Chapter III

Exercise 00 : ft_putstr

<table>
<tr><td></td><td>Exercice : 00</td></tr>
<tr><td colspan="2">ft_putstr</td></tr>
<tr><td colspan="2">Turn-in directory : *ex*00/</td></tr>
<tr><td colspan="2">Files to turn in : ft_putstr.c</td></tr>
<tr><td colspan="2">Allowed functions : ft_putchar</td></tr>
<tr><td colspan="2">Remarks : n/a</td></tr>
</table>

42 - Classics : Theses exercises are key assignments that do not earn points, but are mandatory to validate in order to access to the real assignments of the day.

- Create a function that displays a string of characters on the standard output.
- Here's how it should be prototyped :

```
void    ft_putstr(char *str);
```

Chapter IV

Exercise 01 : ft_putnbr

	Exercice : 01
ft_putnbr	
Turn-in directory : *ex*01/	
Files to turn in : `ft_putnbr.c`	
Allowed functions : `ft_putchar`	
Remarks : `n/a`	

42 - Classics : Theses exercises are key assignments that do not earn points, but are mandatory to validate in order to access to the real assignments of the day.

- Create a function that displays the number entered as a parameter. The function has to be able to display all possible values within an `int` type variable.

- Here's how it should be prototyped :

```
void ft_putnbr(int nb);
```

- For example:

  - `ft_putnbr(42)` displays `"42"`.

Chapter V

Exercise 02 : ft_atoi

	Exercice : 02
ft_atoi	
Turn-in directory : *ex*02/	
Files to turn in : `ft_atoi.c`	
Allowed functions : `Nothing`	
Remarks : `n/a`	

42 - Classics : Theses exercises are key assignments that do not earn points, but are mandatory to validate in order to access to the real assignments of the day.

- Reproduce the behavior of the function `atoi` (man atoi).
- Here's how it should be prototyped :

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

Chapter VI

Exercise 03 : ft_strcpy

	Exercice : 03
ft_strcpy	
Turn-in directory : *ex*03/	
Files to turn in : `ft_strcpy.c`	
Allowed functions : `Nothing`	
Remarks : `n/a`	

- Reproduce the behavior of the function `strcpy` (man strcpy).

- Here's how it should be prototyped :

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

Chapter IX

Exercise 06 : ft_strcmp

	Exercice : 06
ft_strcmp	
Turn-in directory : *ex*06/	
Files to turn in : `ft_strcmp.c`	
Allowed functions : `Nothing`	
Remarks : `n/a`	

- Reproduce the behavior of the function `strcmp` (man strcmp).

- Here's how it should be prototyped :

```
int         ft_strcmp(char *s1, char *s2);
```

Chapter X

Exercise 07 : ft_strncmp

	Exercice : 07
ft_strncmp	
Turn-in directory : *ex*07/	
Files to turn in : `ft_strncmp.c`	
Allowed functions : `Nothing`	
Remarks : `n/a`	

- Reproduce the behavior of the function `strncmp` (man strncmp).

- Here's how it should be prototyped :

```
int         ft_strncmp(char *s1, char *s2, unsigned int n);
```

Chapter XI

Exercise 08 : ft_strupcase

	Exercice : 08
ft_strupcase	
Turn-in directory : *ex*08/	
Files to turn in : `ft_strupcase.c`	
Allowed functions : `Nothing`	
Remarks : `n/a`	

- Create a function that transforms every letter of every word to uppercase.

- Here's how it should be prototyped :

```
char        *ft_strupcase(char *str);
```

- It should return `str`.

Chapter XII

Exercise 09 : ft_strlowcase

	Exercice : 09
ft_strlowcase	
Turn-in directory : *ex*09/	
Files to turn in : `ft_strlowcase.c`	
Allowed functions : `Nothing`	
Remarks : `n/a`	

- Create a function that transforms every letter of every word to lowercase.
- Here's how it should be prototyped :

```
char        *ft_strlowcase(char *str);
```

- It should return `str`.

Chapter XV

Exercise 12 : ft_str_is_numeric

	Exercice : 12
ft_str_is_numeric	
Turn-in directory : *ex*12/	
Files to turn in : `ft_str_is_numeric.c`	
Allowed functions : `Nothing`	
Remarks : `n/a`	

- Create a function that returns 1 if the string given as a parameter contains only digits, and 0 if it contains any other character.

- Here's how it should be prototyped :

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

- It should return 1 if `str` is empty.

Chapter XVI

Exercise 13 : ft_str_is_lowercase

	Exercice : 13
ft_str_is_lowercase	
Turn-in directory : *ex*13/	
Files to turn in : `ft_str_is_lowercase.c`	
Allowed functions : `Nothing`	
Remarks : `n/a`	

- Create a function that returns 1 if the string given as a parameter contains only lowercase alphabetical characters, and 0 if it contains any other character.

- Here's how it should be prototyped :

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

- It should return 1 if `str` is empty.

Chapter XVII

Exercise 14 : ft_str_is_uppercase

	Exercice : 14
ft_str_is_uppercase	
Turn-in directory : *ex*14/	
Files to turn in : `ft_str_is_uppercase.c`	
Allowed functions : `Nothing`	
Remarks : `n/a`	

- Create a function that returns 1 if the string given as a parameter contains only uppercase alphabetical characters, and 0 if it contains any other character.

- Here's how it should be prototyped :

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

- It should return 1 if `str` is empty.

Chapter XVIII

Exercise 15 : ft_str_is_printable

Exercice : 15
ft_str_is_printable
Turn-in directory : *ex*15/
Files to turn in : `ft_str_is_printable.c`
Allowed functions : `Nothing`
Remarks : `n/a`

- Create a function that returns 1 if the string given as a parameter contains only printable characters, and 0 if it contains any other character.

- Here's how it should be prototyped :

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

- It should return 1 if `str` is empty.

Chapter XIX

Exercise 16 : ft_strcat

Exercice : 16
ft_strcat
Turn-in directory : *ex*16/
Files to turn in : `ft_strcat.c`
Allowed functions : `Nothing`
Remarks : `n/a`

- Reproduce the behavior of the function `strcat` (man strcat).

- Here's how it should be prototyped :

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

Chapter XX

Exercise 17 : ft_strncat

	Exercice : 17
ft_strncat	
Turn-in directory : *ex*17/	
Files to turn in : `ft_strncat.c`	
Allowed functions : `Nothing`	
Remarks : `n/a`	

- Reproduce the behavior of the function `strncat` (man strncat).

- Here's how it should be prototyped :

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