

Piscina C C 12

 $Sum{\'a}rio: \quad Este \ documento \ \'e \ o \ enunciado \ do \ m\'odulo \ C \ 12 \ da \ Piscina \ C \ da \ 42.$ 

Versão: 9.0

## Conteúdo

1	Preambulo		2
II	Instruções		4
III	Intruções IA		6
IV	Exercício 00 : ft_	create_elem	9
$\mathbf{V}$	Exercício 01 : ft_	_listpushfront	10
VI	Exercício 02 : ft_	_list_size	11
VII	Exercício 03 : ft_	_listlast	12
VIII	Exercício 04 : ft_	_listpushback	13
IX	Exercício $05:$ ft_	_listpushstrs	14
$\mathbf{X}$	Exercício 06 : ft_	_listclear	15
XI	Exercício $07: {\rm ft}$	_listat	16
XII	Exercício 08 : ft_	_listreverse	17
XIII	Exercício 09 : ft_	_listforeach	18
XIV	Exercício 10 : ft_	_listforeachif	19
XV	Exercício 11 : ft_	_listfind	20
XVI	Exercício 12 : ft_	_listremoveif	21
XVII	Exercício 13 : ft_	_listmerge	22
XVIII	Exercício 14 : ft_	_listsort	23
XIX	Exercício 15 : ft_	_listreversefun	24
XX	Exercício 16 : ft_	sorted_list_merge	25
XXI	Exercício 17 : ft_	sorted_list_merge	26
XXII	Submissão e avalia	acão	27

# Capítulo I Preâmbulo

ALERTA DE SPOILER NÃO LEIA A PRÓXIMA PÁGINA

#### Foi avisado.

- In Star Wars, Dark Vader is Luke's Father.
- In The Usual Suspects, Verbal is Keyser Soze.
- In Fight Club, Tyler Durden and the narrator are the same person.
- In The Sixth Sense, Bruce Willis has been dead since the beginning.
- In The others, the inhabitants of the house are ghosts and vice-versa.
- In Bambi, Bambi's mother dies.
- In The Village, monsters are the villagers and the movie actually takes place in our time.
- In Harry Potter, Dumbledore dies.
- In Planet of apes, the movie takes place on earth.
- In Game of thrones, Robb Stark and Joffrey Baratheon die on their wedding day.
- In Twilight, Vampires shine under the sun.
- In Stargate SG-1, Season 1, Episode 18, O'Neill and Carter are in Antartica.
- In The Dark Knight Rises, Miranda Tate is Talia Al'Gul.
- In Super Mario Bros, The princess is in another castle.

#### Capítulo II

#### Instruções

- Somente este documento servirá de referência; não confie nos boatos.
- Releia bem o enunciado antes de entregar os seus exercícios. A qualquer momento pode haver alterações.
- Tenha atenção aos direitos dos seus ficheiros e pastas.
- Deverá seguir o procedimento de entrega para todos os exercícios.
- Os seus exercícios serão corrigidos pelos seus colegas de piscine.
- Além dos seus colegas, a Moulinette também corrigirá os seus exercícios.
- A Moulinette é extremamente rígida na sua avaliação. É completamente automatizada, e é impossível discutir a sua nota com ela. Portanto, seja rigoroso!
- A Moulinette não tem uma mente muito aberta: não tenta entender código que não respeita a Norma. A Moulinette utiliza o programa norminette para verificar a norma dos ficheiros. Seria uma tontice entregar código que não passa pela norminette...
- Os exercícios são ordenados precisamente do mais simples ao mais complexo. Em caso algum consideraremos um exercício mais complexo se outro mais simples não tiver sido perfeitamente realizado.
- A utilização de qualquer função proibida é um caso de fraude. Qualquer fraude é punida com nota de -42.
- Deve entregar uma função main() se for pedido um programa.
- A Moulinette compila com as textitflags -Wall -Wextra -Werror, e utiliza cc.
- Se o seu programa não compila, terá 0.
- <u>Não deve</u> deixar no repositório de entrega <u>nenhum</u> outro ficheiro além daqueles explicitamente especificados pelo enunciado dos exercícios.

