B1- C Pool

B-CPE-042

Day 04

Pointers

v1.4

Day 04

Pointers

repository name: : CPool_Day04
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** and our **my_putchar.c** files.
- You are only allowed to use the **my_putchar** function to complete the following tasks, but don't push it into your delivery directory, and don't copy it in *any* of your delivered files.
- 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.

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`.

Task 0

Unit Tests

It is highly recommended to test your functions as you develop them. It is common practice to create a function named `main` (and a designated file to host it) to check the functions separately.
Create a directory named `tests`.
Create a `main` function within a file named `tests-$TASK.c`, to be stored in the `tests` directory named.
This function must contain all the necessary calls to the task function in order to cover all of the function's possible situations (normal or irregular).

Task 1

Automated Exams

Open your weekly schedule in the intranet.
Register for Saturday's exam.
Check that you are registered for Saturday's exam.
--> If you are not registered, go back to the previous step.
Double check that you are registered.
--> If you aren't, go back to previous step.
Are you registered for Saturday's exam?
--> If you are not, go back to first step.

Task 2

my_swap

Write a function that swaps the content of two integers, whose addresses are given as parameter. It must be prototyped as follows:

```
int my_swap(int *a, int *b);
```

Delivery: CPool_Day04/my_swap.c

Task 3

my_putstr

Write a function that displays, one-by-one, the characters of a string.
The address of the string's first character will be found in the pointer passed as a parameter to the function, which must be prototyped as follows:

```
int my_putstr(char *str);
```

Delivery: CPool_Day04/my_putstr.c

Task 4

my_strlen

Write a function that counts and returns the number of characters found in the string passed as parameter. It must be prototyped as follows:

```
int my_strlen(char *str);
```

Delivery: CPool_Day04/my_strlen.c

Task 5

my_evil_str

The goal of this task is to swap each of the string's characters, two by two.
In other words, you will swap the first letter with the last one, the second with the second-to-last and so on.
The function should return a pointer to the first character of the reversed string:

```
char *my_evil_str(char *str);
```

Delivery: CPool_Day04/my_evil_str.c

For instance:

```
a => a
ab => ba
abc => cba
abcd => dcba
abcde => edcba
abcdef => fedcba
```

With the following code, the `str` string is in ReadOnly mode. If you try to modify this string you will get a segfault!

```
char *str;

str = "toto";
```

In order to test your string, you will need to duplicate the string in ReadWrite mode using: **strdup**.

```
#include <string.h>

int             main()
{
        char    *str;

        str = strdup("toto");
        my_evil_str(str);
        my_putstr(str);
        return (0);
}
```

For those who are curious: you will be coding your own strdup in a few days, so be patient!

Task 6

my_getnbr

Write a function that returns a number, sent to the function as a string. It must be prototyped as follows:

```
int my_getnbr(char *str);
```

Delivery: CPool_Day04/my_getnbr.c

Here are some tricky examples to be handled:

```
"+---+--++---+---+---+-42" => -42
"42a43" => 42
"1100000000000000000000000042" => 0 (the number doesn't fit in an integer)
"-1000000000000000000000042" => 0 (for the same reason)
```

Task 7

Bonus: my_sort_int_tab

Write a function that sorts an integer array in ascending order, given a pointer to the first element of the array and its size.

```
void my_sort_int_tab(int *tab, int size);
```

Delivery: CPool_Day04/my_sort_int_tab.c