0x06. C - More pointers, arrays and strings

**C**

* By: Julien Barbier
* Weight: 1
* Project over - took place from Sep 21, 2022 6:00 AM to Sep 23, 2022 6:00 AM
* An auto review will be launched at the deadline

In a nutshell…

* **Auto QA review:** 47.0/57 mandatory & 0.0/46 optional
* **Altogether:**  **82.46%**
  + Mandatory: 82.46%
  + Optional: 0.0%
  + Calculation:  82.46% + (82.46% \* 0.0%)  == **82.46%**



Learning Objectives

At the end of this project, you are expected to be able to [explain to anyone](https://intranet.alxswe.com/rltoken/tkwwPs3MT3JT07FSsmXy-A), **without the help of Google**:

General

* What are pointers and how to use them
* What are arrays and how to use them
* What are the differences between pointers and arrays
* How to use strings and how to manipulate them
* Scope of variables

Copyright - Plagiarism

* You are tasked to come up with solutions for the tasks below yourself to meet with the above learning objectives.
* You will not be able to meet the objectives of this or any following project by copying and pasting someone else’s work.
* You are not allowed to publish any content of this project.
* Any form of plagiarism is strictly forbidden and will result in removal from the program.

Requirements

General

* Allowed editors: vi, vim, emacs
* All your files will be compiled on Ubuntu 20.04 LTS using gcc, using the options -Wall -Werror -Wextra -pedantic -std=gnu89
* All your files should end with a new line
* A README.md file, at the root of the folder of the project is mandatory
* Your code should use the Betty style. It will be checked using [betty-style.pl](https://github.com/holbertonschool/Betty/blob/master/betty-style.pl) and [betty-doc.pl](https://github.com/holbertonschool/Betty/blob/master/betty-doc.pl)
* You are not allowed to use global variables
* No more than 5 functions per file
* You are not allowed to use the standard library. Any use of functions like printf, puts, etc… is forbidden
* You are allowed to use [\_putchar](https://github.com/holbertonschool/_putchar.c/blob/master/_putchar.c)
* You don’t have to push \_putchar.c, we will use our file. If you do it won’t be taken into account
* In the following examples, the main.c files are shown as examples. You can use them to test your functions, but you don’t have to push them to your repo (if you do we won’t take them into account). We will use our own main.c files at compilation. Our main.c files might be different from the one shown in the examples
* The prototypes of all your functions and the prototype of the function \_putchar should be included in your header file called main.h
* Don’t forget to push your header file

Quiz questions

**Great!** You've completed the quiz successfully! Keep going! (Show quiz)

Tasks

0. strcat

**mandatory**

Score: 100.0% (*Checks completed: 100.0%*)

Write a function that concatenates two strings.

* Prototype: char \*\_strcat(char \*dest, char \*src);
* This function appends the src string to the dest string, overwriting the terminating null byte (\0) at the end of dest, and then adds a terminating null byte
* Returns a pointer to the resulting string dest

FYI: The standard library provides a similar function: strcat. Run man strcat to learn more.

julien@ubuntu:~/0x06$ cat 0-main.c

#include "main.h"

#include <stdio.h>

/\*\*

\* main - check the code

\*

\* Return: Always 0.

\*/

int main(void)

{

char s1[98] = "Hello ";

char s2[] = "World!\n";

char \*ptr;

printf("%s\n", s1);

printf("%s", s2);

ptr = \_strcat(s1, s2);

printf("%s", s1);

printf("%s", s2);

printf("%s", ptr);

return (0);

}

julien@ubuntu:~/0x06$ gcc -Wall -pedantic -Werror -Wextra -std=gnu89 0-main.c 0-strcat.c -o 0-strcat

julien@ubuntu:~/0x06$ ./0-strcat

Hello

World!

Hello World!

World!

Hello World!

julien@ubuntu:~/0x06$

**Repo:**

* GitHub repository: alx-low\_level\_programming
* Directory: 0x06-pointers\_arrays\_strings
* File: 0-strcat.c

 Done! Help Check your code Get a sandbox QA Review

1. strncat

**mandatory**

Score: 85.71% (*Checks completed: 85.71%*)

Write a function that concatenates two strings.

* Prototype: char \*\_strncat(char \*dest, char \*src, int n);
* The \_strncat function is similar to the \_strcat function, except that
  + it will use at most n bytes from src; and
  + src does not need to be null-terminated if it contains n or more bytes
* Return a pointer to the resulting string dest

FYI: The standard library provides a similar function: strncat. Run man strncat to learn more.

julien@ubuntu:~/0x06$ cat 1-main.c

#include "main.h"

#include <stdio.h>

/\*\*

\* main - check the code

\*

\* Return: Always 0.