Piscina C

C 12

- Tem alguma dúvida? Pergunte ao seu vizinho da direita. Tente, também, com o seu vizinho da esquerda.
- A bibliografia para consulta chama-se Google / man / Internet / ....
- Considere discutir os exercícios no Slack da sua piscine!
- Leia atentamente os exemplos: podem demonstrar coisas que não estão especificadas no enunciado...
- Para os seguintes exercicios, é necessário usar a estrutura seguinte:

- Deves incluir esta estrutura no ficheiro ft\_list.h e entregar esse ficheiro em cada exercicio.
- A partir do exercicio 01, iremos usar o nosso ft\_create\_elem, então tem isso em consideração (pode ser util ter o prototipo no ficheiro ft\_list.h...).

#### Capítulo III

#### Intruções IA

#### Contexto

A Piscina C é intensa. É o teu primeiro grande desafio na 42 — um mergulho profundo na resolução de problemas, autonomia e comunidade.

Nesta fase, o teu principal objetivo é obter uma base sólida — através do esforço, da repetição e, acima de tudo, da partilha de aprendizagens com os teus colegas.

Na era da IA, os atalhos são fáceis de encontrar. No entanto, é importante considerar se o uso da IA está realmente a ajudar-te a crescer — ou apenas a impedir-te de desenvolver competências reais.

A Piscine também é uma experiência humana — e, por agora, nada substitui isso. Nem mesmo a IA.

Para uma visão mais completa da nossa posição sobre a IA — como ferramenta de aprendizagem, como parte do currículo TIC e como uma expectativa crescente no mercado de trabalho — consulta o FAQ dedicado disponível no intranet.

#### Mensagem principal

- Constrói bases sólidas sem atalhos.
- Desenvolve verdadeiramente competências técnicas e interpessoais.
- Vive a aprendizagem entre pares, começa a aprender a aprender e a resolver novos problemas.
- A jornada de aprendizagem é mais importante do que o resultado.
- Aprende os riscos associados à IA e desenvolve práticas de controlo eficazes e contramedidas para evitar os erros mais comuns.

#### Regras para os alunos:

- Deves aplicar o raciocínio nas tarefas atribuídas, especialmente antes de recorreres à IA.
- Não deves pedir respostas diretas à IA.
- Deves aprender sobre a abordagem global da 42 em relação à IA.

#### Resultados esperados:

Nesta fase, vais ter os seguintes resultados:

- Obter bases sólidas em tecnologia e programação.
- Compreender por que razão e de que forma a IA pode ser perigosa durante esta fase.

#### Comentários e exemplos:

- Sim, sabemos que a IA existe e sim, pode resolver os teus projetos. Mas estás aqui para aprender, não para provar que a IA já aprendeu. Não percas tempo (nem o nosso) apenas para demonstrar que a IA consegue resolver o problema.
- Aprender na 42 não é sobre saber a resposta é sobre desenvolver a capacidade de encontrar uma. A IA dá-te a resposta diretamente, mas isso impede-te de construir o teu próprio raciocínio. E o raciocínio exige tempo, esforço e envolve falhas. O caminho para o sucesso não deve ser fácil.
- Lembra-te que nos exames a IA não está disponível sem internet, sem telemóveis, etc. Vais perceber rapidamente se dependeste demasiado da IA no teu processo de aprendizagem.
- A aprendizagem entre pares expõe-te a ideias e abordagens diferentes, melhorando as tuas competências interpessoais e a tua capacidade de pensar de forma divergente.
   Isso é muito mais valioso do que conversar com um bot. Por isso, não sejas tímido — fala, faz perguntas e aprende em conjunto!
- Sim, a IA fará parte do currículo tanto como ferramenta de aprendizagem como tema de estudo. Terás até a oportunidade de construir o teu próprio software de IA.
   Para saberes mais sobre a nossa abordagem em crescendo, consulta a documentação disponível no intranet.

