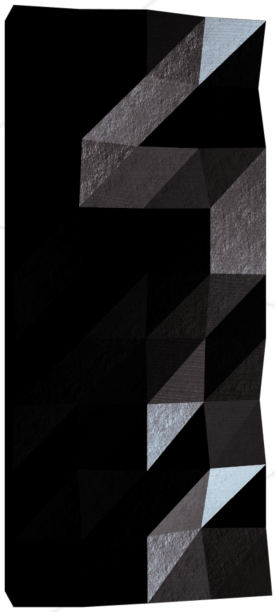

Piscine Reloaded

Es bueno estar de vuelta

Resumen:
La piscina estuvo bien, pero el tiempo pasa. Esta serie de ejercicios te ayudará a recordar los aspectos básicos que has aprendido durante la Piscina. Funciones, bucles, punteros, estructuras: recordemos juntos las bases sintácticas y semánticas de `C`

Versión: 1.2

Índice general

Capítulo I

Preámbulo

Edward Joseph Snowden (nacido el 21 de junio de 1983) es un profesional de la informática estadounidense, exempleado de la Agencia Central de Inteligencia (CIA) y excontratista para el gobierno de los Estados Unidos que copió y filtró información clasificada de la Agencia de Seguridad Nacional (NSA) en 2013 sin autorización. Sus divulgaciones revelaron numerosos programas de vigilancia globales, muchos dirigidos por la NSA y la alianza de inteligencia Cinco Ojos en colaboración con empresas de telecomunicación y gobiernos europeos.

En 2013, Snowden fue contratado por un contratista de la NSA, Booz Allen Hamilton, después de trabajar previamente con Dell y la CIA. El 20 de mayo de 2013, Snowden voló a Hong Kong tras abandonar su trabajo en las instalaciones de la NSA en Hawái y, a principios de junio, reveló miles de documentos clasificados de la NSA a los periodistas Glenn Greenwald, Laura Poitras y Ewen MacAskill.
Snowden atrajo la atención internacional después de que historias basadas en el material apareciesen en los periódicos The Guardian y The Washington Post.
Otras publicaciones, incluyendo Der Spiegel y The New York Times, revelaron divulgaciones adicionales.

El 21 de junio de 2013, el Departamento de Justicia de los EE. UU.
presentó cargos contra Snowden por violación de la Ley de Espionaje de 1917 y robo de propiedad gubernamental. Dos días después, Snowden voló al aeropuerto Sheremétievo de Moscú, pero las autoridades rusas detectaron que su pasaporte estadounidense había sido cancelado y quedó confinado en la terminal del aeropuerto durante un mes. Finalmente, Rusia le concedió el derecho a asilo durante un año, y numerosas prórrogas le han permitido quedarse al menos hasta 2020. Al parecer, vive en una ubicación no revelada en Moscú, y sigue buscando asilo en cualquier parte del mundo.

Un asunto controvertido, Snowden ha sido llamado indistintamente héroe, soplón, disidente, traidor y patriota. Sus divulgaciones han alimentado debates sobre la vigilancia masiva, el secretismo gubernamental y el equilibrio entre la seguridad nacional y la privacidad de la información.

Hay un muy buen documental sobre esto llamado Citizenfour.

Capítulo II

Introducción

La `Piscine Reloaded` es un recopilatorio de los mejores ejercicios que hiciste durante la `C Piscine` para recordarte lo básico del lenguaje de programación `C`.

Puede ser que ya hayas hecho algunos de los ejercicios durante la `C Piscine`, te recomendamos encarecidamente que evites la tentación de recuperar tu antiguo código. Aprender a programar requiere práctica, y reproducir un código ya existente no tiene ningún interés..

Capítulo III

Instrucciones

- Esta página será la única referencia: no te fíes de los rumores.
- Asegúrate de que tus directorios y archivos tienen los permisos adecuados.
- Debes respetar el procedimiento de entrega para todos tus ejercicios.
- Tus ejecicios serán corregidos **únicamente** por la Moulinette.
- La Moulinette es muy estricta a la hora de evaluar. Está completamente automatizada. Es imposible discutir con ella sobre tu nota. Por lo tanto, se extremadamente riguroso para evitar cualquier sorpresa.
- La Moulinette no tiene una mente muy abierta. No intenta comprender el código que no respeta la Norma. La Moulinette utiliza el programa `norminette` para comprobar La Norma en sus archivos. Entiende entonces que es estúpido entregar un código que no pase la `norminette`.
- Los ejercicios en Shell deberán ser ejecutados con `/bin/sh`
- Los ejercicios han sido ordenados con mucha precisión, del más sencillo al más complejo. En ningún caso se tendrá en cuenta un ejercicio complejo si no se ha conseguido realizar perfectamente un ejercicio más sencillo.
- El uso de una función prohibida se considera una trampa. Cualquier trampa será sancionada con la nota `-42`.
- Solamente hay que entregar una función main() si lo que se pide es un programa.
- La Moulinette compila con los flags -Wall -Wextra -Werror y utiliza `cc`.
- Si tu programa no compila, tendrán un `0`.
- No puedes dejar en tu directorio ningún archivo que no se haya indicado de forma explícita en los enunciados de los ejercicios.
- Tu manual de referencia se llama `Google / man / Internet / ...`.
- Si ft_putchar() es una función autorizada, la Moulinette compilará el código usando su propio `ft_putchar.c`
- Razona. ¡Te lo suplico, por Thor, por Odín! Maldita sea.

