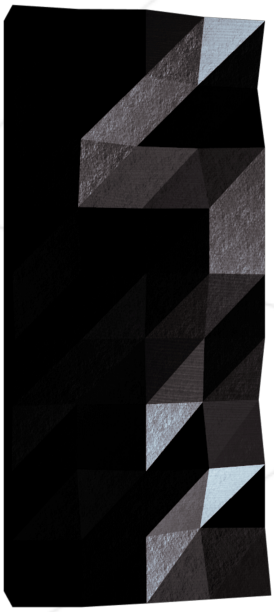

Piscine C

C 11

Staff 42 pedago@42.fr

Résumé: Ce document est le sujet du module C 11 de la piscine C de 42.

Table des matières

I	**Consignes**	**2**
II	**Préambule**	**4**
III	**Exercice 00 : ft_foreach**	**5**
IV	**Exercice 01 : ft_map**	**6**
V	**Exercice 02 : ft_any**	**7**
VI	**Exercice 03 : ft_count_if**	**8**
VII	**Exercice 04 : ft_is_sort**	**9**
VIII	**Exercice 05 : do-op**	**10**
IX	**Exercice 06 : ft_sort_string_tab**	**12**
X	**Exercice 07 : ft_advanced_sort_string_tab**	**13**

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 programme.
- La Moulinette compile avec les flags -Wall -Wextra -Werror, et utilise `gcc`.
- Si votre programme ne compile pas, vous aurez `0`.
- Vous ne devez laisser dans votre répertoire aucun 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.

Chapitre II

Préambule

Citation issue du film `V pour Vendetta` :

`Voilà ! Vois en moi l'image d'un humble Vétéran de VaudeVille, distribué Vicieusement dans les rôles de Victime et de Vilain par les Vicissitudes de la Vie. Ce Visage, plus qu'un Vil Vernis de Vanité, est un Vestige de la Vox populi aujourd'hui Vacante, éVanouie. Cependant, cette Vaillante Visite d'une Vexation passée se retrouVe ViVifiée et a fait Vœu de Vaincre cette Vénale et Virulente Vermine Vantant le Vice et Versant dans la Vicieusement Violente et Vorace Violation de la Volition. Un seul Verdict : la Vengeance. Une Vendetta telle une offrande VotiVe mais pas en Vain car sa Valeur et sa Véracité Viendront un jour faire Valoir le Vigilant et le Vertueux. En Vérité ce Velouté de Verbiage Vire Vraiment au Verbeux, alors laisse-moi simplement ajouter que c'est un Véritable honneur que de te rencontrer.`

`Appelle-moi V.`

`Avoid Aliterations. Always.`

.

Chapitre III

Exercice 00 : ft_foreach

	Exercice : 00
ft_foreach	
Dossier de rendu : $ex00/$	
Fichiers à rendre : `ft_foreach.c`	
Fonctions Autorisées : `Aucune`	

- Écrire une fonction `ft_foreach` qui, pour un tableau d'entiers donné, appliquera une fonction sur tous les éléments de ce tableau. Cette fonction sera appliquée dans l'ordre du tableau.
- La fonction sera prototypée de la manière suivante :

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

- Par exemple, la fonction `ft_foreach` pourra être appelée de la façon suivante pour afficher l'ensemble des entiers du tableau :

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

Chapitre IV

Exercice 01 : ft_map

	Exercice : 01
ft_map	
Dossier de rendu : $ex01/$	
Fichiers à rendre : `ft_map.c`	
Fonctions Autorisées : `malloc`	

- Écrire une fonction `ft_map` qui, pour un tableau d'entiers donné, appliquera une fonction sur tous les éléments de ce tableau (dans l'ordre) et retournera un tableau de toutes les valeurs de retour.
- Cette fonction sera appliquée dans l'ordre du tableau.
- La fonction sera prototypée de la manière suivante :

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

Chapitre V

Exercice 02 : ft_any

	Exercice : 02
ft_any	
Dossier de rendu : $ex02/$	
Fichiers à rendre : `ft_any.c`	
Fonctions Autorisées : `Aucune`	

- Écrire une fonction `ft_any` qui renverra 1 si, en le passant à la fonction `f`, au moins un élément du tableau renvoie autre chose que 0, sinon elle renverra 0.
- Cette fonction sera appliquée dans l'ordre du tableau.
- La fonction sera prototypée de la manière suivante :

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

- Le tableau sera terminé par un pointeur nul.

