

C Piscine C 12

Resumen: Este documento corresponde al enunciado del módulo C 12 de la C Piscine de 42.

Versión: 7.2

# Índice general

I.	Preámbulo	<b>2</b>
II.	Instrucciones	4
III.	Ejercicio 00 : ft_create_elem	6
IV.	Ejercicio 01 : ft_list_push_front	7
v.	Ejercicio 02 : ft_list_size	8
VI.	Ejercicio 03 : ft_list_last	9
VII.	Ejercicio 04 : ft_list_push_back	10
VIII.	Ejercicio 05 : ft_list_push_strs	11
IX.	Ejercicio 06 : ft_list_clear	12
X.	Ejercicio 07 : ft_list_at	13
XI.	Ejercicio 08 : ft_list_reverse	14
XII.	Ejercicio 09 : ft_list_foreach	15
XIII.	Ejercicio 10 : ft_list_foreach_if	16
XIV.	Ejercicio 11 : ft_list_find	17
XV.	Ejercicio 12 : ft_list_remove_if	18
XVI.	Ejercicio 13 : ft_list_merge	19
XVII.	Ejercicio 14 : ft_list_sort	20
XVIII.	Ejercicio 15 : ft_list_reverse_fun	21
XIX.	Ejercicio 16 : ft_sorted_list_insert	22
XX.	Ejercicio $17: ft\_sorted\_list\_merge$	23
XXI	Entrega v evaluación	24

Capítulo I Preámbulo

> AVISO DE SPOILER NO LEA LA PÁGINA SIGUIENTE

#### Tú lo has querido.

- En Star Wars, Darth Vader es el padre de Luke Skywalker.
- En Sospechosos habituales, Verbal es Keyser Soze.
- En El club de la lucha, Tyler Durden y el narrador son la misma persona.
- En El sexto sentido, Bruce Willis está muerto desde el principio.
- En Los otros, los residentes de la casa son los fantasmas y viceversa.
- En Bambi, muere la madre de Bambi.
- En El bosque, los monstruos son los aldeanos y, en realidad, la historia ocurre en nuestra época.
- En Harry Potter, muere Dumbledore.
- En El planeta de los simios, la historia se desarrolla en la Tierra.
- En Juego de tronos, Robb Stark y Joffrey Baratheon mueren en su noche de bodas.
- En Crepúsculo, los vampiros brillan cuando se exponen al sol.
- En Stargate SG-1, Temporada 1, Episodio 18, O'Neill y Carter están en la Antártida.
- En Fullmetal Alchemist: Brotherhood, Alfonse da su vida para que su hermano pueda derrotar a su padre, el auténtico villano.
- En El caballero oscuro: la leyenda renace, Miranda Tate es Talia al Ghul.
- En Super Mario Bros, la princesa se encuentra en otro castillo.
- En Ataque a los Titanes, después de que Eren provoque el rumbling y acabe con el 80 % de la población mundial, Mikasa lo mata para detenerlo

#### Capítulo II

#### Instrucciones

- Esta página será la única referencia: no te fíes de los rumores.
- ¡Ten cuidado! Los enunciados pueden cambiar en cualquier momento.
- Asegúrate de que tus directorios y archivos tienen los permisos adecuados.
- Debes respetar el procedimiento de entrega para todos tus ejercicios.
- Tus compañeros de piscina se encargarán de corregir tus ejercicios.
- Además de por tus compañeros, también serán corregidos por un programa que se llama 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, sé 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 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 <u>ningún</u> archivo que no se haya indicado de forma explícita en los enunciados de los ejercicios.
- ¿Tienes alguna pregunta? Pregunta a tu compañero de la derecha. Si no, prueba con tu compañero de la izquierda.

C Piscine

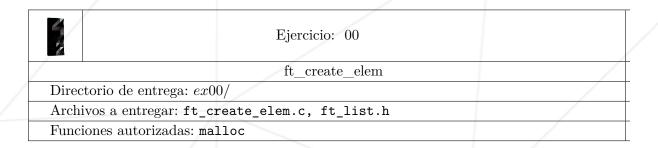
C 12

- Tu manual de referencia se llama Google / man / Internet / ...
- ¡No olvides participar en el slack de tu Piscina!
- Lee detenidamente los ejemplos. Podrían exigir cosas que no se especifican necesariamente en los enunciados. . .
- Razona. ¡Te lo suplico, por Thor, por Odín! Maldita sea.
- Para los ejercicios de hoy, utilizaremos la estructura siguiente:

- Debes colocar esta estructura en un archivo ft\_list.h y entregarlo en cada ejercicio.
- A partir del ejercicio 01 utilizaremos nuestro ft\_create\_elem, así que tenlo en cuenta (podría ser interesante tener tu prototipo en ft\_list.h...).