Capítulo IV

Ejercicio 00 : Oh, sí, másss...

	Ejercicio: 00
Oh, sí, másss...	
Directorio de entrega: *ex*00/	
Archivos a entregar: `exo.tar`	
Funciones autorizadas: `Ninguna`	

- Crea los archivos y directorios siguientes. Haz lo que sea necesario para que cuando utilices el comando `ls -l` en tu directorio, la salida tenga este aspecto:

```
%> ls -l
total XX
drwx--xr-x 2 XX XX  XX Jun 1 20:47 test0
-rwx--xr-- 1 XX XX   4 Jun 1 21:46 test1
dr-x---r-- 2 XX XX  XX Jun 1 22:45 test2
-r-----r-- 2 XX XX   1 Jun 1 23:44 test3
-rw-r----x 1 XX XX   2 Jun 1 23:43 test4
-r-----r-- 2 XX XX   1 Jun 1 23:44 test5
lrwxrwxrwx 1 XX XX   5 Jun 1 22:20 test6 -> test0
%>
```

- Con respecto a las horas, se aceptará que el año que se muestra en la fecha del ejercicio (1 jun) esté desfasado por seis meses o más.
- Una vez hecho esto, ejecuta `tar -cf exo.tar *` para crear el archivo a entregar.

No te preocupes por lo que obtengas en vez de "XX".

Capítulo V

Ejercicio 01 : Z

	Ejercicio: 01
Solo los mejores saben cómo mostrar una Z	
Directorio de entrega: $ex01/$	
Archivos a entregar: `z`	
Funciones autorizadas: `Ninguna`	

- Crea un archivo llamado `z` que devuelva "Z", seguido por un salto de línea, cada vez que se ejecuta el comando `cat` en él.

```
?>cat z
Z
?>
```

Capítulo VI

Ejercicio 02 : clean

	Ejercicio: 02
Directorio de entrega: $ex02/$	
Archivos a entregar: `clean`	
Funciones autorizadas: `Ninguna`	

- En un archivo llamado `clean`, incluye la línea de comando que buscará todos los archivos - en el directorio actual, así como sus subdirectorios - con un nombre acabado en ~, o con un nombre que empiece y acabe con `#`
- La línea de comando mostrará y eliminará todos los archivos encontrados.
- Solo se permite un comando: nada de ';' o '&&' u otros chanchullos.

```
man find
```

Capítulo VII

Ejercicio 03 : find_sh

	Ejercicio: 03
find_sh.sh	
Directorio de entrega: $ex03/$	
Archivos a entregar: `find_sh.sh`	
Funciones autorizadas: `Ninguna`	

- Escribe una línea de comando que busque todos los nombres de archivo acabados en ".sh"(sin las comillas) en el directorio actual y todos sus subdirectorios. Debe mostrar solo los nombres de archivo sin .sh.
- Ejemplo de salida:

```
$>./find_sh.sh | cat -e
find_sh$
file1$
file2$
file3$
$>
```

Capítulo VIII

Ejercicio 04 : MAC

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

- Escribe una línea de comando que muestre las direcciones MAC de tu ordenador. Cada línea debe ir seguida de un salto de línea.

```
man ifconfig
```

Capítulo IX

Ejercicio 05 : ¿Puedes crearlo?

	Ejercicio: 05
¿Puedes crearlo?	
Directorio de entrega: *ex*05/	
Archivos a entregar: `"\?$*'MaRViN'*$?\"`	
Funciones autorizadas: `Ninguna`	

- Crea un archivo que contenga únicamente "42", y NADA más.
- Su nombre será:

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

- Ejemplo:

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

Capítulo X

Ejercicio 06 : ft_print_alphabet

	Ejercicio: 06
ft_print_alphabet	
Directorio de entrega: *ex*06/	
Archivos a entregar: `ft_print_alphabet.c`	
Funciones autorizadas: `ft_putchar`	

- Crea una función que muestre el alfabeto en minúsculas, en una única línea, en orden ascendente, empezando por la letra 'a'.
- Este debe ser su prototipo:

```
void ft_print_alphabet(void);
```

