



# B1- Unix and C Lab Seminar

---

B-CPE-100

## Day 04

---

Pointers

v2.1



# Day 04

## Pointers

---

repository name: CPool\_Day04\_\$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** 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 01

---

## my\_swap

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

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

**Delivery:** CPool\_Day04\_\$ACADEMICYEAR/my\_swap.c

# Task 02

---

## 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 const *str);
```

**Delivery:** CPool\_Day04\_\$ACADEMICYEAR/my\_putstr.c



# Task 03

## 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 const *str);
```

**Delivery:** CPool\_Day04\_\$ACADEMICYEAR/my\_strlen.c

# Task 04

## 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\_\$ACADEMICYEAR/my\_evil\_str.c

For instance:

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



When testing your function you may encounter "Segmentation fault" errors. Either you're messing with the pointers in your function or the string given in parameter is read-only!



Easy way to have read/write string for testing purpose: *man 3 strdup*

[illegible]

# Task 06

**Delivery:** CPool\_Day04\_\$ACADEMICYEAR/my\_sort\_int\_array.c