B1- Unix and C Lab Seminar

B-CPE-100

Day 09

Structures

v1.8

EPITECH.
L'ECOLE DE L'INNOVATION ET DE L'EXPERTISE INFORMATIQUE

Day 09

Structures

repository name:	CPool_Day09_$ACADEMICYEAR
repository rights:	ramassage-tek
language:	C
group size:	1

- Your repository must contain the totality of your source files, but no useless files (binary, temp files, obj files,...).
- Don't push your **main** function into your delivery directory, we will be adding our own. Your files will be compiled adding our **main.c**.
- If one of your files prevents you from compiling with * **.c**, the Autograder will not be able to correct your work and you will receive a 0.

All **.c** files from your delivery folder will be collected and compiled with your **libmy**, which is found in **CPool_Day09_$ACADEMICYEAR/lib/my**. For those of you using **.h** files, they must be located in **CPool_Day09_$ACADEMICYEAR/include**.

Your functions will automatically compiled the following way:

Terminal

```
~/B-CPE-100> cd task01
~/B-CPE-100> cc *.c -c -I../include/
~/B-CPE-100> cc *.o ~autograder/main_task01.o -L../lib/my/ -o task01 -lmy
```

Create your repository at the beginning of the day and submit your work on a regular basis!
The delivery directory is specified within the instructions for each task.
In order to keep your repository clean, pay attention to `gitignore`.

Allowed system function(s): write, malloc, free

Don't forget to write unit tests for all your functions!

Task 01

my_macroABS.h

Write a macro, named **ABS**, that replaces an argument with an absolute value:

```
#define ABS(value)
```

Delivery: CPool_Day09_$ACADEMICYEAR/include/my_macroABS.h

Task 02

my.h

Write your **my.h** header file that contains the prototypes of all the functions exposed by your **libmy.a**.
Delivery: CPool_Day09_$ACADEMICYEAR/include/my.h

To check exposed functions, see the man of *nm*.

Task 03

my_params_to_array

Write a function that stores the program's parameters into an array of structures and returns the address of the array's first cell. All array elements are to be addressed, including **av[0]**.
The function must be prototyped as follows:

```
struct info_param *my_params_to_array(int ac, char **av);
```

The structures contained is the array are to be allocated.
To indicate the end of the array, the **str** field of its last cell must be set to **0**.
The structure is defined as follows:

```
struct info_param
{
        int length;             // parameter's length
        char *str;              // parameter's address
        char *copy;             // parameter's copy
        char **word_array;      // the result of my_str_to_word_array(str)
};
```

Delivery: CPool_Day09_$ACADEMICYEAR/my_params_to_array.c
Do not submit the **struct info_param** structure; the tests set will use its own.

Your function will be tested with your **my_show_word_array**.
As we will not compile **my_show_word_array.c**, you need to make it work using your library.

Task 04

my_show_param_array

Write a function that displays the content of an array created with the previous function, and prototyped as follows:

```
int my_show_param_array(struct info_param const *par);
```

Do not submit the **struct info_param** structure; the tests set will use its own.
For each cell, display one of the following elements per line: parameter, size and words (one per line).
Delivery: CPool_Day09_$ACADEMICYEAR/my_show_param_array.c

Your function will be tested with your **my_str_to_word_array**.
As we will not compile **my_str_to_word_array.c**, you need to make it work using your library.

Task 05

get_color

Write a function that returns the color as an **int** by handling its three **RGB** components. The function must be prototyped as follows:

```
int get_color(unsigned char red, unsigned char green, unsigned char blue);
```

Delivery: CPool_Day09_$ACADEMICYEAR/get_color.c

This task is *only* to be completed with **bit shifts**.

Task 06

swap_endian_color

Write a function that changes the endianness of the color and returns it.
The color should be ordered like this: ARGB
The function must be prototyped as follows:

```
int swap_endian_color(int color);
```

Delivery: CPool_Day09_$ACADEMICYEAR/swap_endian_color.c

This task has to be completed with a **union**.

You will only be working with big and little endians.