

Piscine C C 02

Résumé: Ce document est le sujet du module C 02 de la piscine C de 42.

Table des matières

-	Complemen	
II	Préambule	4
III	Exercice 00 : ft_strcpy	5
IV	Exercice 01 : ft_strncpy	6
\mathbf{V}	Exercice 02 : ft_str_is_alpha	7
\mathbf{VI}	Exercice 03 : ft_str_is_numeric	8
VII	Exercice 04 : ft_str_is_lowercase	9
VIII	Exercice $05: ft_str_is_uppercase$	10
IX	Exercice 06 : ft_str_is_printable	11
X	Exercice 07 : ft_strupcase	12
XI	Exercice 08 : ft_strlowcase	13
XII	Exercice 09 : ft_strcapitalize	14
XIII	Exercice 10 : ft_strlcpy	15
XIV	Exercice 11 : ft_putstr_non_printable	16
XV	Exercice 12 : ft_print_memory	17

Chapitre I

Consignes

- Seule cette page servira de référence : ne vous fiez pas aux bruits de couloir.
- Relisez bien le sujet avant de rendre vos exercices. A tout moment le sujet peut changer.
- Attention aux droits de vos fichiers et de vos répertoires.
- Vous devez suivre la procédure de rendu pour tous vos exercices.
- Vos exercices seront corrigés par vos camarades de piscine.
- En plus de vos camarades, vous serez corrigés par un programme appelé la Moulinette.
- La Moulinette est très stricte dans sa notation. Elle est totalement automatisée. Il est impossible de discuter de sa note avec elle. Soyez d'une rigueur irréprochable pour éviter les surprises.
- La Moulinette n'est pas très ouverte d'esprit. Elle ne cherche pas à comprendre le code qui ne respecte pas la Norme. La Moulinette utilise le programme norminette pour vérifier la norme de vos fichiers. Comprendre par là qu'il est stupide de rendre un code qui ne passe pas la norminette.
- Les exercices sont très précisément ordonnés du plus simple au plus complexe. En aucun cas nous ne porterons attention ni ne prendrons en compte un exercice complexe si un exercice plus simple n'est pas parfaitement réussi.
- L'utilisation d'une fonction interdite est un cas de triche. Toute triche est sanctionnée par la note de -42.
- Vous ne devrez rendre une fonction main() que si nous vous demandons un <u>programme</u>.
- La Moulinette compile avec les flags -Wall -Wextra -Werror, et utilise gcc.
- Si votre programme ne compile pas, vous aurez 0.
- Vous <u>ne devez</u> laisser dans votre répertoire <u>aucun</u> autre fichier que ceux explicitement specifiés par les énoncés des exercices.
- Vous avez une question? Demandez à votre voisin de droite. Sinon, essayez avec

votre voisin de gauche.

- Votre manuel de référence s'appelle Google / man / Internet /
- Pensez à discuter sur le forum Piscine de votre Intra, ainsi que sur le slack de votre Piscine!
- Lisez attentivement les exemples. Ils pourraient bien requérir des choses qui ne sont pas autrement précisées dans le sujet...
- Réfléchissez. Par pitié, par Odin! Nom d'une pipe.



Pour cette journée, la norminette doit être lancée avec le flag -R CheckForbiddenSourceHeader. La moulinette l'utilisera aussi.

Chapitre II

Préambule

Voici une discussion extraite de la série Silicon Valley :

- I mean, why not just use Vim over Emacs? (CHUCKLES)
- I do use Vim over Emac.
- Oh, God, help us! Okay, uh you know what? I just don't think this is going to work. I'm so sorry. Uh, I mean like, what, we're going to bring kids into this world with that over their heads? That's not really fair to them, don't you think?
- Kids? We haven't even slept together.
- And guess what, it's never going to happen now, because there is no way I'm going to be with someone who uses spaces over tabs.
- Richard! (PRESS SPACE BAR MANY TIMES)
- Wow. Okay. Goodbye.
- One tab saves you eight spaces! (DOOR SLAMS) (BANGING)

. .

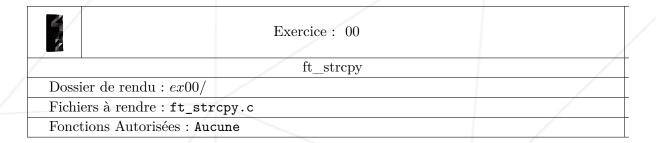
(RICHARD MOANS)

- Oh, my God! Richard, what happened?
- I just tried to go down the stairs eight steps at a time. I'm okay, though.
- See you around, Richard.
- Just making a point.

Heureusement, vous n'êtes pas obliger d'utiliser emacs et votre barre espace pour compléter les exercices suivants.