\*/

int main(void)

{

char s1[98] = "Hello ";

char s2[] = "World!\n";

char \*ptr;

printf("%s\n", s1);

printf("%s", s2);

ptr = \_strncat(s1, s2, 1);

printf("%s\n", s1);

printf("%s", s2);

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

ptr = \_strncat(s1, s2, 1024);

printf("%s", s1);

printf("%s", s2);

printf("%s", ptr);

return (0);

}

julien@ubuntu:~/0x06$ gcc -Wall -pedantic -Werror -Wextra -std=gnu89 1-main.c 1-strncat.c -o 1-strncat

julien@ubuntu:~/0x06$ ./1-strncat

Hello

World!

Hello W

World!

Hello W

Hello WWorld!

World!

Hello WWorld!

julien@ubuntu:~/0x06$

**Repo:**

* GitHub repository: alx-low\_level\_programming
* Directory: 0x06-pointers\_arrays\_strings
* File: 1-strncat.c

 Done? Help Check your code Ask for a new correction Get a sandbox QA Review

2. strncpy

**mandatory**

Score: 87.5% (*Checks completed: 87.5%*)

Write a function that copies a string.

* Prototype: char \*\_strncpy(char \*dest, char \*src, int n);
* Your function should work exactly like strncpy

FYI: The standard library provides a similar function: strncpy. Run man strncpy to learn more.

julien@ubuntu:~/0x06$ cat 2-main.c

#include "main.h"

#include <stdio.h>

/\*\*

\* main - check the code

\*

\* Return: Always 0.

\*/

int main(void)

{

char s1[98];

char \*ptr;

int i;

for (i = 0; i < 98 - 1; i++)

{

s1[i] = '\*';

}

s1[i] = '\0';

printf("%s\n", s1);

ptr = \_strncpy(s1, "First, solve the problem. Then, write the code\n", 5);

printf("%s\n", s1);

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

ptr = \_strncpy(s1, "First, solve the problem. Then, write the code\n", 90);

printf("%s", s1);

printf("%s", ptr);

for (i = 0; i < 98; i++)

{

if (i % 10)

{

printf(" ");

}

if (!(i % 10) && i)

{

printf("\n");

}

printf("0x%02x", s1[i]);

}

printf("\n");

return (0);

}

julien@ubuntu:~/0x06$ gcc -Wall -pedantic -Werror -Wextra -std=gnu89 2-main.c 2-strncpy.c -o 2-strncpy

julien@ubuntu:~/0x06$ ./2-strncpy

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

First\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

First\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

First, solve the problem. Then, write the code

First, solve the problem. Then, write the code

0x46 0x69 0x72 0x73 0x74 0x2c 0x20 0x73 0x6f 0x6c

0x76 0x65 0x20 0x74 0x68 0x65 0x20 0x70 0x72 0x6f

0x62 0x6c 0x65 0x6d 0x2e 0x20 0x54 0x68 0x65 0x6e

0x2c 0x20 0x77 0x72 0x69 0x74 0x65 0x20 0x74 0x68

0x65 0x20 0x63 0x6f 0x64 0x65 0x0a 0x00 0x00 0x00

0x00 0x00 0x00 0x00 0x00 0x00 0x00 0x00 0x00 0x00

0x00 0x00 0x00 0x00 0x00 0x00 0x00 0x00 0x00 0x00

0x00 0x00 0x00 0x00 0x00 0x00 0x00 0x00 0x00 0x00

0x00 0x00 0x00 0x00 0x00 0x00 0x00 0x00 0x00 0x00

0x2a 0x2a 0x2a 0x2a 0x2a 0x2a 0x2a 0x00

julien@ubuntu:~/0x06$

**Repo:**

* GitHub repository: alx-low\_level\_programming
* Directory: 0x06-pointers\_arrays\_strings
* File: 2-strncpy.c

 Done! Help Check your code Ask for a new correction Get a sandbox QA Review

3. strcmp

**mandatory**

Score: 85.71% (*Checks completed: 85.71%*)

Write a function that compares two strings.

