

Piscina C C 13

 $Sum{\'a}rio:\ ESTE\ documento\ \'e\ o\ enunciado\ do\ m\'odulo\ C\ 13\ da\ Piscina\ C\ da\ 42.$

Versão: 4.2

Conteúdo

1	Instruções	2
II	Preâmbulo	4
III	Exercice 00 : btree_create_node	5
IV	Exercício 01 : btree_apply_prefix	6
\mathbf{V}	Exercício 02 : btree_apply_infix	7
VI	Exercício 03 : btree_apply_suffix	8
VII	Exercício 04 : btree_insert_data	9
VIII	Exercício 05 : btree_search_item	10
IX	Exercício 06 : btree_level_count	11
\mathbf{X}	Exercício 07 : btree_apply_by_level	12
XI	Submissão e avaliação	13

Capítulo I

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.
- Você <u>não deve</u> deixar em sua pasta <u>nenhum</u> outro arquivo além daqueles explicitamente especificados pelos enunciados dos exercícios.

Piscina C

• 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...
- <u>Não deve</u> deixar no repositório de entrega <u>nenhum</u> outro ficheiro além daqueles explicitamente especificados pelo enunciado dos exercícios.
- 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 exercícios, será usada a seguinte estrutura :

```
typedef struct s_btree
{
    struct s_btree *left;
    struct s_btree *right;
    void *item;
}
```

- Deverá incluir esta estrutura no ficheiro ft_btree.h e submeter o ficheiro em cada exercício.
- A partir do exercício 01, a função btree_create_node será usada regularmente, por isso ajuste o seu código (poderá ser útil ter o seu protótipo no ficheiro ft_btree.h...).

Capítulo II

Preâmbulo

Here's the list of releases for Venom:

- In League with Satan (single, 1980)
- Welcome to Hell (1981)
- Black Metal (1982)
- Bloodlust (single, 1983)
- Die Hard (single, 1983)
- Warhead (single, 1984)
- At War with Satan (1984)
- Hell at Hammersmith (EP, 1985)
- American Assault (EP, 1985)
- Canadian Assault (EP, 1985)
- French Assault (EP, 1985)
- Japanese Assault (EP, 1985)
- Scandinavian Assault (EP, 1985)
- Manitou (single, 1985)
- Nightmare (single, 1985)
- Possessed (1985)
- German Assault (EP, 1987)
- Calm Before the Storm (1987)
- Prime Evil (1989)
- Tear Your Soul Apart (EP, 1990)
- Temples of Ice (1991)
- The Waste Lands (1992)
- Venom '96 (EP, 1996)
- Cast in Stone (1997)
- Resurrection (2000)
- Anti Christ (single, 2006)
- Metal Black (2006)
- Hell (2008)
- Fallen Angels (2011)

Today's subject will seem easier if you listen to Venom.

Capítulo III

Exercice 00: btree_create_node

	Exercício: 00	
	btree_create_node	/
Pasta de entrega : $ex00/$		
Ficheiros para entregar : t	otree_create_node.c, ft_btree.h	/
Funções autorizadas : mal	loc	

- Escreva a função btree_create_node que aloca um novo elemento. Inicialize o novo item com o valor do parâmetro passado e todos os outros elementos com 0.
- O endereço do nó criado é retornado.
- Deverá ser prototipada da seguinte forma:

t_btree *btree_create_node(void *item);

Capítulo IV

Exercício 01 : btree_apply_prefix

	Exercício: 01	
/	btree_apply_prefix	/
Pasta de entrega : $ex01/$		
Ficheiros para entregar :	btree_apply_prefix.c, ft_btree.h	/
Funções autorizadas : Ner	huma	/

- Escreva a função btree_apply_prefix que aplica a função passada como parâmetro ao item de cada nó, usando prefix traversal para percorrer a árvore.
- Deverá ser prototipada da seguinte forma:

void btree_apply_prefix(t_btree *root, void (*applyf)(void *));

Capítulo V

Exercício 02 : btree_apply_infix

	Exercício: 02	
/	btree_apply_infix	/
Pasta de entrega : $ex02/$		
Ficheiros para entregar :	btree_apply_infix.c, ft_btree.h	/
Funções autorizadas : Ne:	nhuma	/