Chapitre III

Exercice 00: ft_strcpy

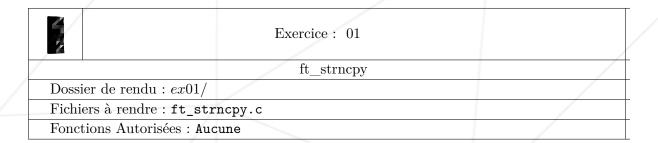


- Reproduire à l'identique le fonctionnement de la fonction strcpy (man strcpy).
- Elle devra être prototypée de la façon suivante :

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

Chapitre IV

Exercice 01: ft_strncpy



- Reproduire à l'identique le fonctionnement de la fonction strncpy (man strncpy).
- Elle devra être prototypée de la façon suivante :

char *ft_strncpy(char *dest, char *src, unsigned int n);

Chapitre V

Exercice 02: ft_str_is_alpha

	Exercice: 02	
/	ft_str_is_alpha	
Dossier de rendu : $ex02/$		
Fichiers à rendre : ft_sti	_is_alpha.c	/
Fonctions Autorisées : Au	cune	

- Écrire une fonction qui renvoie 1 si la chaîne passée en paramètre ne contient que des caractères alphabétiques et renvoie 0 si la fonction contient d'autres types de caractères.
- Elle devra être prototypée de la façon suivante :

int ft_str_is_alpha(char *str);

Chapitre VI

Exercice 03: ft_str_is_numeric

	Exercice: 03	
/	$ft_str_is_numeric$	
Dossier de rendu : $ex03/$		
Fichiers à rendre : ft_str_:	is_numeric.c	
Fonctions Autorisées : Aucu	ne	/

- Écrire une fonction qui renvoie 1 si la chaîne passée en paramètre ne contient que des chiffres et renvoie 0 si la fonction contient d'autres types de caractères.
- Elle devra être prototypée de la façon suivante :

int ft_str_is_numeric(char *str);

Chapitre VII

Exercice 04: ft_str_is_lowercase

Exercice	: 04
ft_str_is_	_lowercase
Dossier de rendu : $ex04/$	
Fichiers à rendre : ft_str_is_lowercase.c	
Fonctions Autorisées : Aucune	

- Écrire une fonction qui renvoie 1 si la chaîne passée en paramètre ne contient que des caractères alphabétiques minuscules et renvoie 0 si la fonction contient d'autres types de caractères.
- Elle devra être prototypée de la façon suivante :

int ft_str_is_lowercase(char *str);

Chapitre VIII

${\bf Exercice} \ \ {\bf 05:ft_str_is_uppercase}$

	Exercice: 05	
	ft_str_is_uppercase	/
Dossier de rendu : $ex0$	5/	
Fichiers à rendre : ft_	str_is_uppercase.c	
Fonctions Autorisées :	Aucune	

- Écrire une fonction qui renvoie 1 si la chaîne passée en paramètre ne contient que des caractères alphabétiques majuscules et renvoie 0 si la fonction contient d'autres types de caractères.
- Elle devra être prototypée de la façon suivante :

int ft_str_is_uppercase(char *str);

Chapitre IX

Exercice 06: ft_str_is_printable

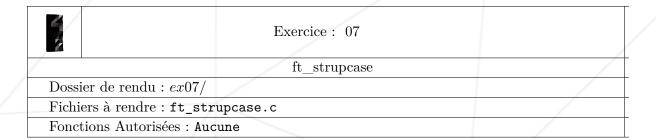
	Exercice: 06	
/	ft_str_is_printable	
Dossier de rendu : $ex06/$		
Fichiers à rendre : ft_st	_is_printable.c	
Fonctions Autorisées : Au	cune	/

- Écrire une fonction qui renvoie 1 si la chaîne passée en paramètre ne contient que des caractères affichables et renvoie 0 si la fonction contient d'autres types de caractères.
- Elle devra être prototypée de la façon suivante :

int ft_str_is_printable(char *str);

Chapitre X

Exercice 07: ft_strupcase



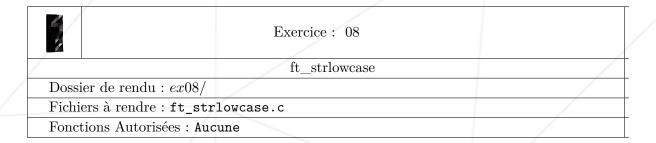
- Écrire une fonction qui met en majuscule chaque lettre.
- $\bullet\,$ Elle devra être prototypée de la façon suivante :

char *ft_strupcase(char *str);

• Elle devra renvoyer str.

Chapitre XI

Exercice 08: ft_strlowcase



- Écrire une fonction qui met en minuscule chaque lettre.
- Elle devra être prototypée de la façon suivante :

char *ft_strlowcase(char *str);

• Elle devra renvoyer str.

