



## Piscina recargada

es bueno estar de regreso

### Resumen:

La piscina estaba bien pero ya pasó el tiempo. Esta serie de ejercicios te ayudarán a recordar todos los conceptos básicos que has aprendido durante la piscina. Funciones, bucles, punteros, estructuras, recordemos juntos las bases sintácticas y semánticas del C.

Versión: 1.2

# Contenido

I	Prefacio	2
II	Introducción	3
III	Reglas generales	4
IV	Ejercicio 00: Oh, sí, más...	6
V	Ejercicio 01: Z	7
VI	Ejercicio 02: limpiar	8
VII	Ejercicio 03: find_sh	9
VIII	Ejercicio 04: MAC	10
IX	Ejercicio 05: ¿Puedes crearlo?	11
X	Ejercicio 06: ft_print_alphabet	12
XI	Ejercicio 07: ft_print_numbers	13
XII	Ejercicio 08: ft_is_negative	14
XIII	Ejercicio 09: pies_pies	15
XIV	Ejercicio 10: ft_swap	16
XV	Ejercicio 11: ft_div_mod	17
XVI	Ejercicio 12: ft_iterative_factorial	18
XVII	Ejercicio 13: ft_recursive_factorial	19
XVIII	Ejercicio 14: ft_sqrt	20
XIX	Ejercicio 15: ft_putstr	21
XX	Ejercicio 16: ft_strlen	22
XXI	Ejercicio 17: ft_strcmp	23
XXII	Ejercicio 18: ft_print_params	24
XXIII	Ejercicio 19: ft_sort_params	25

Piscina recargadaes bueno estar de regreso

XXIV Ejercicio 20: ft_strdup	26
XXV Ejercicio 21: ft_range	27
XXVI Ejercicio 22: ft_abs.h	28
XXVII Ejercicio 23: ft_point.h	29
XXVIII Ejercicio 24: Makefile	30
XXIX Ejercicio 25: ft_foreach	31
XXX Ejercicio 26: ft_count_if	32
XXXI Ejercicio 27: display_file	33
XXXII Presentación y evaluación por pares	34

# Capítulo I

## Prefacio

Edward Joseph Snowden (nacido el 21 de junio de 1983) es un profesional informático estadounidense, ex empleado de la Agencia Central de Inteligencia (CIA) y ex contratista del gobierno de los Estados Unidos que copió y filtró información clasificada de la Agencia de Seguridad Nacional (NSA) en 2013 sin autorización. Sus revelaciones revelaron numerosos programas de vigilancia global, muchos de ellos dirigidos por la NSA y la Alianza de Inteligencia Five Eyes con la cooperación de empresas de telecomunicaciones y gobiernos europeos.

En 2013, Snowden fue contratado por un contratista de la NSA, Booz Allen Hamilton, después de haber trabajado anteriormente en Dell y la CIA. El 20 de mayo de 2013, Snowden voló a Hong Kong después de dejar su trabajo en una instalación de la NSA en Hawaii, y a principios de junio reveló miles de documentos clasificados de la NSA a los periodistas Glenn Greenwald, Laura Poitras y Ewen MacAskill. Snowden llamó la atención internacional después de que aparecieran historias basadas en el material en The Guardian y The Washington Post. Otras publicaciones, incluidas Der Spiegel y The New York Times, hicieron más revelaciones.

El 21 de junio de 2013, el Departamento de Justicia de Estados Unidos reveló cargos contra Snowden por dos cargos de violación de la Ley de Espionaje de 1917 y robo de propiedad gubernamental. Dos días después, voló al aeropuerto Sheremetyevo de Moscú, pero las autoridades rusas notaron que su pasaporte estadounidense había sido cancelado y estuvo restringido a la terminal del aeropuerto durante más de un mes. Rusia finalmente le concedió el derecho de asilo por un año, y repetidas prórrogas le han permitido quedarse al menos hasta 2020. Según se informa, vive en un lugar no revelado en Moscú y continúa buscando asilo en otras partes del mundo.

Snowden, objeto de controversia, ha sido llamado de diversas maneras héroe, denunciante, disidente, traidor y patriota. Sus revelaciones han alimentado debates sobre la vigilancia masiva, el secreto gubernamental y el equilibrio entre la seguridad nacional y la privacidad de la información.

Si quieres saber más, te recomendamos ver el documental Citizenfour.

# Capítulo II

## Introducción

Piscine Reloaded es lo mejor de los ejercicios que hiciste durante C Piscine para recordarte todos los conceptos básicos del lenguaje de programación C.

