**W7 -** PRACTICE

*Strings*

## *At the end of this practice, you should be able to…*

* Input **strings** and **list of strings**
* Manipulate **string with functions**
* Implement functions like strcmp, strcat, atoi, strlen, tolower, toupper
* Applying the **Top to Bottom approach** to string-based problems

## *How do we structure exercises?*

We organize this practice into 4 parts:

| ANALYSE | **Understand** existing codes, find the **bugs** or **complete** missing gaps |
| --- | --- |
| MANIPULATE | Ensure you can **apply the theory** with some basic challenges |
| CREATE | **Express your creativity** with more complex challenges |

## *Are you lost?*

You can read the following documentation to be ready for this practice

Strings

<https://www.w3schools.com/c/c_strings.php>

String functions

<https://www.w3schools.com/c/c_strings_functions.php>

**ANALYSE**

**EX 1** **(From Code to Memory Dump)**

Look at the bellow code.

char text[] = "game";

char newText[5];

strcpy(newText, text);

**QUESTION –** Complete the memory dump **at the end** of the code execution [(ASCCI TABLE HERE)](https://www.ascii-code.com/)

| ADDRESS | CONTENT | COMMENT |
| --- | --- | --- |
| 100 | 103 | ‘g’ |
| 101 | 97 | ‘a’ |
| 102 | 109 | ‘m’ |
| 103 | 101 | ‘e’ |
| 104 | 0 | ‘\0’ |
| 105 | 103 | ‘g’ |
| 106 | 97 | ‘a’ |
| 107 | 109 | ‘m’ |
| 108 | 101 | ‘e’ |
| 109 | 0 | ‘\0’ |
| 110 |  |  |
| 111 |  |  |
| 112 |  |  |

**EX 2** **(Fix Buggy Code)**

Look at the bellow code.

char name[10];

printf("Enter your name: ");

scanf("%s", name);

printf("Hello, %s\n", name);

**Q1 –** What’s happen if the user enters a name with more than 9 characters?

the compiler will say sagmentation fault core dumped

**EX 3** **(Fix Buggy Code)**

Look at the bellow code.

char names[3][10] = {"Alice", "Bob", "Eve"};

names[1] = "Charlie"; // Bug here !!!

printf("%s\n", names[1]);

**Q1 –** The second line has a bug. Explain **why**

because names[1] is a block of memory not an address of the memory that we can reassign.

*Note that names[1] is an array !*

*Can we reassign an array? As an example, can we do this: int numbers[] = {1,2,3} ; numbers = {4,5,6}*

**Q2 –** Fix the bug using the strcpy() function to properly copy the string into the existing slot

#include <stdio.h>

#include <string.h>

int main(){

char names[3][10] = {"Alice", "Bob", "Eve"};

strcpy(names[1], "Charlie");

for(int i = 0 ; i < 3; i++){

printf("%s\n", names[i]);

}

return 0 ;

}

**MANIPULATE**

**EX 1 (*COMPUTE A STRING LENGTH*)**

**Q1 -** Write a function in C that takes a string as input and the number of characters (*excluding the null terminator*).

| FUNCTION NAME | lengthOf | |
| --- | --- | --- |
| FUNCTION DESCRIPTION | Get the number of characters in given string | |
| PARAMETERS | const char\* | The string |
| RETURN | void | the number of characters in given string |

**Example of usage:**

int size = lengthOf("abcd"); // size = 4

#include <stdio.h>

int lengthOf(const char\* string){

int i = 0;

while(string[i] != '\0'){

i++ ;

}

return i;

}

int main(){

int size = lengthOf("abcd");

printf("%d\n", size);

return 0 ;

}

**Requirements:**

* You cannot use strlen from <string.h>

**Q2 -** Write a C program (main) that performs the following tasks:

1. Reads the user's full name from console using a character array.
2. Displays the number of characters in the name (*use the previous defined function*).

**Example of usage**

Enter your name: Ronan

Your name has 5 characters

#include <stdio.h>

int lengthOf(const char\* string){

int i = 0;

while(string[i] != '\0'){

i++ ;

}

return i;

}

int main(){

char name[10];

printf("Enter your name : ");

scanf(" %s", name);

int size = lengthOf(name);

printf("Your name has %d characters \n", size);

return 0 ;

}

**EX 2 (*COUNT VOWELS*)**

Write a function that takes a string as input and returns the number of vowels in the string (both uppercase and lowercase).

| FUNCTION NAME | countVowels | |
| --- | --- | --- |
| FUNCTION DESCRIPTION | Get the number of vowels in the string | |
| PARAMETERS | const char\* | The string |
| RETURN | void | the number of vowels in the string |

**Example of usage:**

int number = countVowels("who is the bEst?"); // 4 vowels

**Requirements:**

* Count vowels: a, e, i, o, u (both uppercase and lowercase).

#include <stdio.h>

#include <ctype.h>

void countVowels(const char\* string, int \*count){

for(int i = 0; string[i] != '\0'; i++){

int s = tolower(string[i]);

if (s == 'a' || s == 'e' || s == 'i' || s == 'o' || s == 'u') \*count += 1;

}

}

int main(){

char string[100];

int count = 0;

fgets(string, sizeof(string), stdin);

countVowels(string, &count);

printf("%d\n", count);

return 0 ;

}

**EX 3 (*REVERSE A STRING*)**

Write a function that reverses the given string in place (i.e., without using a second array).

| FUNCTION NAME | reverseString | |
| --- | --- | --- |
| FUNCTION DESCRIPTION | Reverse the given string | |
| PARAMETERS | char\* | The string |
| RETURN | void |  |

**Example of usage**

char text[] = "CADT

reverseString(text);

print("%s", text); // TDAC

#include <stdio.h>

int lengthOf(const char\* string){

int i = 0;

while(string[i] != '\0'){

i++ ;

}

return i;

}

void reverseString(char\* string){

int length = lengthOf(string);

for(int i = 0; i < length/2; i++){

char temp = string[i];

string[i] = string[length - 1 - i];

string[length - 1 - i] = temp;

}

}

int main(){

char string[10];

printf("Enter your word : ");

scanf(" %s", string);

reverseString(string);

printf("Reversed string : %s\n", string);

return 0;

}

**EX 4 (*TO UPPER FUNCTION*)**

Write a function in C that takes a string as input and converts all its lowercase letters to uppercase.

The function should modify the string in place.

| FUNCTION NAME | toUpper | |
| --- | --- | --- |
| FUNCTION DESCRIPTION | Convert all lowercase letters in given string to uppercase | |
| PARAMETERS | char\* | The string |
| RETURN | void |  |

**Example of usage**

char text[] = "Ronan is The Best!";

toUpper(text);

print("%s", text); // RONAN IS THE BEST!

| FUNCTION NAME | toUpper |
| --- | --- |

**Requirements:**

* Use the toupper() function from <ctype.h> ([see the documentation](https://www.programiz.com/c-programming/library-function/ctype.h/toupper))
* The function should work on any valid null-terminated string.

#include <stdio.h>

#include <ctype.h>

void toUpper(char\* string){

for(int i = 0; string[i] != '\0'; i++){

if(islower(string[i])){

string[i] = toupper(string[i]);

}

}

}

int main(){

char string[10];

printf("Enter your word : ");

scanf(" %s", string);

toUpper(string);

printf("Updated string : %s\n", string);

return 0;

}