#### ✓ Boa prática:

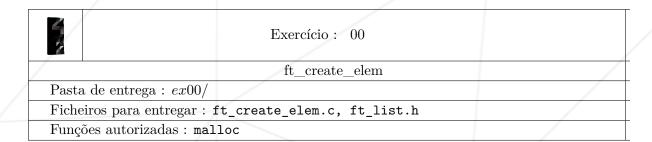
Estou com dificuldades num novo conceito. Pergunto a alguém ao meu lado como o abordou. Falamos durante 10 minutos — e de repente faz sentido. Percebo.

#### X Má prática:

Uso a IA em segredo, copio algum código que parece estar certo. Durante a avaliação por pares, não consigo explicar nada. Falho. Durante o exame — sem IA — fico novamente bloqueado. Falho.

#### Capítulo IV

Exercício 00 : ft\_create\_elem



- Escreve a função ft\_create\_elem que cria um novo elemento do tipo t\_list.
- Deve atribuir data ao parâmetro fornecido e next a NULL.
- Deve ser prototipada da seguinte forma:

## Capítulo V

## Exercício 01 : ft\_list\_push\_front

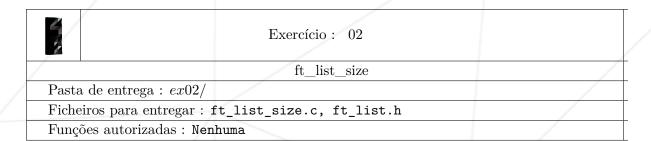
	Exercício: 01	
/	ft_list_push_front	
Pasta de entrega : $ex01/$		/
Ficheiros para entregar : f	t_list_push_front.c, ft_list.h	/
Funções autorizadas : ft_	create_elem	

- Escreve a função ft\_list\_push\_front que acrescenta ao início da lista um novo elemento de tipo t\_list.
- Deve atribuir data ao parâmetro fornecido.
- Se necessário, vai atualizar o ponteiro para o início da lista.
- Deve ser prototipada da seguinte forma:

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

## Capítulo VI

Exercício 02 : ft\_list\_size

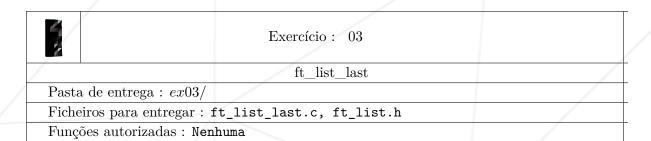


- Escreve a função ft\_list\_size que retorna o número de elementos da lista.
- Deve ser prototipada da seguinte forma:

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

## Capítulo VII

Exercício 03 : ft\_list\_last

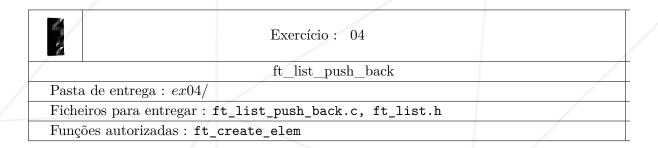


- Escreve a função ft\_list\_last que retorna o último elemento da lista.
- Deve ser prototipada da seguinte forma:

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

#### Capítulo VIII

#### Exercício 04 : ft\_list\_push\_back



- Escreve a função ft\_list\_push\_back que acrescenta no final da lista um novo elemento de tipo t\_list.
- Deve atribuir data ao parâmetro fornecido.
- Se necessário, vai atualizar o ponteiro para o início da lista.
- Deve ser prototipada da seguinte forma:

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

#### Capítulo IX

## Exercício 05 : ft\_list\_push\_strs

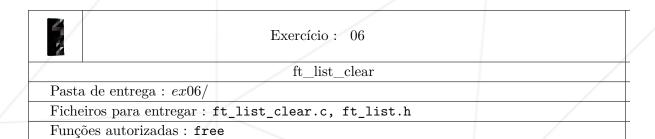
	Exercício: 05	
/	ft_list_push_strs	
Pasta de entrega : $ex05/$		
Ficheiros para entregar :	ft_list_push_strs.c, ft_list.h	/
Funções autorizadas : ft	_create_elem	/

- Escreve a função ft\_list\_push\_strs que cria uma nova lista, incluindo nela todas as strings apontadas pelos elementos do array strs.
- size é o tamanho de strs.
- O primeiro elemento do deve estar no final da lista.
- O endereço do primeiro elemento da lista é retornado.
- Deve ser prototipada da seguinte forma:

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

#### Capítulo X

Exercício 06 : ft\_list\_clear

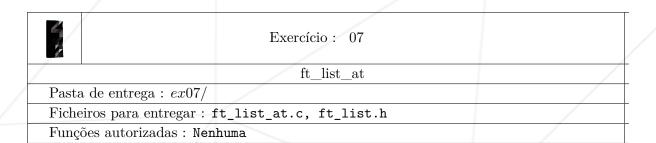


- Escreve a função ft\_list\_clear que remove e liberta todos os elementos da lista.
- O free\_fct é usado para libertar cada data
- Deve ser prototipada da seguinte forma:

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

#### Capítulo XI

Exercício 07 : ft\_list\_at



- Escreve a função ft\_list\_at que retorna o n-ésimo elemento da lista, sabendo que o primeiro elemento é quando o nbr for 0
- Em caso de erro, retorna um ponteiro nulo.
- Deve ser prototipada da seguinte forma:

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

## Capítulo XII

Exercício 08 : ft\_list\_reverse

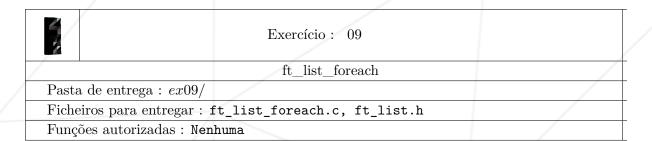
	Exercício: 08	
/	ft_list_reverse	/
Pasta de entrega : $ex08/$		
Ficheiros para entregar :	ft_list_reverse.c	/
Funções autorizadas : Nen	huma	

- Escreve a função ft\_list\_reverse que inverte a ordem dos elementos da lista. O valor de cada elemento deve manter se o mesmo.
- Atenção: neste exercício vamos usar nosso próprio ft\_list.h
- Deve ser prototipada da seguinte forma:

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

#### Capítulo XIII

Exercício 09 : ft\_list\_foreach



- Escreve a função ft\_list\_foreach que aplica uma função fornecida como parâmetro a cada elemento da lista.
- f deve ser aplicada na ordem dos elementos da lista
- Deve ser prototipada da seguinte forma:

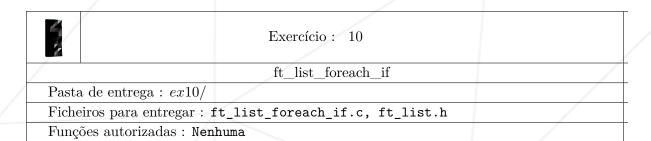
```
void ft_list_foreach(t_list *begin_list, void (*f)(void *));
```

• A função apontada por f será utilizada da seguinte forma:

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

#### Capítulo XIV

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



- Escreve a função ft\_list\_foreach\_if que aplica uma função dada como parâmetro em determinados elementos da lista.
- f só será aplicada nos elementos quando o cmp com data ref, cmp retornem 0
- f deve ser aplicada na ordem dos elementos da lista
- Deve ser prototipada da seguinte forma:

```
void ft_list_foreach_if(t_list *begin_list, void (*f)(void *), void
*data_ref, int (*cmp)())
```

• As funções apontadas por f e por cmp serão usadas da seguinte forma:

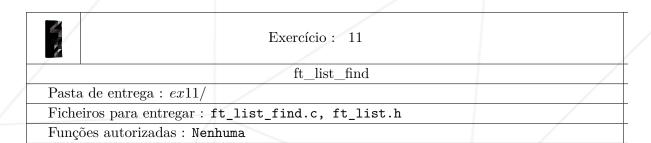
```
(*f)(list_ptr->data);
(*cmp)(list_ptr->data, data_ref);
```



A função cmp pode ser, por exemplo, ft\_strcmp...