Chapitre VI

Exercice 03 : ft_count_if

	Exercice : 03
ft_count_if	
Dossier de rendu : *ex*03/	
Fichiers à rendre : `ft_count_if.c`	
Fonctions Autorisées : `Aucune`	

- Écrire une fonction `ft_count_if` qui renverra le nombre d'éléments du tableau qui, en les passant à la fonction `f`, ne renvoient pas 0.
- Cette fonction sera appliquée dans l'ordre du tableau.
- La fonction sera prototypée de la manière suivante :

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

Chapitre VII

Exercice 04 : ft_is_sort

	Exercice : 04
ft_is_sort	
Dossier de rendu : *ex*04/	
Fichiers à rendre : `ft_is_sort.c`	
Fonctions Autorisées : `Aucune`	

- Écrire une fonction `ft_is_sort` qui renverra 1 si le tableau est trié et 0 dans le cas contraire.
- La fonction passée en paramètre renverra un entier négatif si le premier argument est inférieur au deuxième, 0 s'ils sont égaux et un entier positif autrement.
- La fonction sera prototypée de la manière suivante :

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

Chapitre VIII

Exercice 05 : do-op

	Exercice : 05
do-op	
Dossier de rendu : $ex05/$	
Fichiers à rendre : `Tous les fichiers nécessaires a votre programme`	
Fonctions Autorisées : `write`	

- Écrire un programme qui s'appelle `do-op`.
- Le programme devra être lancé avec trois arguments : `do-op valeur1 operateur valeur2`
- Exemple :

```
$>./do-op 42 "+" 21
63
$>
```

- Vous devriez utilisé un tableau de pointeur sur fonction afin d'appeler la fonction correspondant à un `operateur`.
- En cas d'opérateur inconnu votre programme doit afficher `0`.
- Si le nombre d'arguments n'est pas correct, `do-op` n'affiche rien.
- Votre programme doit accepter et afficher le résultat avec les opérateurs suivant : '+' '-' '/' '*' et '%'
- En cas de division par 0 votre programme doit afficher :

```
Stop : division by zero
```

- En cas de modulo par 0 votre programme doit afficher :

```
Stop : modulo by zero
```

- Voici un exemple de tests de la Moulinette :

```
$> make clean
$> make
$> ./do-op
$> ./do-op 1 + 1
2
$> ./do-op 42amis - --+-20toto12
62
$> ./do-op 1 p 1
0
$> ./do-op 1 + toto3
1
$>
$> ./do-op toto3 + 4
4
$> ./do-op foo plus bar
0
$> ./do-op 25 / 0
Stop : division by zero
$> ./do-op 25 % 0
Stop : modulo by zero
$>
```

Chapitre IX

Exercice 06 : ft_sort_string_tab

	Exercice : 06
ft_sort_string_tab	
Dossier de rendu : *ex*06/	
Fichiers à rendre : `ft_sort_string_tab.c`	
Fonctions Autorisées : `Aucune`	

- Écrire la fonction `ft_sort_string_tab` qui trie par ordre `ascii` les `chaîne de caractères`.
- `tab` sera terminé par un pointeur nul
- Le tri s'effectuera en échangeant les pointeurs du tableau.
- Elle devra être prototypée de la façon suivante :

```
void ft_sort_string_tab(char **tab);
```

Chapitre X

Exercice 07 : ft_advanced_sort_string_tab

	Exercice : 07
ft_advanced_sort_string_tab	
Dossier de rendu : *ex*07/	
Fichiers à rendre : `ft_advanced_sort_string_tab.c`	
Fonctions Autorisées : `Aucune`	

- Écrire la fonction `ft_advanced_sort_string_tab` qui trie, en fonction du retour de la fonction passée en paramètre
- Le tri s'effectuera en échangeant les pointeurs du tableau.
- `tab` sera terminé par un pointeur nul
- Elle devra être prototypée de la façon suivante :

```
void ft_advanced_sort_string_tab(char **tab, int(*cmp)(char *, char *));
```

`Un appel à ft_advanced_sort_string_tab() avec en second paramètre ft_strcmp donnera le même resultat que ft_sort_string_tab().`