- Escreva a função btree_apply_infix que aplica a função passada como parâmetro ao item de cada nó, usando infix traversal para percorrer a árvore.
- Ela deverá ser prototipada da seguinte forma:

void btree_apply_infix(t_btree *root, void (*applyf)(void *));

Capítulo VI

Exercício 03 : btree_apply_suffix

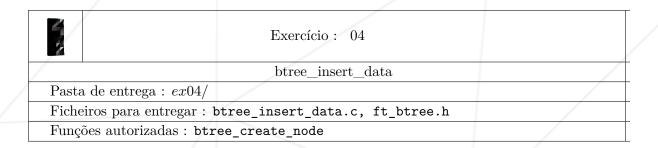
	Exercício: 03	
	btree_apply_suffix	
Pasta de entrega : $ex03/$		/
Ficheiros para entregar :	btree_apply_suffix.c, ft_btree.h	
Funções autorizadas : Ne	nhuma	/

- Escreva a função btree_apply_suffix que aplica a função passada como parâmetro ao item de cada nó, usando sufix traversal para percorrer a árvore.
- Deverá ser prototipada da seguinte forma:

void btree_apply_suffix(t_btree *root, void (*applyf)(void *));

Capítulo VII

Exercício 04 : btree_insert_data

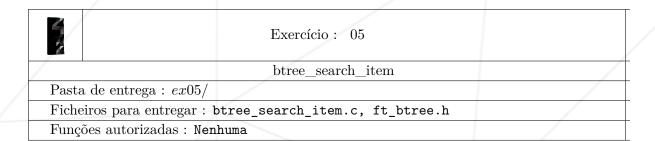


- Escreva a função btree_insert_data que insere o elemento item em uma árvore. A árvore passada como parâmetro será organizada: por cada nó todos os elementos inferiores encontram-se à esquerda e todos os elementos superiores ou iguais, à direita. Será enviada como parâmetro uma função de comparação que tem o mesmo comportamento que strcmp.
- O parâmetro root aponta para o nó raiz da árvore. Na primeira chamada, deverá apontar para NULL.
- Deverá ser prototipada da seguinte forma:

void btree_insert_data(t_btree **root, void *item, int (*cmpf)(void *, void *));

Capítulo VIII

Exercício 05: btree_search_item



- Escreva a função btree_search_item que retorna o primeiro elemento correspondente ao dado de referência passado como parâmetro. A árvore deverá ser percorrida usando infix traversal. Se o elemento não for encontrado, a função deverá retornar NULL.
- Deverá ser prototipada da seguinte forma:

void *btree_search_item(t_btree *root, void *data_ref, int (*cmpf)(void *, void *));

Capítulo IX

Exercício 06: btree_level_count

	Exercício: 06	
	btree_level_count	/
Pasta de entrega : $ex06/$		/
Ficheiros para entregar : btree_level_count.c, ft_btree.h		/
Funções autorizadas : Ner	nhuma	/

- Escreva a função btree_level_count que retorna a profundidade do maior ramo passado como parâmetro.
- Deverá ser prototipada da seguinte forma:

int btree_level_count(t_btree *root);

Capítulo X

Exercício 07: btree_apply_by_level

-	Exercício: 07	
	btree_apply_by_level	/
Pasta	a de entrega : $ex07/$	
Fiche	piros para entregar : btree_apply_by_level.c, ft_btree.h	/
Funç	ões autorizadas : malloc, free	/

- Escreva a função btree_apply_by_level que aplica a função passada como parâmetro a cada nó da árvore. A árvore deve percorrer nível de profundidade por nível de profundidade. A função chamada terá três parâmetros:
 - o O primeiro parâmetro, do tipo void *, corresponde ao item do nó;
 - O segundo parâmetro, do tipo int, corresponde ao nível de profundidade no qual estamos: 0 para a raiz, 1 para seus filhos, 2 para seus netos, etc. ;
 - o O terceiro parâmetro, do tipo int, vale 1 se for o primeiro nó do nível, caso contrário vale 0.
- Deverá ser prototipada da seguinte forma:

void btree_apply_by_level(t_btree *root, void (*applyf)(void *item, int current_level, int is_firs

Capítulo XI

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.