Capítulo XI

Ejercicio 07 : ft_print_numbers

	Ejercicio: 07
ft_print_numbers	
Directorio de entrega: *ex*07/	
Archivos a entregar: `ft_print_numbers.c`	
Funciones autorizadas: `ft_putchar`	

- Crea una función que muestre todos los dígitos, en una única línea, en orden ascendente.
- Este debe ser su prototipo:

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

Capítulo XII

Ejercicio 08 : ft_is_negative

	Ejercicio: 08
ft_is_negative	
Directorio de entrega: *ex*08/	
Archivos a entregar: `ft_is_negative.c`	
Funciones autorizadas: `ft_putchar`	

- Crea una función que muestre 'N' o 'P' dependiendo del signo del entero introducido como parámetro. Si `n` es negativo, muestra 'N'. Si `n` es positivo o cero, muestra 'P'.
- Este debe ser su prototipo:

```
void ft_is_negative(int n);
```

Capítulo XIII

Ejercicio 09 : ft_ft

	Ejercicio: 09
ft_ft	
Directorio de entrega: *ex*09/	
Archivos a entregar: `ft_ft.c`	
Funciones autorizadas: `Ninguna`	

- Crea una función que tome un puntero a un entero como un parámetro y fije el valor "42.ªl entero.
- Este debe ser su prototipo:

```
void  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 autorizadas: `Ninguna`	

- Crea una función que intercambie el valor de dos enteros cuyas direcciones se introducen como parámetros.
- Este debe ser su prototipo:

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

Capítulo XV

Ejercicio 11 : ft_div_mod

	Ejercicio: 11
ft_div_mod	
Directorio de entrega: $ex11/$	
Archivos a entregar: `ft_div_mod.c`	
Funciones autorizadas: `Ninguna`	

- Crea una función `ft_div_mod` con el siguiente prototipo:

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

- Esta función divide el parámetro `a` entre el parámetro `b` y almacena el resultado en el entero al que apunta `div`. Además, almacena el resto de la división de `a` entre `b` en el entero al que apunta `mod`.

Capítulo XVI

Ejercicio 12 : ft_iterative_factorial

	Ejercicio: 12
ft_iterative_factorial	
Directorio de entrega: $ex12/$	
Archivos a entregar: `ft_iterative_factorial.c`	
Funciones autorizadas: `Ninguna`	

- Crea una función iterativa que devuelva un número. Dicho número es el resultado de una operación factorial basada en el número introducido como parámetro.
- Si hay un error, la función debe devolver 0.
- Este debe ser su prototipo:

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

- Tu 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 a entregar: `ft_recursive_factorial.c`	
Funciones autorizadas: `Ninguna`	

- Crea una función recursiva que devuelva el factorial del número introducido como parámetro.
- Si hay un error, la función debe devolver 0.
- Este debe ser su 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 autorizadas: `Ninguna`	

- Crea una función que devuelva la raíz cuadrada de un número (si existe), o 0 si la raíz cuadrada es un número irracional.
- Este debe ser su prototipo:

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

- Tu 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 autorizadas: `ft_putchar`	

- Crea una función que muestre una cadena de caracteres por la salida estándar.
- Este debe ser su prototipo:

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

Capítulo XX

Ejercicio 16 : ft_strlen

	Ejercicio: 16
ft_strlen	
Directorio de entrega: $ex16/$	
Archivos a entregar: `ft_strlen.c`	
Funciones autorizadas: `Ninguna`	

- Reproduce el comportamiento de la función `strlen` (man strlen).
- Este debe ser su prototipo:

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

Capítulo XXI

Ejercicio 17 : ft_strcmp

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

- Reproduce el comportamiento de la función `strcmp` (man strcmp).
- Este debe ser su 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 autorizadas: `ft_putchar`	

- Esto se trata de un programa, así que deberías tener una función `main` en tu archivo `.c`.
- Crea un programa que muestre sus argumentos dados.
- Ejemplo:

```
$>./a.out test1 test2 test3
test1
test2
test3
$>
```

Capítulo XXIII

Ejercicio 19 : ft_sort_params

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

- Esto se trata de un programa, así que deberías tener una función `main` en tu archivo `.c`.
- Crea un programa que muestre sus argumentos dados ordenados en orden ascii.
- Debe mostrar todos los argumentos, excepto `argv[0]`.
- Todos los argumentos deberán estar separados con un salto de línea.

Capítulo XXIV

Ejercicio 20 : ft_strdup

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

- Reproduce el comportamiento de la función `strdup` (man strdup).
- Este debe ser su prototipo:

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

Capítulo XXV

Ejercicio 21 : ft_range

	Ejercicio: 21
ft_range	
Directorio de entrega: $ex21/$	
Archivos a entregar: `ft_range.c`	
Funciones autorizadas: `malloc`	

- Crea una función `ft_range` que devuelva una matriz de `enteros`. Este array de `enteros` debe contener todos los valores entre `min` y `max`.
- `Min` incluido - `max` excluido.
- Este debe ser su prototipo:

```
int    *ft_range(int min, int max);
```

- Si el valor `min` es mayor o igual al valor `max`, debe devolverse un puntero nulo.

Capítulo XXVI

Ejercicio 22 : ft_abs.h

	Ejercicio: 22
ft_abs.h	
Directorio de entrega: $ex22/$	
Archivos a entregar: `ft_abs.h`	
Funciones autorizadas: `Ninguna`	

- Crea una `macro ABS` que sustituya su argumento por su valor absoluto:

```
#define ABS(Value)
```

Se te está pidiendo que hagas algo normalmente prohibido por la Norma; esta será la única vez que lo autoricemos.

Capítulo XXVII

Ejercicio 23 : ft_point.h

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

- Crea un archivo `ft_point.h` que deberá compilar el siguiente main:

```c
#include "ft_point.h"

void  set_point(t_point *point)
{
  point->x = 42;
  point->y = 21;
}

int   main(void)
{
  t_point   point;

  set_point(&point);
  return (0);
}
```

Capítulo XXVIII

Ejercicio 24 : Makefile

	Ejercicio: 24
Makefile	
Directorio de entrega: $ex24/$	
Archivos a entregar: `Makefile`	
Funciones autorizadas: `Ninguna`	

- Crea el `Makefile` que compile tu `libft.a`.
- El `Makefile` obtendrá sus archivos fuente del directorio “srcs”.
- El `Makefile` obtendrá sus archivos de encabezado del directorio “includes”.
- La lib estará en la raíz del ejercicio.
- El `Makefile` también debe implementar las reglas siguientes: `clean`, `fclean` y `re` además de `all`.
- `fclean` hace lo equivalente a un make clean y también borra el binario creado durante el make. re hace lo equivalente a un make fclean seguido de un make.
- Solo recogeremos tu Makefile y lo probaremos con nuestros archivos. Para este ejercicio, solo se manejarán las siguientes 5 funciones obligatorias de tu lib: (`ft_putchar`, `ft_putstr`, `ft_strcmp`, `ft_strlen` y `ft_swap`).

`¡Cuidado con los wildcards!`

Capítulo XXIX

Ejercicio 25 : ft_foreach

	Ejercicio: 25
ft_foreach	
Directorio de entrega: $ex25/$	
Archivos a entregar: `ft_foreach.c`	
Funciones autorizadas: `Ninguna`	

- Crea la función `ft_foreach` que, para una matriz de enteros dada, aplica una función a todos los elementos de la matriz. Esta función se aplicará siguiendo el orden de la matriz.
- Este debe ser el prototipo de la función:

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

- Por ejemplo, la función `ft_foreach` puede llamarse como sigue para mostrar todos los enteros de la matriz:

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

Capítulo XXX

Ejercicio 26 : ft_count_if

	Ejercicio: 26
ft_count_if	
Directorio de entrega: $ex26/$	
Archivos a entregar: `ft_count_if.c`	
Funciones autorizadas: `Ninguna`	

- Crea una función `ft_count_if` que devuelva el número de elementos de la matriz que devuelven 1 al pasarlos por la función `f`.
- Este debe ser el prototipo de la función:

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

- La matriz estará delimitada por 0.

Capítulo XXXI

Ejercicio 27 : display_file

	Ejercicio: 27
display_file	
Directorio de entrega: $ex27/$	
Archivos a entregar: `Makefile, and files needed for your program`	
Funciones autorizadas: `close, open, read, write`	

- Crea un programa llamado `ft_display_file` que muestre, en la salida estándar, solo el contenido del archivo dado como argumento.
- El directorio de entrega debe tener un `Makefile` con las reglas siguientes: `all`, `clean`, `fclean`. El binario se llamará `ft_display_file`.
- La función `malloc` está prohibida. Solo puedes hacer este ejercicio declarando una matriz de tamaño fijo.
- Todos los archivos dados como argumento serán válidos.
- Los mensajes de error deben mostrarse en su salida reservada seguidos por un salto de línea.
- Si no se da ningún argumento, el programa deberá mostrar

```
File name missing.
```

- Si hay más de un argumento, el programa deberá mostrar

```
Too many arguments.
```

- Si el archivo no puede leerse, el programa deberá mostrar

```
Cannot read file.
```

Capítulo XXXII

Presentación y evaluación por pares

Entrega tu tarea en tu repositorio `Git` como de costumbre. Solo el trabajo dentro de tu repositorio se evaluará durante la defensa. No dudes en revisar varias veces los nombres de tus carpetas y archivos para asegurarte de que son correctos.