* Prototype: int \_strcmp(char \*s1, char \*s2);
* Your function should work exactly like strcmp

FYI: The standard library provides a similar function: strcmp. Run man strcmp to learn more.

julien@ubuntu:~/0x06$ cat 3-main.c

#include "main.h"

#include <stdio.h>

/\*\*

\* main - check the code

\*

\* Return: Always 0.

\*/

int main(void)

{

char s1[] = "Hello";

char s2[] = "World!";

printf("%d\n", \_strcmp(s1, s2));

printf("%d\n", \_strcmp(s2, s1));

printf("%d\n", \_strcmp(s1, s1));

return (0);

}

julien@ubuntu:~/0x06$ gcc -Wall -pedantic -Werror -Wextra -std=gnu89 3-main.c 3-strcmp.c -o 3-strcmp

julien@ubuntu:~/0x06$ ./3-strcmp

-15

15

0

julien@ubuntu:~/0x06$

**Repo:**

* GitHub repository: alx-low\_level\_programming
* Directory: 0x06-pointers\_arrays\_strings
* File: 3-strcmp.c

 Done! Help Check your code Ask for a new correction Get a sandbox QA Review

4. I am a kind of paranoid in reverse. I suspect people of plotting to make me happy

**mandatory**

Score: 85.71% (*Checks completed: 85.71%*)

Write a function that reverses the content of an array of integers.

* Prototype: void reverse\_array(int \*a, int n);
* Where n is the number of elements of the array

julien@ubuntu:~/0x06$ cat 4-main.c

#include "main.h"

#include <stdio.h>

/\*\*

\* main - check the code

\* @a: an array of integers

\* @n: the number of elements to swap

\*

\* Return: nothing.

\*/

void print\_array(int \*a, int n)

{

int i;

i = 0;

while (i < n)

{

if (i != 0)

{

printf(", ");

}

printf("%d", a[i]);

i++;

}

printf("\n");

}

/\*\*

\* main - check the code

\*

\* Return: Always 0.

\*/

int main(void)

{

int a[] = {0, 1, 2, 3, 4, 5, 6, 7, 8, 9, 98, 1024, 1337};

print\_array(a, sizeof(a) / sizeof(int));

reverse\_array(a, sizeof(a) / sizeof(int));

print\_array(a, sizeof(a) / sizeof(int));

return (0);

}

julien@ubuntu:~/0x06$ gcc -Wall -pedantic -Werror -Wextra -std=gnu89 4-main.c 4-rev\_array.c -o 4-rev\_array

julien@ubuntu:~/0x06$ ./4-rev\_array

0, 1, 2, 3, 4, 5, 6, 7, 8, 9, 98, 1024, 1337

1337, 1024, 98, 9, 8, 7, 6, 5, 4, 3, 2, 1, 0

julien@ubuntu:~/0x06$

**Repo:**

* GitHub repository: alx-low\_level\_programming
* Directory: 0x06-pointers\_arrays\_strings
* File: 4-rev\_array.c

 Done! Help Check your code Ask for a new correction Get a sandbox QA Review

5. Always look up

**mandatory**

Score: 85.71% (*Checks completed: 85.71%*)

Write a function that changes all lowercase letters of a string to uppercase.

* Prototype: char \*string\_toupper(char \*);

julien@ubuntu:~/0x06$ cat 5-main.c

#include "main.h"

#include <stdio.h>

/\*\*

\* main - check the code

\*

\* Return: Always 0.

\*/

int main(void)

{

char str[] = "Look up!\n";

char \*ptr;

ptr = string\_toupper(str);

printf("%s", ptr);

printf("%s", str);

return (0);

}

julien@ubuntu:~/0x06$ gcc -Wall -pedantic -Werror -Wextra -std=gnu89 5-main.c 5-string\_toupper.c -o 5-string\_toupper

julien@ubuntu:~/0x06$ ./5-string\_toupper

LOOK UP!

LOOK UP!

julien@ubuntu:~/0x06$

**Repo:**

* GitHub repository: alx-low\_level\_programming
* Directory: 0x06-pointers\_arrays\_strings
* File: 5-string\_toupper.c

 Done? Help Check your code Ask for a new correction Get a sandbox QA Review

6. Expect the best. Prepare for the worst. Capitalize on what comes

**mandatory**

Score: 71.43% (*Checks completed: 71.43%*)

Write a function that capitalizes all words of a string.

* Prototype: char \*cap\_string(char \*);
* Separators of words: space, tabulation, new line, ,, ;, ., !, ?, ", (, ), {, and }

julien@ubuntu:~/0x06$ cat 6-main.c

#include "main.h"

#include <stdio.h>

/\*\*

\* main - check the code

\*

\* Return: Always 0.

