else if (*(str1 + i) > *(str2 + i))
            return 1;
        i++;
    }
    if (i == l1 && i == l2)
        return 0;
    else if (l1 < l2)
        return -1;
    else
        return 1;
}
```

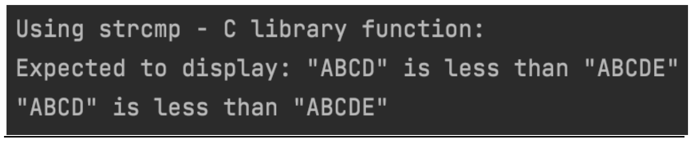
<u>Output:</u>



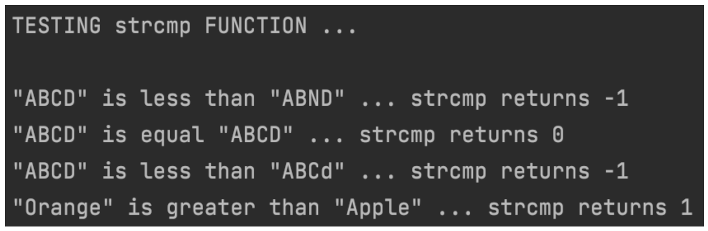
```
TESTING strlen FUNCTION ...

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"
```

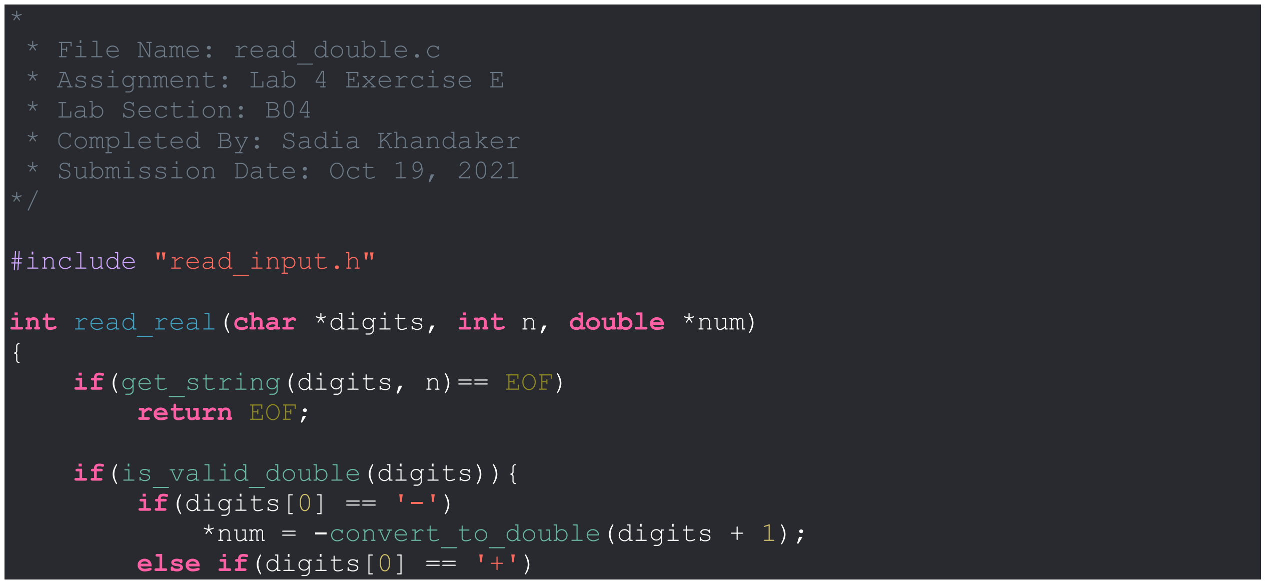


```
TESTING strcmp FUNCTION ...

"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: Reading Numeric Input as a String

Program to Read Numeric Input as a String:

read_double.c

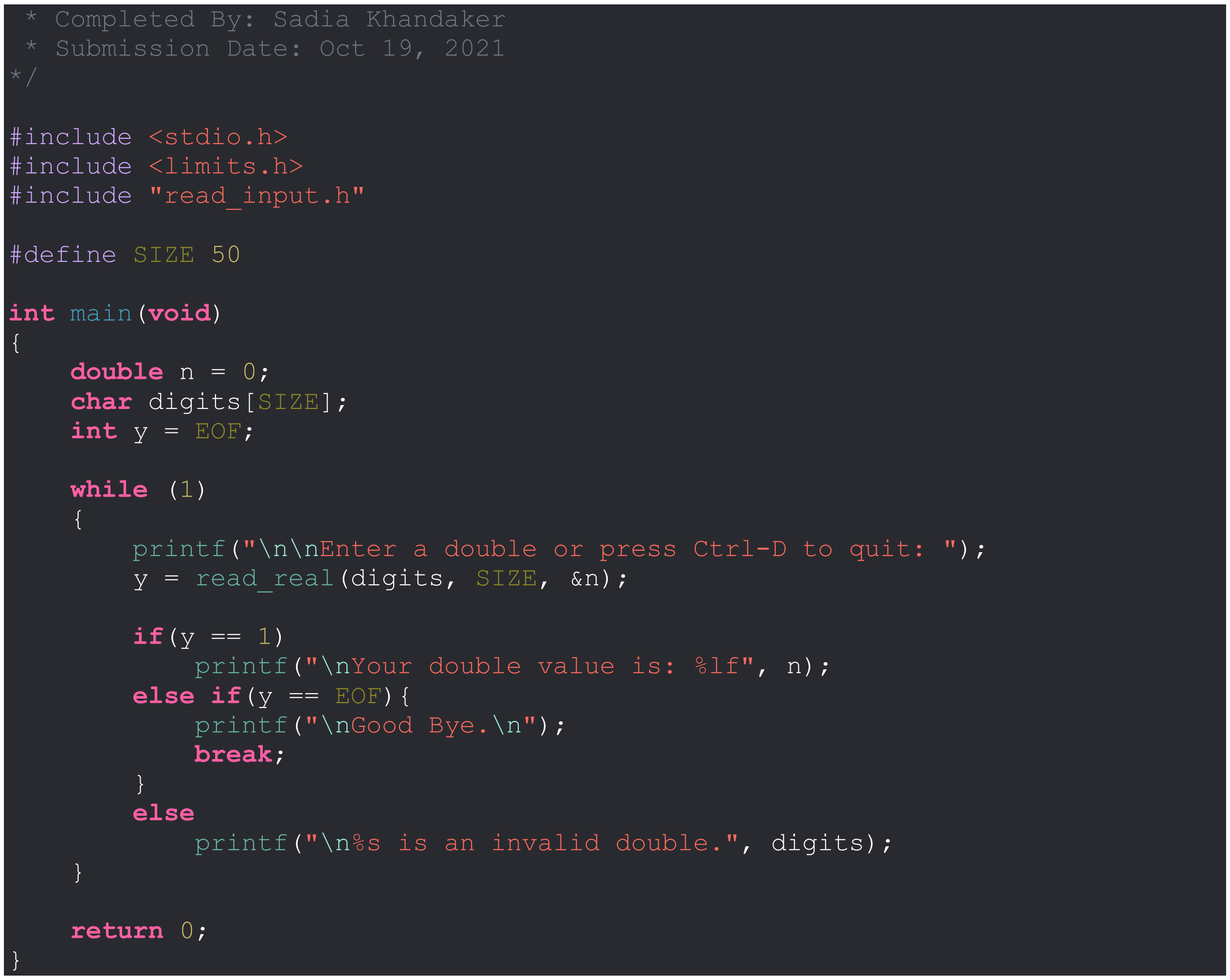


```
*
 * File Name: read_double.c
 * Assignment: Lab 4 Exercise E
 * Lab Section: B04
 * Completed By: Sadia Khandaker
 * Submission Date: Oct 19, 2021
*/

#include "read_input.h"

int read_real(char *digits, int n, double *num)
{
    if(get_string(digits, n)== EOF)
        return EOF;

    if(is_valid_double(digits)){
        if(digits[0] == '-')
            *num = -convert_to_double(digits + 1);
        else if(digits[0] == '+')
```

```c
            *num = convert_to_double(digits + 1);
        else
            *num = convert_to_double(digits);
        return 1;
    }
    return 0;
}


int is_valid_double(const char* digits) {
    int i, valid = 1, dec = 0;
    if(digits[0] == '+' || digits[0] == '-')
        i = 1;
    else
        i = 0;
    if (digits[i] == '\0')
        valid = 0;
    else
        while ((digits[i] != '\0') && valid) {
            if((digits[i] < '0' ||  digits[i] > '9') && (digits[i] != '.' ||
dec > 1))
                valid = 0;
            i++;
            if(digits[i] == '.')
                dec++;
        }
    return valid;
}

double convert_to_double(const char* digits) {
    int i = 0;
    double sum = 0, sum2 = 0;
    while(*digits != '\0') {
        if(digits[i] == '.'){
            while(digits[i] != '\0'){
                digits++;
            }
            digits--;
            while(digits[i] != '.'){
                sum2 = 0.1*sum2 + (digits[i] - '0');
                i--;
            }
            sum2 *= 0.1;
            break;
        }
        sum = 10 * sum + (digits[i] - '0');
        i++;
    }
    sum += sum2;
    return sum;
```