Si ya has realizado algunos de estos ejercicios durante Piscine C, te recomendamos encarecidamente que no caigas en la tentación de recuperar tu antiguo código. El aprendizaje de programación implica práctica y hacer un código existente no tiene ningún interés.

## Capítulo III

### Reglas generales

- Sólo esta página servirá como referencia; No te fíes de los rumores.
- Asegúrese de tener los permisos adecuados en sus archivos y directorios.
- Tienes que seguir los procedimientos de entrega para cada ejercicio.
- Tus ejercicios sólo serán revisados y calificados por un programa llamado Moulinette.
- Moulinette es muy meticulosa y estricta en la evaluación de su trabajo. Está completamente automatizado y no hay forma de negociar con él. Así que si quieres evitar malas sorpresas, sé lo más minucioso posible.
- Los ejercicios en Shell deben ser ejecutables con `/bin/sh`.
- No puede dejar ningún archivo adicional en su directorio aparte de los especificados en el sujeto.
- ¿Tiene alguna pregunta? Pregúntale a tu compañero de la derecha. De lo contrario, prueba con tu compañero de la izquierda.
- Su guía de referencia se llama Google / man / Internet / ....
- Examinar los ejemplos detenidamente. Es muy posible que soliciten detalles que no se mencionan explícitamente en el tema...
- Moulinette no tiene una mentalidad muy abierta. No intentará comprender su código si no respeta la norma. Moulinette utiliza un programa llamado norminette para comprobar si sus archivos respetan la norma. TL;DR: sería una idiotez enviar un trabajo que no pase el control de Norminette.
- Usar una función prohibida se considera hacer trampa. Los tramposos obtienen -42, y esta calificación no es negociable.
- Sólo tendrás que enviar una función `main()` si te pedimos un programa. \_\_\_\_\_
- Moulinette compila con estos indicadores: `-Wall -Wextra -Werror` y usa CC.
- Si `ft_putchar()` es una función autorizada, compilaremos su código con nuestro `ft_putchar.c`.
- Si su programa no se compila, obtendrá 0.


Piscina recargada

es bueno estar de regreso

- ¡Por Odín, por Thor! Usa tu cerebro!!!

## Capítulo IV

### Ejercicio 00: Oh, sí, más...

	Ejercicio 00
Oh sí, más...	
Directorio de entrega: ex00/	
Archivos para entregar: exo.tar	
Funciones permitidas: Ninguna	

- Cree los siguientes archivos y directorios. Haga lo que sea necesario para que cuando use el comando `ls -l` en su directorio, el resultado se vea así:

```
%> ls-l
total XX
drwx--xr-x 2 XX XX XX 1 de junio 20:47 prueba0 -rwx--xr-- 1 XX XX 4 1
de junio 21:46 prueba1
dr-x---r-- 2 XX XX XX 1 de junio 22:45 prueba2 -r----- 2 XX XX 1 1 de
junio 23:44 prueba3
-rw-r----- 1 XX XX 2 1 de junio 23:43 prueba4 -r----- 2 XX XX 1 1 de
junio 23:44 prueba5
lrwxrwxrwx 1 XX XX 5 1 de junio 22:20 prueba6 -> prueba0 %>
```

- Respecto a los horarios, se aceptará si se muestra el año en el caso del la fecha del ejercicio (1 de junio) está desactualizada por seis meses o más.
- Una vez que haya hecho eso, ejecute `tar -cf exo.tar *` para crear el archivo que se enviará.




No te preocupes por lo que tienes en lugar de "XX".



## Capítulo V

### Ejercicio 01: Z

	Ejercicio 01
Sólo los mejores saben cómo mostrar Z	
Directorio de entrega: ex01/	
Archivos para entregar: z	
Funciones permitidas: Ninguna	


- Cree un archivo llamado z que devuelva "Z", seguido de una nueva línea, siempre que se ejecute el comando. Se utiliza gato en él.

```
?>gato z
```

```
¿Z?>
```

## Capítulo VI

### Ejercicio 02: limpiar

	Ejercicio 02
Directorio de entrega: ex02/	
Archivos para entregar: limpios	
Funciones permitidas: Ninguna	


- En un archivo llamado clean, coloque la línea de comando que buscará todos los archivos, tanto en el directorio actual como en sus subdirectorios, con un nombre que termine en ~, o con un nombre que comience y termine en #.
- La línea de comando mostrará y borrará todos los archivos encontrados.
- Sólo se permite un comando: no ';' o '&&' u otras travesuras.



