



Piscina C

C 11

*Sumário: Este documento é o tema do módulo C 11 da Piscina C da 42.*

# Conteúdo

I	Instruções	2
II	Preâmbulo	4
III	Exercício 00 : ft_FOREACH	5
IV	Exercice 01 : ft_map	6
V	Exercício 02 : ft_any	7
VI	Exercício 03 : ft_count_if	8
VII	Exercício 04 : ft_is_sort	9
VIII	Exercício 05 : do-op	10
IX	Exercício 06 : ft_SORT_STRING_TAB	12
X	Exercício 07 : ft_advanced_SORT_STRING_TAB	13

# Capítulo I

## Instruções

- Somente esta página servirá de referência, não confie nos boatos.
- Releia bem o tema antes de entregar seus exercícios. A qualquer momento o tema pode mudar.
- Atenção aos direitos de seus arquivos e suas pastas.
- Você deve seguir o procedimento de entrega para todos os seus exercícios.
- Os seus exercícios serão corrigidos por seus colegas de piscina.
- Além dos seus colegas, haverá a correção de um programa chamado Moulinette.
- A Moulinette é muito rigorosa na sua avaliação. Ela é completamente automatizada. É impossível discutir sua nota com ela. Tenha um rigor exemplar para evitar surpresas.
- A Moulinette não tem a mente muito aberta. Ela não tenta entender o código que não respeita a Norma. A Moulinette utiliza o programa **norminette** para verificar a norma dos seus arquivos. Então é uma tólice entregar um código que não passa pela **norminette**.
- Os exercícios estão rigorosamente ordenados do mais simples ao mais complexo. Em nenhum caso daremos atenção, nem levaremos em conta um exercício complexo se outro mais simples não tiver sido perfeitamente realizado.
- A utilização de uma função proibida é um caso de fraude. Qualquer fraude é punida com nota de -42.
- Você não deve entregar uma função main() se nós pedirmos um programa.
- A Moulinette compila com as sinalizações -Wall -Wextra -Werror, e utiliza **gcc**.
- Se o seu programa não compila, você terá 0.

- Você não deve deixar em sua pasta nenhum outro arquivo além daqueles explicitamente especificados pelos enunciados dos exercícios.
- Você tem alguma dúvida? Pergunte ao seu vizinho da direita. Ou tente também perguntar ao seu vizinho da esquerda.
- Seu manual de referência se chama `Google / man / Internet / ...`
- Considere discutir no fórum Piscina do seu Intra, assim como no slack da sua Piscina!
- Leia atentamente os exemplos. Eles podem muito bem pedir coisas que não estão especificadas no tema...
- Reflita. Por favor, por Odin! Por tudo que é mais sagrado.

# Capítulo II

## Preâmbulo

Citação do filme V de Vingança:

Voilà! À sua Vista, um humilde Veterano do VaudeVille, destinado a ViVer tanto como Vítima e Vilão pelas Vicissitudes o Destino. Este Vislumbre, não um mero Verniz de Vaidade, é um Vestígio da Vox populi, agora Vaga e Vazia. Porém, esta Valorosa Visita de uma Vexação passada se encontra ViVificada e fez um Voto de Vencer esses Vermes Venais e Virulentos, que se Valem do Vício e Valorizam a Viciosamente Violenta e Voraz Violação da Vontade. O único Veredito: a Vingança. Uma Vendeta como uma oferta Votiva, mas não em Vão pois seu Valor e sua Veracidade Virão um dia Vindicar o Vigilante e o Virtuoso. É Verdade que essa Vívida Verborreia torna-se Verdadeiramente Verbosa, deixe-me simplesmente acrescentar que é uma Verdadeira honra conhecê-la.

Pode me chamar de V.



Evite aliterações. Sempre.

# Capítulo III

## Exercício 00 : ft\_foreach

	Exercício : 00
	ft_foreach
	Pasta de entrega : <i>ex00/</i>
	Arquivos para entregar : <b>ft_foreach.c</b>
	Funções autorizadas : Nenhuma

- Escreva uma função **ft\_foreach** que, para uma matriz de inteiros dada, aplique uma função sobre todos os elementos dessa matriz. Essa função será aplicada na ordem da matriz.
- A função deverá ser prototipada da seguinte maneira:

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

- Por exemplo, a função **ft\_foreach** poderá ser chamada da seguinte forma para mostrar o conjunto de inteiros da matriz:

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

# Capítulo IV

## Exercice 01 : ft\_map

	Exercício : 01
	ft_map
Pasta de entrega :	<i>ex01/</i>
Arquivos para entregar :	<b>ft_map.c</b>
Funções autorizadas :	<b>malloc</b>