#### Capítulo XV

## Exercício 11: ft\_list\_find



- Escreve a função ft\_list\_find que retorna o endereço do primeiro elemento cujo data comparado a data\_ref com cmp faz com que cmp retorne 0.
- Deve ser prototipada da seguinte forma:

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

• A função apontada por cmp será usada da seguinte forma:

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

#### Capítulo XVI

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

	Exercício: 12	
	ft_list_remove_if	
Pasta de entrega : $ex12/$		/
Ficheiros para entregar : f	t_list_remove_if.c, ft_list.h	/
Funções autorizadas : fre	е	/

- Escreve a função ft\_list\_remove\_if que apaga da lista todos os elementos cujo a data comparado a data\_ref, com o auxílio de cmp faz com que cmp retorne 0.
- O data de um elemento que será apagado deverá também ser libertado com o auxílio de free\_fct
- Deve ser prototipada da seguinte forma:

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

• As funções apontadas por free\_fct e por cmp serão usadas da seguinte forma:

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

## Capítulo XVII

## Exercício 13: ft\_list\_merge

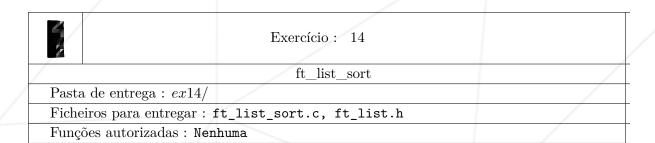
	Exercício: 13	
/	ft_list_merge	
Pasta de entrega : $ex13/$		/
Ficheiros para entregar : ft_list_merge.c, ft_list.h		/
Funções autorizadas : Nei	nhuma	/

- Escreve a função ft\_list\_merge que coloca os elementos de uma lista begin2 no fim de outra lista begin1.
- A criação de elementos não é permitida.
- Deve ser prototipada da seguinte forma:

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

#### Capítulo XVIII

## Exercício 14: ft\_list\_sort



- Escreve a função ft\_list\_sort que organiza em ordem crescente o conteúdo da lista, ao comparar a data de dois elementos com uma função
- Deve ser prototipada da seguinte forma:

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

• A função apontada por cmp será usada da seguinte forma:

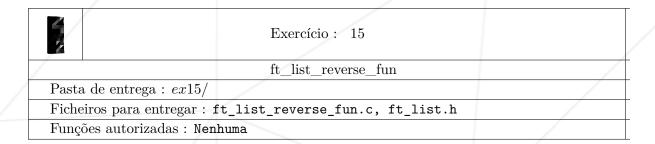
```
(*cmp)(list_ptr->data, other_list_ptr->data);
```



A função cmp pode ser, por exemplo, ft\_strcmp.

## Capítulo XIX

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

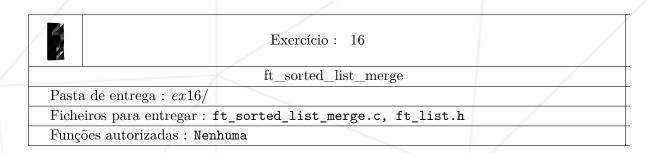


- $\bullet\,$  Escreve a função ft\_list\_reverse\_fun que inverte a ordem dos elementos da lista.
- Deve ser prototipada da seguinte forma:

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

## Capítulo XX

# Exercício 16: ft\_sorted\_list\_merge



- Escreve a função ft\_sorted\_list\_merge que integra os elementos de uma lista organizada begin2 em uma outra lista organizada begin1, de modo que a lista begin1 fique em ordem crescente.
- Deve ser prototipada da seguinte forma:

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

• A função apontada por cmp será usada da seguinte forma:

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

#### Capítulo XXI

## Exercício 17: ft\_sorted\_list\_merge

/
/

- Escreve a função ft\_sorted\_list\_merge que integra os elementos de uma lista organizada begin2 em uma outra lista organizada begin1, de modo que a lista begin1 fique em ordem crescente.
- Deve ser prototipada da seguinte forma:

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

• A função apontada por cmp será usada da seguinte forma:

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

## Capítulo XXII

#### Submissão e avaliação

Entrega o teu trabalho no teu repositório Git, como habitual. Apenas o trabalho dentro do teu repositório será avaliado durante a defesa. Não hesites em confirmar os nomes dos teus ficheiros para ter a certeza que estão corretos.



Apenas precisas de entregar os ficheiros pedidos no enunciado deste projeto.