hombre encontrar

## Capítulo VII

### Ejercicio 03: buscar\_sh


	Ejercicio 03
encontrar_sh.sh	
Directorio de entrega: ex03/	
Archivos para entregar: find_sh.sh	
Funciones permitidas: Ninguna	

- Escriba una línea de comando que busque todos los nombres de archivos que terminen con ".sh" (sin comillas) en el directorio actual y todos sus subdirectorios. Debería mostrar solo los nombres de los archivos sin .sh.
- Ejemplo de salida:

```
$>./find_sh.sh | cat -e find_sh$  
archivo1$  
archivo2$  
archivo3$  
$>
```

## Capítulo VIII

### Ejercicio 04: MAC

	Ejercicio 04
	MAC.sh
Directorio de entrega: ex04/	
Archivos a entregar: MAC.sh	
Funciones permitidas: Ninguna	


- Escriba una línea de comando que muestre las direcciones MAC de su máquina. Cada dirección debe ir seguido de un salto de línea.



hombre ifconfig

# Capítulo IX

## Ejercicio 05: ¿Puedes crearlo?

	Ejercicio 05
¿Puedes crearlo?	
Directorio de entrega: ex05/	
Archivos para entregar: "\?\$*MaRViN*\$?"	
Funciones permitidas: Ninguna	

- Cree un archivo que contenga sólo "42" y NADA más.
- Su nombre será:


```
"\?$*MaRViN*$?"
```

- Ejemplo:

```
$>ls -lRa *MaRV* | cat -e -rw--xr-- 1
75355 32015 2 2 de octubre 12:21 "\?$*MaRViN*$?"$ $>
```

# Capítulo X

## Ejercicio 06: ft\_print\_alphabet


	Ejercicio 06
ft_print_alphabet	
Directorio de entrega: ex06/	
Archivos a entregar: ft_print_alphabet.c Funciones	
permitidas: ft_putchar	

- Crear una función que muestre el alfabeto en minúsculas, en una sola línea, orden ascendente, empezando por la letra 'a'.
- Así es como se debe crear el prototipo:

```
vacío ft_print_alphabet(vacio);
```

# CAPÍTULO XI

## Ejercicio 07: ft\_print\_numbers


	Ejercicio 07
ft_print_numbers	
Directorio de entrega: ex07/	
Archivos a entregar: ft_print_numbers.c Funciones	
permitidas: ft_putchar	

- Cree una función que muestre todos los dígitos, en una sola línea, en orden ascendente.
- Así es como se debe crear el prototipo:

```
void ft_print_numbers(nulo);
```

## Capítulo XII

### Ejercicio 08: ft\_is\_negative

	Ejercicio 08
ft_is_negative	
Directorio de entrega: ex08/	
Archivos a entregar: ft_is_negative.c Funciones	
permitidas: ft_putchar	


- Cree una función que muestre 'N' o 'P' dependiendo del signo del número entero ingresado como parámetro. Si n es negativo, muestra 'N'. Si n es positivo o nulo, muestra 'P'.
- Así es como se debe crear el prototipo:

```
vacío ft_is_negative(int n);
```



## Capítulo XIII

### Ejercicio 09: pies\_pies


	Ejercicio 09
pies_pies	
Directorio de entrega: ex09/	
Archivos para entregar: ft_ft.c	
Funciones permitidas: Ninguna	

- Cree una función que tome un puntero a int como parámetro y establezca el valor "42" a ese int.
- Así es como se debe crear el prototipo:

```
vacío ft_ft(int *nbr);
```

# Capítulo XIV

## Ejercicio 10: ft\_swap

	Ejercicio 10
ft_swap	
Directorio de entrega: ex10/	
Archivos a entregar: ft_swap.c	
Funciones permitidas: Ninguna	

- Crear una función que intercambie el valor de dos números enteros cuyas direcciones se ingresan como parámetros.
- Así es como se debe crear el prototipo:

```
vacío    ft_swap(int *a, int *b);
```

## Capítulo XV

### Ejercicio 11: ft\_div\_mod

	Ejercicio 11
ft_div_mod	
Directorio de entrega: ex11/	
Archivos para entregar: ft_div_mod.c	
Funciones permitidas: Ninguna	


- Cree un prototipo de función ft\_div\_mod como este:

```
vacío    ft_div_mod(int a, int b, int *div, int *mod);
```

- Esta función divide los parámetros a por b y almacena el resultado en el int señalado por div. También almacena el resto de la división de a por b en el int señalado por mod.

