Chapter III

Exercise 00 : Makefile

	Exercice : 00
Makefile	
Turn-in directory : $ex00/$	
Files to turn in : `Makefile`	
Allowed functions : `Nothing`	
Remarks : `n/a`	

- Create the `Makefile` that'll compile your `libft.a`.
- The `Makefile` will get its source files from the "srcs" directory.
- The `Makefile` will get its header files from the "includes" directory.
- The lib will be at the root of the exercise.
- The `Makefile` should also implement the following rules: `clean`, `fclean` and `re` as well as `all`.
- `fclean` does the equivalent of a make clean and also erases the binary created during the make. `re` does the equivalent of a make fclean followed by a make.
- We'll only fetch your Makefile and test it with our files. For this exercise, only the following 5 mandatory functions of your lib have to be handled : (`ft_putchar`, `ft_putstr`, `ft_strcmp`, `ft_strlen` and `ft_swap`).

`Watch out for wildcards!`

Chapter IV

Exercise 01 : ft_foreach

	Exercice : 01
ft_foreach	
Turn-in directory : $ex01/$	
Files to turn in : `ft_foreach.c`	
Allowed functions : `Nothing`	
Remarks : `n/a`	

- Create the function `ft_foreach` which, for a given ints array, applies a function on all elements of the array. This function will be applied following the array's order.
- Here's how the function should be prototyped :

```
void        ft_foreach(int *tab, int length, void(*f)(int));
```

- For example, the function `ft_foreach` could be called as follows in order to display all ints of the array :

```
ft_foreach(tab, 1337, &ft_putnbr);
```

Chapter VI

Exercise 03 : ft_any

	Exercice : 03
ft_any	
Turn-in directory : $ex03/$	
Files to turn in : `ft_any.c`	
Allowed functions : `Nothing`	
Remarks : `n/a`	

- Create a function `ft_any` which will return 1 if, passed to the function `f`, at least one element of the array returns 1. Else, it should return 0.
- Here's how the function should be prototyped :

```
int		ft_any(char **tab, int(*f)(char*));
```

- The array will be delimited by 0.

Chapter VII

Exercise 04 : ft_count_if

	Exercice : 04
ft_count_if	
Turn-in directory : *ex*04/	
Files to turn in : `ft_count_if.c`	
Allowed functions : `Nothing`	
Remarks : `n/a`	

- Create a function `ft_count_if` which will return the number of elements of the array that return 1, passed to the function `f`.
- Here's how the function should be prototyped :

```
int                ft_count_if(char **tab, int(*f)(char*));
```

- The array will be delimited by 0.

Chapter VIII

Exercise 05 : ft_is_sort

	Exercice : 05
ft_is_sort	
Turn-in directory : $ex05/$	
Files to turn in : `ft_is_sort.c`	
Allowed functions : `Nothing`	
Remarks : `n/a`	

- Create a function `ft_is_sort` which returns 1 if the array is sorted and 0 if it isn't.
- The function given as argument should return a negative integer if the first argument is lower than the second, 0 if they're equal or a positive integer for anything else.
- Here's how the function should be prototyped :

```
int             ft_is_sort(int *tab, int length, int(*f)(int, int));
```