# Capítulo III

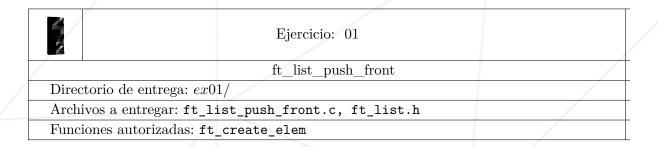
# Ejercicio 00: ft\_create\_elem



- Crea la función ft\_create\_elem que cree un elemento nuevo de tipo t\_list.
- Tendrá que asignar data al parámetro proporcionado y next a NULL.
- El prototipo de la función deberá ser el siguiente:

## Capítulo IV

# Ejercicio 01 : ft\_\_list\_\_push\_\_front

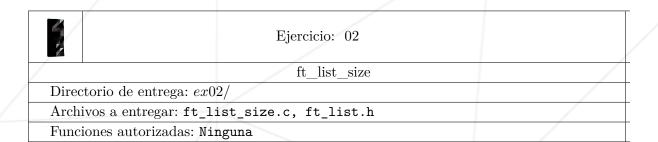


- Crea la función ft\_list\_push\_front que añada al principio de la lista un elemento nuevo de tipo t\_list.
- Tendrá que asignar data al parámetro proporcionado.
- Actualizará, si es preciso, el puntero al principio de la lista.
- El prototipo de la función deberá ser el siguiente:

void ft\_list\_push\_front(t\_list \*\*begin\_list, void \*data);

# Capítulo V

Ejercicio 02: ft\_list\_size



- Crea la función ft\_list\_size que devuelva el número de elementos de la lista.
- El prototipo de la función deberá ser el siguiente:

int ft\_list\_size(t\_list \*begin\_list);

# Capítulo VI

Ejercicio 03: ft\_list\_last

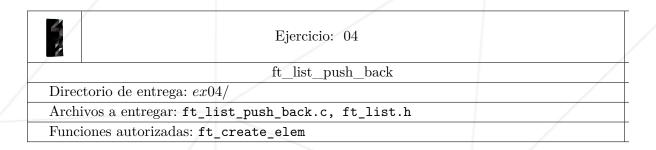


- Crea la función ft\_list\_last que devuelva el último elemento de la lista.
- El prototipo de la función deberá ser el siguiente:

t\_list \*ft\_list\_last(t\_list \*begin\_list);

## Capítulo VII

# Ejercicio 04 : ft\_list\_push\_back

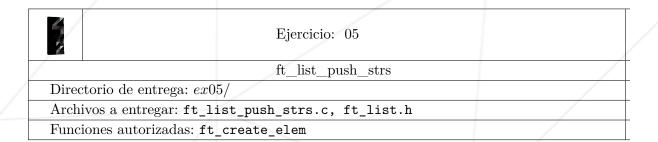


- Crea la función ft\_list\_push\_back que añada al final de la lista un elemento nuevo de tipo t\_list.
- Tendrá que asignar data al parámetro proporcionado.
- Actualizará, si es preciso, el puntero al principio de la lista.
- El prototipo de la función deberá ser el siguiente:

void ft\_list\_push\_back(t\_list \*\*begin\_list, void \*data);

### Capítulo VIII

Ejercicio 05 : ft\_list\_push\_strs

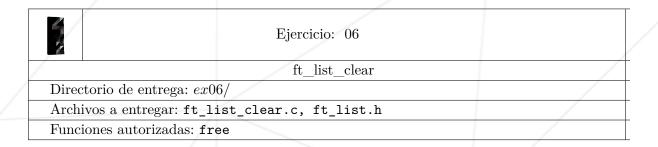


- Crea la función ft\_list\_push\_strs que cree una lista nueva e introduzca en ella las cadenas de caracteres apuntadas por los elementos de la tabla strs.
- size es el tamaño de strs
- El primer elemento de la tabla se encontrará al final de la lista.
- Se devolverá la dirección del primer elemento de la lista.
- El prototipo de la función deberá ser el siguiente:

t\_list \*ft\_list\_push\_strs(int size, char \*\*strs);

# Capítulo IX

Ejercicio 06: ft\_list\_clear

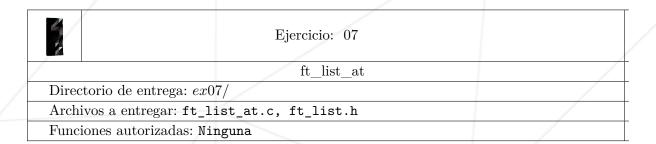


- Crea la función ft\_list\_clear que retire y libere todos los elementos de la lista.
- También tendrán que liberarse todos los data usando free\_fct.
- El prototipo de la función deberá ser el siguiente:

void ft\_list\_clear(t\_list \*begin\_list, void (\*free\_fct)(void \*));