\*/

int main(void)

{

char str[] = "Expect the best. Prepare for the worst. Capitalize on what comes.\nhello world! hello-world 0123456hello world\thello world.hello world\n";

char \*ptr;

ptr = cap\_string(str);

printf("%s", ptr);

printf("%s", str);

return (0);

}

julien@ubuntu:~/0x06$ gcc -Wall -pedantic -Werror -Wextra -std=gnu89 6-main.c 6-cap\_string.c -o 6-cap

julien@ubuntu:~/0x06$ ./6-cap

Expect The Best. Prepare For The Worst. Capitalize On What Comes.

Hello World! Hello-world 0123456hello World Hello World.Hello World

Expect The Best. Prepare For The Worst. Capitalize On What Comes.

Hello World! Hello-world 0123456hello World Hello World.Hello World

julien@ubuntu:~/0x06$

**Repo:**

* GitHub repository: alx-low\_level\_programming
* Directory: 0x06-pointers\_arrays\_strings
* File: 6-cap\_string.c

 Done? Help Check your code Ask for a new correction Get a sandbox QA Review

7. Mozart composed his music not for the elite, but for everybody

**mandatory**

Score: 57.14% (*Checks completed: 57.14%*)

Write a function that encodes a string into [1337](https://intranet.alxswe.com/rltoken/9v9KfpvWnL0GoMu5mozbug).

* Letters a and A should be replaced by 4
* Letters e and E should be replaced by 3
* Letters o and O should be replaced by 0
* Letters t and T should be replaced by 7
* Letters l and L should be replaced by 1
* Prototype: char \*leet(char \*);
* You can only use one if in your code
* You can only use two loops in your code
* You are not allowed to use switch
* You are not allowed to use any ternary operation

julien@ubuntu:~/0x06$ cat 7-main.c

#include "main.h"

#include <stdio.h>

/\*\*

\* main - check the code for

\*

\* Return: Always 0.

\*/

int main(void)

{

char s[] = "Expect the best. Prepare for the worst. Capitalize on what comes.\n";

char \*p;

p = leet(s);

printf("%s", p);

printf("%s", s);

return (0);

}

julien@ubuntu:~/0x06$ gcc -Wall -pedantic -Werror -Wextra -std=gnu89 7-main.c 7-leet.c -o 7-1337

julien@ubuntu:~/0x06$ ./7-1337

3xp3c7 7h3 b3s7. Pr3p4r3 f0r 7h3 w0rs7. C4pi741iz3 0n wh47 c0m3s.

3xp3c7 7h3 b3s7. Pr3p4r3 f0r 7h3 w0rs7. C4pi741iz3 0n wh47 c0m3s.

julien@ubuntu:~/0x06$

**Repo:**

* GitHub repository: alx-low\_level\_programming
* Directory: 0x06-pointers\_arrays\_strings
* File: 7-leet.c

 Done? Help Check your code Ask for a new correction Get a sandbox QA Review

8. rot13

**#advanced**

Score: 0.0% (*Checks completed: 0.0%*)

Write a function that encodes a string using [rot13](https://intranet.alxswe.com/rltoken/YRxmNA7BnP6yZhl09TKX3A).

* Prototype: char \*rot13(char \*);
* You can only use if statement once in your code
* You can only use two loops in your code
* You are not allowed to use switch
* You are not allowed to use any ternary operation

julien@ubuntu:~/0x06$ cat 100-main.c

#include "main.h"

#include <stdio.h>

/\*\*

\* main - check the code

\*

\* Return: Always 0.

\*/

int main(void)

{

char s[] = "ROT13 (\"rotate by 13 places\", sometimes hyphenated ROT-13) is a simple letter substitution cipher.\n";

char \*p;

p = rot13(s);

printf("%s", p);

printf("------------------------------------\n");

printf("%s", s);

printf("------------------------------------\n");

p = rot13(s);

printf("%s", p);

printf("------------------------------------\n");

printf("%s", s);

printf("------------------------------------\n");

p = rot13(s);

printf("%s", p);

printf("------------------------------------\n");

printf("%s", s);

return (0);

}

julien@ubuntu:~/0x06$ gcc -Wall -pedantic -Werror -Wextra -std=gnu89 100-main.c 100-rot13.c -o 100-rot13

julien@ubuntu:~/0x06$ ./100-rot13

EBG13 ("ebgngr ol 13 cynprf", fbzrgvzrf ulcurangrq EBG-13) vf n fvzcyr yrggre fhofgvghgvba pvcure.

