



# MINI\_PRINTF

A SIMPLE VERSION OF PRINTF



# MINI\_PRINTF



**language:** C  
**compilation:** gcc \*.c



- ✓ The totality of your source files, except all useless files (binary, temp files, obj files,...), must be included in your delivery.
- ✓ All the bonus files (including a potential specific Makefile) should be in a directory named bonus.
- ✓ Error messages have to be written on the error output.

You must create a function named **mini\_printf** to learn how to use `va_args`.  
`mini_printf` is a first step to achieve the project `my_printf`. You will need to implement few flags and would not manage any text formatting and buffering.  
You should not push your **main** function as we are going to compile using **gcc \*.c**.



The whole libC is forbidden, except **write**, **va\_start**, **va\_arg**, **va\_end**, **malloc**, **free**.



.gitignore is a good way to manage it

Your function must be prototyped like this:

```
int mini_printf(const char *format, ...);
```

That function has to print all the characters in the string **format** and print variable when **%** is used before.

You must process all of the following flags :



%d, %i, %s, %c, %%

Upon successful return, the function should return the number of characters printed (excluding the null byte used to end output to strings).

If an output error is encountered, a negative value is returned.



The manual of printf and stdarg is available for your understanding [man 3 printf](#) / [man 3 stdarg](#)



You do not have to implement the C library printf buffer handling.

## Unit tests



Criterion includes mechanisms to test standard output and standard error, [you can learn more about it there...](#)

```
#include <riterion/criterion.h>
#include <riterion/redirect.h>
#include "my.h"
```

```
void redirect_all_std(void)
{
    cr_redirect_stdout();
    cr_redirect_stderr();
}
```

```
Test(mini_printf, simple_string, .init = redirect_all_std)
{
    mini_printf("hello world");
    cr_assert_stdout_eq_str("hello world");
}
```

## Examples

```
char str[6];

my_strcpy(str, "world");
mini_printf("Hello %s\n", str);
```

```
Terminal
~/B-CPE-101> ./a.out
Hello world
~/B-CPE-101>
```

```
int nb = 21;

mini_printf("If you multiple %d by %d, the result is %i.\n", nb, 2, nb * 2);
```

```
Terminal
~/B-CPE-101> ./a.out | cat -e
If you multiple 21 by 2, the result is 42.$
~/B-CPE-101>
```

```
char str[8];

my_strcpy(str, "Epitech");
mini_printf("The word %s has %i characters.\n", str, my_strlen(str));
```

```
Terminal
~/B-CPE-101> ./a.out | cat -e
The word %Epitech% has 7 characters.$
~/B-CPE-101>
```

{EPITECH}