# Capítulo X

# Ejercicio 07: ft\_list\_at

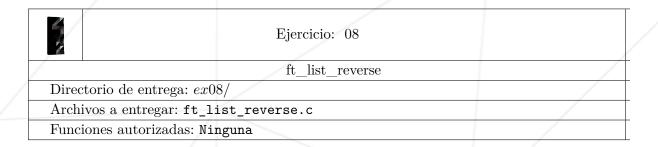


- Crea la función ft\_list\_at que devuelva el n-ésimo elemento de la lista, sabiendo que el primer elemento es el elemento 0.
- En caso de error, devolverá un puntero nulo.
- El prototipo de la función deberá ser el siguiente:

t\_list \*ft\_list\_at(t\_list \*begin\_list, unsigned int nbr);

## Capítulo XI

# Ejercicio 08 : ft\_list\_reverse

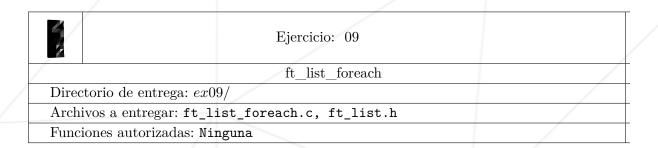


- Crea la función ft\_list\_reverse que invierta el orden de los elementos de la lista. El valor de cada elemento debe mantenerse igual.
- Atención, en este ejercicio utilizaremos nuestro propio ft\_list.h.
- El prototipo de la función deberá ser el siguiente:

void ft\_list\_reverse(t\_list \*\*begin\_list);

#### Capítulo XII

## Ejercicio 09: ft\_list\_foreach



- Crea la función ft\_list\_foreach que aplique una función pasada como parámetro al valor incluido en cada elemento de la lista.
- Se debe aplicar f en el orden de los elementos de la lista.
- El prototipo de la función deberá ser el siguiente:

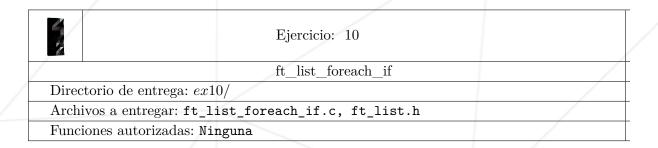
```
void ft_list_foreach(t_list *begin_list, void (*f)(void *));
```

• La función apuntada por f será utilizada de la siguiente forma:

(\*f)(list\_ptr->data);

#### Capítulo XIII

#### Ejercicio 10: ft\_list\_foreach\_if



- Crea la función ft\_list\_foreach\_if que aplique una función pasada como parámetro al valor incluido en algunos elementos de la lista.
- Solo se aplicará f a los elementos que, al ser pasados como argumento a cmp con data ref, hagan que cmp devuelva 0.
- Se debe aplicar f en el orden de los elementos de la lista.
- El prototipo de la función deberá ser el siguiente:

• Las funciones apuntadas por f y por cmp serán utilizadas de la siguiente forma:

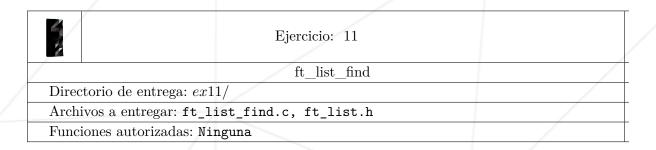
```
(*f)(list_ptr->data);
(*cmp)(list_ptr->data, data_ref);
```



La función cmp podría ser, por ejemplo, ft\_strcmp...

## Capítulo XIV

# Ejercicio 11: ft\_list\_find



- Crea la función ft\_list\_find que devuelva la dirección del primer elemento cuyos datos, cuando se los compare con data\_ref usando cmp, hagan que cmp devuelva 0
- El prototipo de la función deberá ser el siguiente:

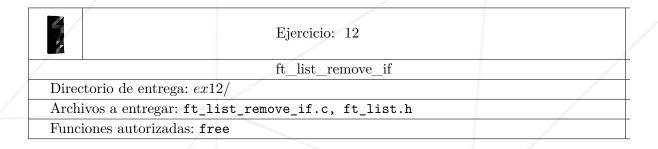
```
t_list *ft_list_find(t_list *begin_list, void *data_ref, int (*cmp)());
```

• La función apuntada por cmp será utilizada de la siguiente forma:

(\*cmp)(list\_ptr->data, data\_ref);

#### Capítulo XV

#### Ejercicio 12: ft\_list\_remove\_if



- Crea la función ft\_list\_remove\_if que borre de la lista todos los elementos cuyos datos, cuando se los compare con data\_ref usando cmp, hagan que cmp devuelva 0.
- También tendrán que liberarse todos los data de un elemento que se tenga que borrar usando free\_fct.
- El prototipo de la función deberá ser el siguiente:

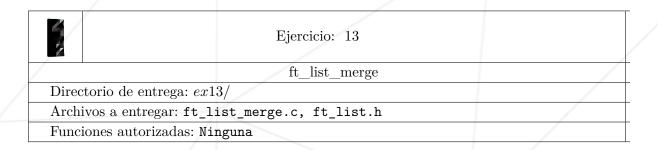
```
void ft_list_remove_if(t_list **begin_list, void *data_ref, int (*cmp)(), void (*free_fct)(void *
```

• Las funciones apuntadas por free\_fct y por cmp serán utilizadas de la siguiente forma:

```
(*cmp)(list_ptr->data, data_ref);
(*free_fct)(list_ptr->data);
```

## Capítulo XVI

# Ejercicio 13: ft\_list\_merge



- Crea la función ft\_list\_merge que coloque los elementos de una lista begin2 al final de otra lista begin1.
- No se permite la creación de elementos.
- El prototipo de la función deberá ser el siguiente:

void ft\_list\_merge(t\_list \*\*begin\_list1, t\_list \*begin\_list2);

## Capítulo XVII

# Ejercicio 14: ft\_list\_sort



- Crea la función ft\_list\_sort que ordene de forma creciente el contenido de la lista, comparando dos elementos mediante una función de comparación de datos de dos elementos.
- El prototipo de la función deberá ser el siguiente:

```
void ft_list_sort(t_list **begin_list, int (*cmp)());
```

• La función apuntada por cmp será utilizada de la siguiente forma:

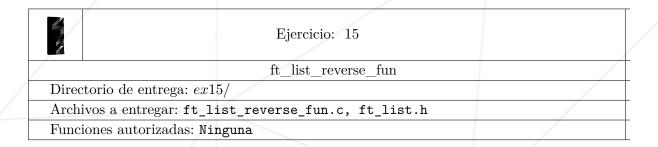
```
(*cmp)(list_ptr->data, other_list_ptr->data);
```



La función cmp podría ser, por ejemplo, ft\_strcmp.

# Capítulo XVIII

Ejercicio 15 : ft\_\_list\_\_reverse\_\_fun

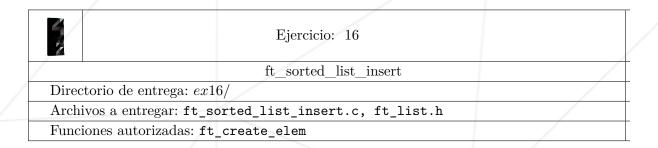


- Crea la función ft\_list\_reverse\_fun que invierta el orden de los elementos de la lista.
- El prototipo de la función deberá ser el siguiente:

void ft\_list\_reverse\_fun(t\_list \*begin\_list);

#### Capítulo XIX

# Ejercicio 16: ft\_sorted\_list\_insert



- Crea la función ft\_sorted\_list\_insert que cree un elemento nuevo y lo inserte en una lista ordenada de tal modo que la lista quede en orden creciente.
- El prototipo de la función deberá ser el siguiente:

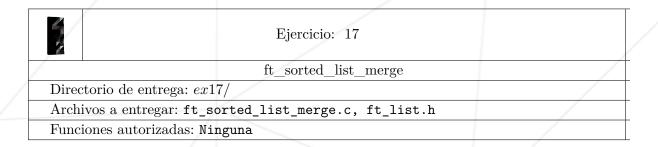
```
void ft_sorted_list_insert(t_list **begin_list, void *data, int (*cmp)());
```

• La función apuntada por cmp será utilizada de la siguiente forma:

(\*cmp)(list\_ptr->data, other\_list\_ptr->data);

#### Capítulo XX

# Ejercicio 17: ft\_sorted\_list\_merge



- Crea la función ft\_sorted\_list\_merge que integre los elementos de una lista ordenada begin2 dentro de otra lista ordenada begin1, de tal modo que la lista begin1 quede en orden creciente.
- El prototipo de la función deberá ser el siguiente:

```
void ft_sorted_list_merge(t_list **begin_list1, t_list *begin_list2, int (*cmp)());
```

• La función apuntada por cmp será utilizada de la siguiente forma:

(\*cmp)(list\_ptr->data, other\_list\_ptr->data);

# Capítulo XXI

# Entrega y evaluación

Entrega tu proyecto en tu repositorio Git como de costumbre. Solo el trabajo entregado en el repositorio será evaluado durante la defensa. No dudes en comprobar varias veces los nombres de los archivos para verificar que sean correctos.



Sólo necesitas entregar los archivos requeridos por el enunciado de este proyecto.