Chapitre XII

Exercice 09: ft_strcapitalize

	Exercice: 09	
/	$ft_strcapitalize$	
Dossier de rendu : $ex09/$		/
Fichiers à rendre : ft_st	rcapitalize.c	
Fonctions Autorisées : Au	icune	

- Écrire une fonction qui met en majuscule la première lettre de chaque mot et le reste du mot en minuscule.
- Un mot est une suite de caractères alphanumériques.
- $\bullet\,$ Elle devra être prototypée de la façon suivante :

char *ft_strcapitalize(char *str);

- Elle devra renvoyer str.
- Par exemple :

salut, comment tu vas ? 42mots quarante-deux; cinquante+et+un

• Doit donner:

Salut, Comment Tu Vas ? 42mots Quarante-Deux; Cinquante+Et+Un

Chapitre XIII

Exercice 10: ft_strlcpy

	Exercice: 10	
/	ft_strlcpy	/
Dossier de rendu : $ex10$	0/	
Fichiers à rendre : ft_	strlcpy.c	/
Fonctions Autorisées :	Aucune	

- Reproduire à l'identique le fonctionnement de la fonction strlcpy (man strlcpy).
- $\bullet\,$ Elle devra être prototypée de la façon suivante :

unsigned int ft_strlcpy(char *dest, char *src, unsigned int size);

Chapitre XIV

Exercice 11: ft_putstr_non_printable

4	Exercice: 11	
	ft_putstr_with_non_printable	
Dossi	er de rendu : $ex11/$	/
Fichi	ers à rendre : ft_putstr_non_printable.c	/
Fonct	cions Autorisées : write	/

- Écrire une fonction qui affiche une chaîne de caractères à l'écran. Si cette chaîne contient des caractères non-imprimables, ils devront être affichés sous forme hexadécimale (en minuscules) en les précédant d'un "backslash".
- Par exemple, avec ce paramètre :

Coucou\ntu vas bien ?

• La fonction devra afficher :

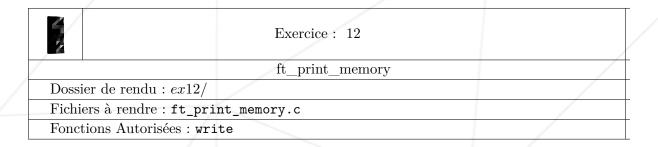
Coucou\Oatu vas bien ?

• Elle devra être prototypée de la façon suivante :

void ft_putstr_non_printable(char *str);

Chapitre XV

Exercice 12: ft_print_memory



- Écrire une fonction qui affiche une zone mémoire à l'écran.
- L'affichage de la zone mémoire est séparée en trois "colonnes" séparées par un espace :
 - o L'adresse en hexadécimal du premier caractère de la ligne suivi d'un ' :'.
 - Le contenu en hexadécimal avec un espace tous les deux caractères et doit etre complété avec des espaces si nécessaire (voir l'exemple en dessous).
 - Le contenu en caractères imprimables.
- Si un caractère est non-imprimable il sera remplacé par un point.
- Chaque ligne doit gérer seize caractères.
- Si size est égale à 0, rien ne sera affiché.

Piscine C C 02

• Exemple:

```
$> ./ft_print_memory
00000010a161f40: 426f 6e6a 6f75 7220 6c65 7320 616d 696e Bonjour les amin
00000010a161f50: 6368 6573 090a 0963 2020 6573 7420 666f ches...c est fo
00000010a161f60: 7509 746f 7574 0963 6520 7175 206f 6e20 u.tout.ce qu on
00000010a161f70: 7065 7574 2066 6169 7265 2061 7665 6309 peut faire avec.
00000010a161f80: 0a09 7072 696e 745f 6d65 6d6f 7279 0a0a ..print_memory..
00000010a161f90: 0a09 6c6f 6c2e 6c6f 6c0a 2000 ..lol.lol. .

$> ./ft_print_memory | cat -te
000000107ff9f40: 426f 6e6a 6f75 7220 6c65 7320 616d 696e Bonjour les amin$
000000107ff9f50: 6368 6573 090a 0963 2020 6573 7420 666f ches...c est fo$
000000107ff9f60: 7509 746f 7574 0963 6520 7175 206f 6e20 u.tout.ce qu on $
000000107ff9f70: 7065 7574 2066 6169 7265 2061 7665 6309 peut faire avec.$
000000107ff9f80: 0a09 7072 696e 745f 6d65 6d6f 7279 0a0a ..print_memory..$
000000107ff9f90: 0a09 6c6f 6c2e 6c6f 6c0a 2000 ..lol.lol. .$
$>
```

• Elle devra être prototypée de la façon suivante :

```
void *ft_print_memory(void *addr, unsigned int size);
```

• Elle devra renvoyer addr.