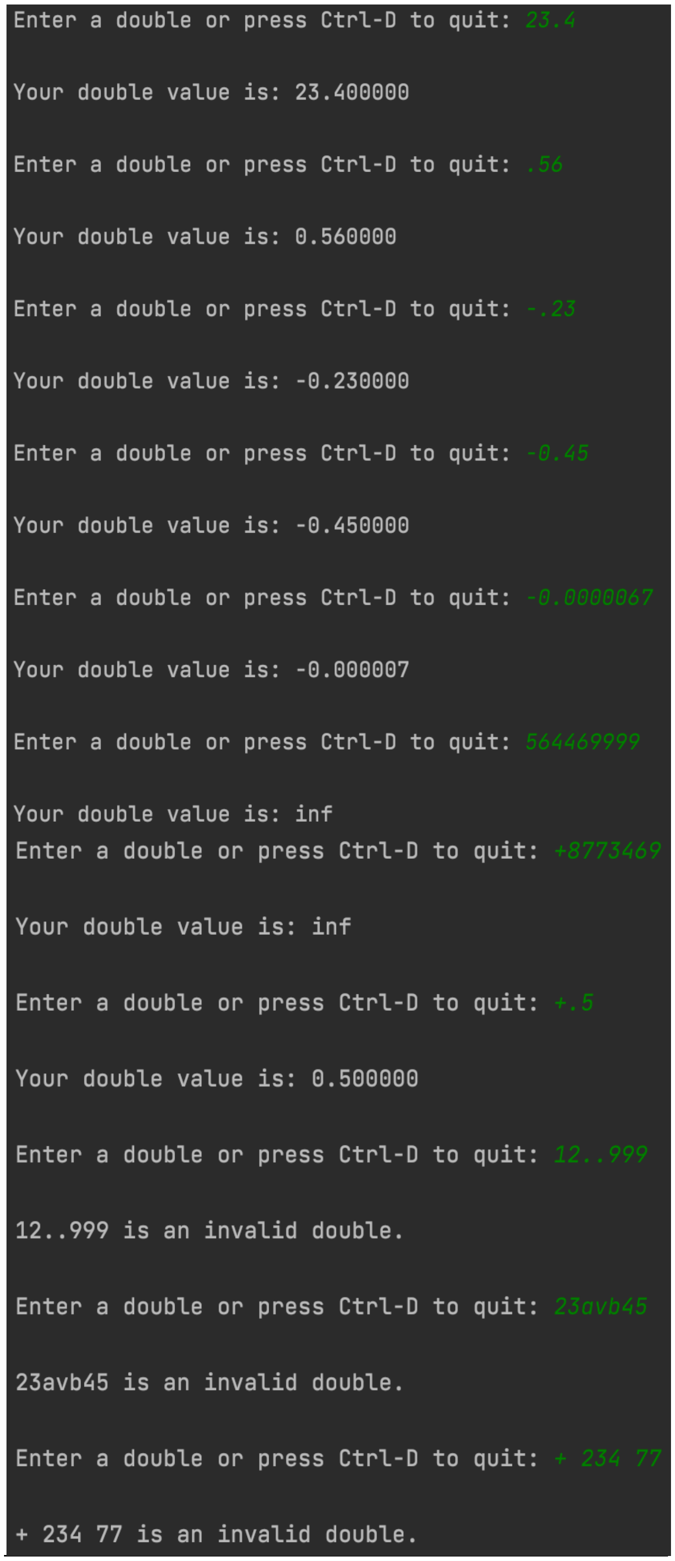
prog_two.c

```c
/*
 * File Name: prog_two.c
 * Assignment: Lab 4 Exercise E
 * Lab Section: B04
```



```c
 * Completed By: Sadia Khandaker
 * Submission Date: Oct 19, 2021
*/

#include <stdio.h>
#include <limits.h>
#include "read_input.h"

#define SIZE 50

int main(void)
{
    double n = 0;
    char digits[SIZE];
    int y = EOF;

    while (1)
    {
        printf("\n\nEnter a double or press Ctrl-D to quit: ");
        y = read_real(digits, SIZE, &n);

        if(y == 1)
            printf("\nYour double value is: %lf", n);
        else if(y == EOF){
            printf("\nGood Bye.\n");
            break;
        }
        else
            printf("\n%s is an invalid double.", digits);
    }

    return 0;
}
```

Output:



```
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: -0.0000067

Your double value is: -0.000007

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

Your double value is: inf
Enter a double or press Ctrl-D to quit: +8773469

Your double value is: inf

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: 12..999

12..999 is an invalid double.

Enter a double or press Ctrl-D to quit: 23avb45

23avb45 is an invalid double.

Enter a double or press Ctrl-D to quit: + 234 77

+ 234 77 is an invalid double.
```