## Capítulo XVI

### Ejercicio 12: ft\_iterative\_factorial

	Ejercicio 12
ft_iterative_factorial	
Directorio de entrega: ex12/	
Archivos para entregar: ft_iterative_factorial.c	
Funciones permitidas: Ninguna	


- Crear una función iterada que devuelva un número. Este número es el resultado de una operación factorial basada en el número dado como parámetro.
- Si hay un error, la función debería devolver 0.
- Así es como se debe crear el prototipo:

```
int ft_iterative_factorial(int nb);
```

- Su función debe devolver su resultado en menos de dos segundos.

## Capítulo XVII

### Ejercicio 13: ft\_recursive\_factorial

	Ejercicio 13
ft_recursive_factorial	
Directorio de entrega: ex13/	
Archivos para entregar: ft_recursive_factorial.c	
Funciones permitidas: Ninguna	

- Crear una función recursiva que devuelva el factorial del número dado como parámetro.
- Si hay un error, la función debería devolver 0.
- Así es como se debe crear el prototipo:

```
int ft_recursive_factorial(int nb);
```

# Capítulo XVIII

## Ejercicio 14: ft\_sqrt

	Ejercicio 14
ft_sqrt	
Directorio de entrega: ex14/	
Archivos a entregar: ft_sqrt.c	
Funciones permitidas: Ninguna	

- Cree una función que devuelva la raíz cuadrada de un número (si existe), o 0 si la raíz cuadrada es un número irracional.
- Así es como se debe crear el prototipo:

```
int ft_sqrt(int nb);
```

- Su función debe devolver su resultado en menos de dos segundos.

# Capítulo XIX

## Ejercicio 15: ft\_putstr

	Ejercicio 15
ft_putstr	
Directorio de entrega: ex15/	
Archivos a entregar: ft_putstr.c Funciones	
permitidas: ft_putchar	

- Cree una función que muestre una cadena de caracteres en la salida estándar.
- Así es como se debe crear el prototipo:

```
vacío    ft_putstr(char *cadena);
```

# Capítulo XX

## Ejercicio 16: ft\_strlen

	Ejercicio 16
ft_strlen	
Directorio de entrega: ex16/	
Archivos para entregar: ft_strlen.c	
Funciones permitidas: Ninguna	

- Reproducir el comportamiento de la función strlen (man strlen).
- Así es como se debe crear el prototipo:

```
int ft_strlen(char *cadena);
```



# Capítulo XXI

## Ejercicio 17: ft\_strcmp


	Ejercicio 17
ft_strcmp	
Directorio de entrega: ex17/	
Archivos a entregar: ft_strcmp.c	
Funciones permitidas: Ninguna	

- Reproducir el comportamiento de la función strcmp (man strcmp).
- Así es como se debe crear el prototipo:

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

# Capítulo XXII

## Ejercicio 18: ft\_print\_params


	Ejercicio 18
ft_print_params	
Directorio de entrega: ex18/	
Archivos a entregar: ft_print_params.c Funciones	
permitidas: ft_putchar	

- Estamos tratando con un programa aquí, por lo tanto deberías tener una función principal en su archivo .c.
- Crear un programa que muestre los argumentos dados.
- Ejemplo:

```
$> ./a.out prueba1 prueba2 prueba3 prueba1  
prueba2  
prueba3  
$>
```

## Capítulo XXIII

### Ejercicio 19: ft\_sort\_params

	Ejercicio 19
ft_sort_params	
Directorio de entrega: ex19/	
Archivos a entregar: ft_sort_params.c Funciones	
permitidas: ft_putchar	

- Estamos tratando con un programa aquí, por lo tanto deberías tener una función principal en su archivo .c.
- Cree un programa que muestre los argumentos dados ordenados por orden ASCII.
- Debería mostrar todos los argumentos, excepto argv[0].
- Todos los argumentos deben tener su propia línea.

# Capítulo XXIV

## Ejercicio 20: ft\_strdup


	Ejercicio 20
ft_strdup	
Directorio de entrega: ex20/	
Archivos a entregar: ft_strdup.c	
Funciones permitidas: malloc	

- Reproducir el comportamiento de la función strdup (man strdup).
- Así es como se debe crear el prototipo:

```
char* ft_strdup(char *src);
```

# Capítulo XXV

## Ejercicio 21: ft\_range

	Ejercicio 21
rango_pies	
Directorio de entrega: ex21/	
Archivos a entregar: ft_range.c	
Funciones permitidas: malloc	

- Cree una función `ft_range` que devuelva una matriz de entradas. Esta matriz `int` debe contener todos los valores entre mínimo y máximo.
- Mínimo incluido - máximo excluido.
- Así es como se debe crear el prototipo:

```
entero *ft_range(int mínimo, int máximo);
```

- Si el valor mínimo es mayor o igual al valor máximo, se debe devolver un puntero nulo.