------------------------------------

EBG13 ("ebgngr ol 13 cynprf", fbzrgvzrf ulcurangrq EBG-13) vf n fvzcyr yrggre fhofgvghgvba pvcure.

------------------------------------

ROT13 ("rotate by 13 places", sometimes hyphenated ROT-13) is a simple letter substitution cipher.

------------------------------------

ROT13 ("rotate by 13 places", sometimes hyphenated ROT-13) is a simple letter substitution cipher.

------------------------------------

EBG13 ("ebgngr ol 13 cynprf", fbzrgvzrf ulcurangrq EBG-13) vf n fvzcyr yrggre fhofgvghgvba pvcure.

------------------------------------

EBG13 ("ebgngr ol 13 cynprf", fbzrgvzrf ulcurangrq EBG-13) vf n fvzcyr yrggre fhofgvghgvba pvcure.

julien@ubuntu:~/0x06$

**Repo:**

* GitHub repository: alx-low\_level\_programming
* Directory: 0x06-pointers\_arrays\_strings
* File: 100-rot13.c

 Done? Help Check your code Ask for a new correction Get a sandbox QA Review

9. Numbers have life; they're not just symbols on paper

**#advanced**

Score: 0.0% (*Checks completed: 0.0%*)

Write a function that prints an integer.

* Prototype: void print\_number(int n);
* You can only use \_putchar function to print
* You are not allowed to use long
* You are not allowed to use arrays or pointers
* You are not allowed to hard-code special values

julien@ubuntu:~/0x06$ cat 101-main.c

#include "main.h"

/\*\*

\* main - check the code

\*

\* Return: Always 0.

\*/

int main(void)

{

print\_number(98);

\_putchar('\n');

print\_number(402);

\_putchar('\n');

print\_number(1024);

\_putchar('\n');

print\_number(0);

\_putchar('\n');

print\_number(-98);

\_putchar('\n');

return (0);

}

julien@ubuntu:~/0x06$ gcc -Wall -pedantic -Werror -Wextra -std=gnu89 \_putchar.c 101-main.c 101-print\_number.c -o 101-print\_numbers

julien@ubuntu:~/0x06$ ./101-print\_numbers

98

402

1024

0

-98

julien@ubuntu:~/0x06$

**Repo:**

* GitHub repository: alx-low\_level\_programming
* Directory: 0x06-pointers\_arrays\_strings
* File: 101-print\_number.c

 Done? Help Check your code Ask for a new correction Get a sandbox QA Review

10. A dream doesn't become reality through magic; it takes sweat, determination and hard work

**#advanced**

Score: 0.0% (*Checks completed: 0.0%*)

  
  
Add one line to [this code](https://github.com/holbertonschool/make_magic_happen/blob/master/magic.c), so that the program prints a[2] = 98, followed by a new line.

* You are not allowed to use the variable a in your new line of code
* You are not allowed to modify the variable p
* You can only write one statement
* You are not allowed to use ,
* You are not allowed to code anything else than the line of expected line of code at the expected line
* Your code should be written at line 19, before the ;
* Do not remove anything from the initial code (not even the comments)
* and don’t change anything but the line of code you are adding (don’t change the spaces to tabs!)
* You are allowed to use the standard library

**Repo:**

* GitHub repository: alx-low\_level\_programming
* Directory: 0x06-pointers\_arrays\_strings
* File: 102-magic.c

 Done? Help Check your code Ask for a new correction Get a sandbox QA Review

11. It is the addition of strangeness to beauty that constitutes the romantic character in art

**#advanced**

Score: 0.0% (*Checks completed: 0.0%*)

Write a function that adds two numbers.

* Prototype: char \*infinite\_add(char \*n1, char \*n2, char \*r, int size\_r);
* Where n1 and n2 are the two numbers
* r is the buffer that the function will use to store the result
* size\_r is the buffer size
* The function returns a pointer to the result
* You can assume that you will always get positive numbers, or 0
* You can assume that there will be only digits in the strings n1 and n2
* n1 and n2 will never be empty
* If the result can not be stored in r the function must return 0

julien@ubuntu:~/0x06$ cat 103-main.c

#include "main.h"

#include <stdio.h>

/\*\*

\* main - check the code

\*

\* Return: Always 0.

