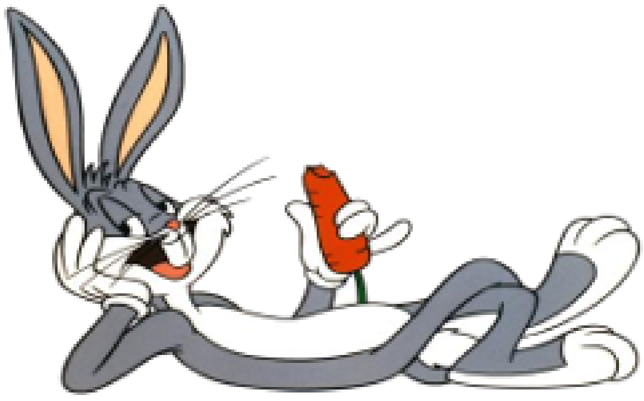

B1- C Pool

B-CPE-042

Day 07

Libmy, arguments

v1.7

Day 07

Libmy, arguments

repository name: : CPool_Day07
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.

All **.c** files from your delivery folder will be collected and compiled with your **libmy**, which is found in **CPool_Day07/lib/my**. For those of you using **.h** files, they must be located in **CPool_Day07/include**.

The Autograder will compile your functions the following way:

```
Terminal
~/B-CPE-042> cd task01
~/B-CPE-042> cc *.c -c -I../include/
~/B-CPE-042> 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`.

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-$FUNCTION_NAME.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).

Always check the empty strings and *int*'s special values (0, MIN, MAX)!

Task 1

libmy.a

Build your own library in **CPool_Day07/lib/my** and name it *libmy.a*
The library **MUST** contain **ALL** of the following functions:

```c
void my_putchar(char c);
int my_isneg(int nb);
int my_put_nbr(int nb);
int my_swap(int *a, int *b);
int my_putstr(char *str);
int my_strlen(char *str);
int my_getnbr(char *str);
void my_sort_int_tab(int *tab, int size);
int my_power_rec(int nb, int power);
int my_square_root(int nb);
int my_is_prime(int nombre);
int my_find_prime_sup(int nb);
char *my_strcpy(char *dest, char *src);
char *my_strncpy(char *dest, char *src, int n);
char *my_revstr(char *str);
char *my_strstr(char *str, char *to_find);
int my_strcmp(char *s1, char *s2);
int my_strncmp(char *s1, char *s2, int n);
char *my_strupcase(char *str);
char *my_strlowcase(char *str);
char *my_strcapitalize(char *str);
int my_str_isalpha(char *str);
int my_str_isnum(char *str);
int my_str_islower(char *str);
int my_str_isupper(char *str);
int my_str_isprintable(char *str);
int my_showstr(char *str);
int my_showmem(char *str, int size);
char *my_strcat(char *dest, char *src);
char *my_strncat(char *dest, char *src, int
    nb);
```

Beware to deliver your **libmy.a** library in the correct folder because it will be used to compile all of your programs.
Delivery: CPool_Day07/lib/my/libmy.a

The functions from the following two tasks must be included in your library.
From tomorrow onwards, none of the functions present in your library must be present in your sources.

Task 2

my_strcat

Write a function that concatenates two strings. It must be prototyped the follwing way:

```
char *my_strcat(char *dest, char *src);
```

Delivery: CPool_Day07/my_strcat.c

man strcat

Task 3

my_strncat

Write a function that concatenates n characters of the **src** string to the end of the **dest** string.
It must be prototyped the following way:

```
char *my_strncat(char *dest, char *src, int nb);
```

Delivery: CPool_Day07/my_strncat.c

Task 4

my_aff_params

Write a program that displays its arguments (received on the command line). Since it is a `PROGRAM`, you need to put the **main** function in your delivered file.
You are to display all arguments (including `argv[0]`), on different lines.
Delivery: CPool_Day07/task04/my_aff_params.c

Your main function must return `0`.

```
~/B-CPE-042> ./a.out test "This is a test " retest | cat -e
./a.out$
test$
This is a test $
retest$
```

Task 5

my_rev_params

Write a program that displays all the arguments received on the command line in reverse order.
You are to display all arguments (including `argv[0]`), on different lines.
Delivery: CPool_Day07/task05/my_rev_params.c

Your main function must return `0`.

```
~/B-CPE-042> ./a.out test "This is a test " retest | cat -e
retest$
This is a test $
test$
./a.out$
```

Task 6

my_sort_params

Write a program that displays all it arguments, in **ascii** order.
You are to display all arguments (including `argv[0]`), on different lines.
Delivery: CPool_Day07/task06/my_sort_params.c

Your main function must return 0.

Terminal

```
~/B-CPE-042> ./a.out test "This is a test " retest | cat -e
./a.out$
This is a test $
retest$
test$
```