## Capítulo XXVI

### Ejercicio 22: ft\_abs.h

	Ejercicio 22
	ft_abs.h
	Directorio de entrega: ex22/
	Archivos para entregar: ft_abs.h
	Funciones permitidas: Ninguna

- Cree una macro ABS que reemplace su argumento por su valor absoluto:

```
#definir ABS(Valor)
```




Se le pide que haga algo que normalmente está prohibido por la única vez que lo autorizamos.

Norma,

# Capítulo XXVII

## Ejercicio 23: ft\_point.h

	Ejercicio 23
ft_point.h	
Directorio de entrega: ex23/	
Archivos a entregar: ft_point.h	
Funciones permitidas: Ninguna	

- Cree un archivo ft\_point.h que compilará el siguiente archivo principal:

```
#include "ft_point.h"

void set_point(t_point *punto) {

    punto->x = 42;
    punto->y = 21;
}

int principal (vacío) {

    punto_t punto;

    set_point(&punto);
    devolver (0);
}
```

## Capítulo XXVIII

### Ejercicio 24: archivo Make

	Ejercicio 24
Archivo Make	
Directorio de entrega: ex24/	
Archivos para entregar: Makefile	
Funciones permitidas: Ninguna	

- Cree el Makefile que compilará su libft.a.
- El Makefile obtendrá sus archivos fuente del directorio "srcs".
- El Makefile obtendrá sus archivos de encabezado del directorio "incluye".
- La biblioteca estará en la raíz del ejercicio.
- El Makefile también debe implementar las siguientes reglas: clean, fclean y re así como todos.
- fclean hace el equivalente a un make clean y también borra el binario creado durante el make. re hace el equivalente a make fclean seguido de make.
- Sólo buscaremos su Makefile y lo probaremos con nuestros archivos. Para este ejercicio, sólo se deben manejar las siguientes 5 funciones obligatorias de su lib: (ft\_putchar, ft\_putstr, ft\_strcmp, ft\_strlen y ft\_swap).



¡Cuidado con los comodines!



# Capítulo XXIX

## Ejercicio 25: ft\_foreach

	Ejercicio 25
ft_foreach	
Directorio de entrega: ex25/	
Archivos para entregar: ft_foreach.c	
Funciones permitidas: Ninguna	

- Cree la función `ft_foreach` que, para una matriz `ints` determinada, aplica una función a todos los elementos de la matriz. Esta función se aplicará siguiendo el orden del array.
- Así es como se debe crear el prototipo de la función:


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

- Por ejemplo, la función `ft_foreach` podría llamarse de la siguiente manera para mostrar todos los enteros de la matriz:

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

# Capítulo XXX

## Ejercicio 26: ft\_count\_if

	Ejercicio 26
ft_count_if	
Directorio de entrega: ex26/	
Archivos para entregar: ft_count_if.c	
Funciones permitidas: Ninguna	


- Cree una función ft\_count\_if que devolverá el número de elementos del matriz que devuelve 1, pasada a la función f.
- Así es como se debe crear el prototipo de la función:

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

- La matriz estará delimitada por 0.

# Capítulo XXXI

## Ejercicio 27: mostrar\_archivo

	Ejercicio 27
mostrar_archivo	
Directorio de entrega: ex27/	
Archivos para entregar: Makefile y archivos necesarios para su programa	
Funciones permitidas: cerrar, abrir, leer, escribir	

- Cree un programa llamado `ft_display_file` que muestre, en la salida estándar, sólo el contenido del archivo dado como argumento.
- El directorio de envío debe tener un Makefile con las siguientes reglas: `all`, `clean`, `fclean`. El binario se llamará `ft_display_file`.
- La función `malloc` está prohibida. Sólo puedes hacer este ejercicio declarando un matriz de tamaño fijo.
- Todos los archivos dados como argumentos serán válidos.
- Los mensajes de error deben mostrarse en su salida reservada seguidos de un nuevo línea.

- Si no se proporciona ningún argumento, debería mostrarse

Falta el nombre del archivo.

- Si hay más de un argumento, debería mostrarse

Demasiados argumentos.

- Si el archivo no se puede leer, debería mostrarse

No se puede leer el archivo.

## Capítulo XXXII

# Presentación y evaluación por pares

Entregue su tarea en su repositorio de Git como de costumbre. Durante la defensa sólo se evaluará el trabajo dentro de su repositorio. No dude en volver a verificar los nombres de sus carpetas y archivos para asegurarse de que sean correctos.