\*/

int main(void)

{

char \*n = "1234567892434574367823574575678477685785645685876876774586734734563456453743756756784458";

char \*m = "9034790663470697234682914569346259634958693246597324659762347956349265983465962349569346";

char r[100];

char r2[10];

char r3[11];

char \*res;

res = infinite\_add(n, m, r, 100);

if (res == 0)

{

printf("Error\n");

}

else

{

printf("%s + %s = %s\n", n, m, res);

}

n = "1234567890";

m = "1";

res = infinite\_add(n, m, r2, 10);

if (res == 0)

{

printf("Error\n");

}

else

{

printf("%s + %s = %s\n", n, m, res);

}

n = "999999999";

m = "1";

res = infinite\_add(n, m, r2, 10);

if (res == 0)

{

printf("Error\n");

}

else

{

printf("%s + %s = %s\n", n, m, res);

}

res = infinite\_add(n, m, r3, 11);

if (res == 0)

{

printf("Error\n");

}

else

{

printf("%s + %s = %s\n", n, m, res);

}

return (0);

}

julien@ubuntu:~/0x06$ gcc -Wall -pedantic -Werror -Wextra -std=gnu89 103-main.c 103-infinite\_add.c -o 103-add

julien@ubuntu:~/0x06$ ./103-add

1234567892434574367823574575678477685785645685876876774586734734563456453743756756784458 + 9034790663470697234682914569346259634958693246597324659762347956349265983465962349569346 = 10269358555905271602506489145024737320744338932474201434349082690912722437209719106353804

Error

Error

999999999 + 1 = 1000000000

julien@ubuntu:~/0x06$

**Repo:**

* GitHub repository: alx-low\_level\_programming
* Directory: 0x06-pointers\_arrays\_strings
* File: 103-infinite\_add.c

 Done? Help Check your code Ask for a new correction Get a sandbox QA Review

12. Noise is a buffer, more effective than cubicles or booth walls

**#advanced**

Score: 0.0% (*Checks completed: 0.0%*)

Write a function that prints a buffer.

* Prototype: void print\_buffer(char \*b, int size);
* The function must print the content of size bytes of the buffer pointed by b
* The output should print 10 bytes per line
* Each line starts with the position of the first byte of the line in hexadecimal (8 chars), starting with 0
* Each line shows the hexadecimal content (2 chars) of the buffer, 2 bytes at a time, separated by a space
* Each line shows the content of the buffer. If the byte is a printable character, print the letter, if not, print .
* Each line ends with a new line \n
* If size is 0 or less, the output should be a new line only \n
* You are allowed to use the standard library
* The output should look like the following example, and formatted exactly the same way:

julien@ubuntu:~/0x06$ cat 104-main.c

#include "main.h"

#include <stdio.h>

/\*\*

\* main - check the code

\*

\* Return: Always 0.

\*/

int main(void)

{

char buffer[] = "This is a string!\0And this is the rest of the #buffer :)\1\2\3\4\5\6\7#cisfun\n\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\x20\x21\x34\x56#pointersarefun #infernumisfun\n";

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

printf("---------------------------------\n");

print\_buffer(buffer, sizeof(buffer));

return (0);

}

julien@ubuntu:~/0x06$ gcc -Wall -pedantic -Werror -Wextra -std=gnu89 104-main.c 104-print\_buffer.c -o 104-buffer

julien@ubuntu:~/0x06$ ./104-buffer

This is a string!

---------------------------------

00000000: 5468 6973 2069 7320 6120 This is a

0000000a: 7374 7269 6e67 2100 416e string!.An

00000014: 6420 7468 6973 2069 7320 d this is

0000001e: 7468 6520 7265 7374 206f the rest o

00000028: 6620 7468 6520 2362 7566 f the #buf

00000032: 6665 7220 3a29 0102 0304 fer :)....

0000003c: 0506 0723 6369 7366 756e ...#cisfun

00000046: 0a00 0000 0000 0000 0000 ..........

00000050: 0000 0000 0000 0000 0000 ..........

0000005a: 2021 3456 2370 6f69 6e74 !4V#point

00000064: 6572 7361 7265 6675 6e20 ersarefun

0000006e: 2369 6e66 6572 6e75 6d69 #infernumi

00000078: 7366 756e 0a00 sfun..

julien@ubuntu:~/0x06$

**Repo:**

* GitHub repository: alx-low\_level\_programming
* Directory: 0x06-pointers\_arrays\_strings
* File: 104-print\_buffer.c