- Escreva uma função **ft\_map** que, para uma matriz de inteiros dada, aplicará uma função sobre todos os elementos dessa matriz (em sequência) e retornará uma matriz de todos os valores de retorno.
- Essa função será aplicada na ordem da matriz.
- A função deverá ser prototipada da seguinte maneira:

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

# Capítulo V

## Exercício 02 : ft\_any

	Exercício : 02
	ft_any
Pasta de entrega :	<i>ex02/</i>
Arquivos para entregar :	<b>ft_any.c</b>
Funções autorizadas :	Nenhuma

- Escreva uma função `ft_any` que retornará 1 se pelo menos um elemento da matriz retornar algo diferente de 0 ao ser passado para a função `f`. Caso contrário, ela deve retornar 0.
- Essa função será aplicada na ordem da matriz.
- A função deverá ser prototipada da seguinte maneira:

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

- A matriz terminará com um ponteiro nulo.

# Capítulo VI

## Exercício 03 : ft\_count\_if

	Exercício : 03
	ft_count_if
	Pasta de entrega : <i>ex03/</i>
	Arquivos para entregar : <b>ft_count_if.c</b>
	Funções autorizadas : Nenhuma

- Escreva uma função `ft_count_if` que retornará o número de elementos da matriz que, ao serem passados para a função `f`, não retornam 0.
- Essa função será aplicada na ordem da matriz.
- A função deverá ser prototipada da seguinte maneira:

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

# Capítulo VII

## Exercício 04 : ft\_is\_sort

	Exercício : 04
	ft_is_sort
Pasta de entrega :	<i>ex04/</i>
Arquivos para entregar :	<b>ft_is_sort.c</b>
Funções autorizadas :	Nenhuma

- Escreva uma função `ft_is_sort` que retornará 1 se a matriz estiver ordenada e 0 no caso contrário.
- A função passada como parâmetro retornará um inteiro negativo se o primeiro argumento for inferior ao segundo, 0 se forem iguais e um inteiro positivo em caso contrário.
- A função deverá ser prototipada da seguinte maneira:

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

# Capítulo VIII

## Exercício 05 : do-op

	Exercício : 05
	do-op
Pasta de entrega :	<i>ex05/</i>
Arquivos para entregar :	Todos os arquivos necessários para o seu programa
Funções autorizadas :	<code>write</code>

- Escreva um programa chamado **do-op**.
- O programa deverá ser executado com três argumentos: **do-op valeur1 operateur valeur2**
- Exemplo:

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

- Você deve utilizar uma matriz de ponteiros para função a fim de chamar a função correspondente a um **operador**.
- Em caso de operador desconhecido, seu programa deverá mostrar 0.
- Se o número de argumentos não estiver correto, **do-op** não mostra nada.
- Seu programa deve aceitar e mostrar o resultado com os seguintes operadores: '+' '-' '/' '\*' e '%'
- Em caso de divisão por 0, seu programa deve mostrar:

```
Stop : division by zero
```

- Em caso de módulo por 0, seu programa deve mostrar:

```
Stop : modulo by zero
```

- Veja um exemplo de testes da 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
$>
```

# Capítulo IX

## Exercício 06 : ft\_sort\_string\_tab

	Exercício : 06
	ft_sort_string_tab
	Pasta de entrega : <i>ex06/</i>
	Arquivos para entregar : <i>ft_sort_string_tab.c</i>
	Funções autorizadas : Nenhuma

- Escreva a função `ft_sort_string_tab` que classifica por ordem `ascii` as cadeia de caracteres.
- `tab` terminará com um ponteiro nulo
- A classificação será realizada trocando os ponteiros da matriz.
- Ela deverá ser prototipada da seguinte maneira:

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

# Capítulo X

## Exercício 07 : ft\_advanced\_sort\_string\_tab

	Exercício : 07
	ft_advanced_sort_string_tab
Pasta de entrega :	ex07/
Arquivos para entregar :	ft_advanced_sort_string_tab.c
Funções autorizadas :	Nenhuma

- Escreva a função `ft_advanced_sort_string_tab` que classifica em função do retorno da função passada como parâmetro
- A classificação será realizada trocando os ponteiros da matriz.
- `tab` terminará com um ponteiro nulo
- Ela deverá ser prototipada da seguinte maneira:

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



Um chamado à `ft_advanced_sort_string_tab()` com em segundo parâmetro `ft_strcmp` dará o mesmo resultado que `ft_sort_string_tab()`.