

Piscina C C 13

 $Sommario: \quad Questo \ documento \ tratta \ il \ modulo \ C \ 13 \ della \ Piscina \ C \ @ \ 42.$

Indice

1	Istruzioni	2
II	Preambolo	4
III	Esercizio 00 : btree_create_node	5
IV	Esercizio 01 : btree_apply_prefix	6
\mathbf{V}	Esercizio 02 : btree_apply_infix	7
VI	Esercizio 03 : btree_apply_suffix	8
VII	Esercizio 04 : btree_insert_data	9
VIII	Esercizio 05 : btree_search_item	10
IX	Esercizio 06 : btree_level_count	11
\mathbf{X}	Esercizio 07 : btree_apply_by_level	12
XI	Consegna e valutazione tra pari	13

Capitolo I

Istruzioni

- Fate riferimento solo a questa pagina: non fidatevi delle dicerie.
- Questo documento può subire variazioni prima della scadenza per la presentazione.
- Controllate i permessi dei vostri file e delle vostre cartelle.
- Dovete seguire le procedure di presentazione per tutti gli esercizi.
- I vostri esercizi saranno controllati e valutati dai vostri compagni di corso.
- Moulinette sarà estremamente meticolosa e severa nel valutare il vostro lavoro. Essendo il suo un processo automatico senza possibilità di ricorso, assicuratevi di essere il più precisi possibile al fine di evitare brutte sorprese.
- I vostri esercizi saranno soggetti, oltre alla valutazione tra pari, al controllo e alla valutazione da parte di un programma chiamato Moulinette.
- Moulinette non ha una mentalità aperta. Non proverà a comprendere il vostro codice se non rispetta la Norma. Moulinette utilizza un programma di nome norminette per controllare la validità dei vostri file. TL;DR: sarebbe scocco tentare di consegnare un esercizio che non pass il controllo di norminette.
- Gli esercizi sono presentati seguendo un ordine di difficoltà crescente. Ai fini della valutazione NON si prendono in considerazione gli esercizi se i precedenti non sono stati completati correttamente
- Usare una funzione non autorizzata viene considerato come barare. Chi bara ottiene un -42 senza possibilità di ricorso.
- Dovrete consegnare una funzione main() solo se l'esercizio richiede un programma.
- Moulinette compila per mezzo di gcc utilizzando queste flag: -Wall -Wextra Werror.
- Se il vostro programma non compila, il voto sarà 0.
- <u>NON</u> sarà tollerato <u>ALCUN</u> file aggiuntivo nelle cartelle presentate oltre a quelli specificati in questo documento.

Piscina C

• Dubbi o domande? Chiedi a chi si trova alla tua destra, altrimenti a chi si trova alla tua sinistra

- Your reference guide is called Google / man / the Internet /
- Date un occhiata alla sezione Piscina C del forum dell Intranet.
- Prestate attenzione agli esempi proposti, in quanto potrebbero mostrare dettagli non esplicitamente presentati nel documento...
- Per Odin, Per Thor! Usate la testa!!!
- Per gli esercizi presenti in questo documento dovrete usare la seguente struttura :

- Dovrà essere inclusa nel file ft_btree.h da consegnare per ogni esercizio.
- Useremo il nostro btree_create_node dall'esercizio 01 in poi, agite quindi di conseguenza (potrebbe essere utile averne il prototipo nel file ft_btree.h...).

Capitolo II

Preambolo

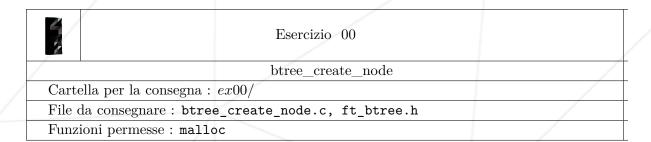
Here's the list of releases for Venom:

- 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.

Capitolo III

Esercizio 00 : btree_create_node

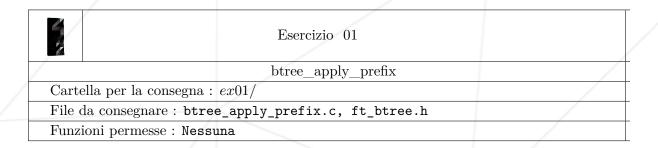


- Creare la funzione btree_create_node che alloca un nuovo elemento. Deve impostare come valore di item quello dell'argomento e 0 a tutti gli altri.
- Restituirà l'indirizzo del nodo creato.
- Il prototipo è il seguente :

t_btree *btree_create_node(void *item);

Capitolo IV

Esercizio 01 : btree_apply_prefix



- Creare la funzione btree_apply_prefix che applica la funzione data come argomento all'elemento item di ogni nodo utilizzando un algoritmo di visita in preordine.
- Il prototipo è il seguente :

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

Capitolo V

Esercizio 02 : btree_apply_infix

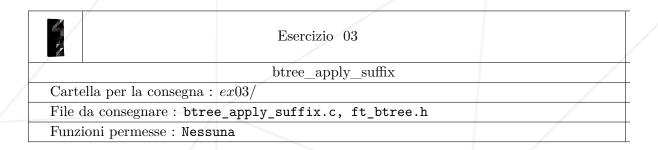
	Esercizio 02	
/	btree_apply_infix	
Cartella per la consegna	/	
File da consegnare : btre	/	
Funzioni permesse : Ness	/	

- Creare la funzione btree_apply_infix che applica la funzione data come argomento all'elemento item di ogni nodo utilizzando un algoritmo di visita in ordine.
- Il prototipo è il seguente :

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

Capitolo VI

Esercizio 03: btree_apply_suffix

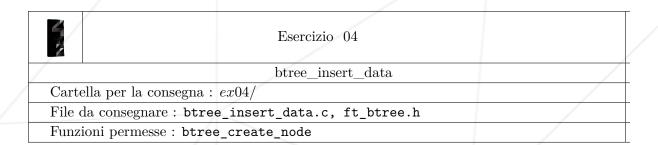


- Creare la funzione btree_apply_sufflix che applica la funzione data come argomento all'elemento item di ogni nodo utilizzando un algoritmo di visita in postordine.
- Il prototipo è il seguente :

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

Capitolo VII

Esercizio 04 : btree_insert_data

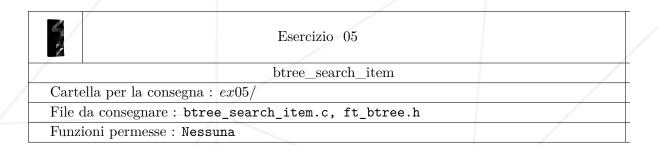


- Creare la funzione btree_insert_data che inserisce l'elemento item in un albero binario. Ogni nodo sarà ordinato posizionando tutti gli elementi minori nel lato sinistro e i maggiori sul lato sinistro. La funzione usata per comparare sarà una funzione simile a strcmp.
- L'elemento root punta alla radice dell'albero. La prima volta che viene chiamato dovrebbe puntare a NULL.
- Il prototipo è il seguente :

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

Capitolo VIII

Esercizio 05 : btree_search_item

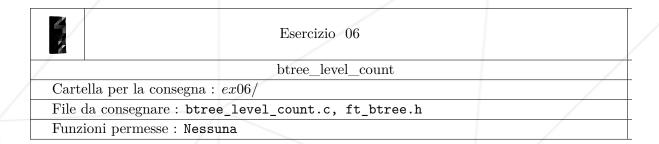


- Creare la funzionebtree_search_item che restituisce il primo elemento dell'albero che ha una relazione con l'argomento data_ref. Utilizzerete un algoritmo di visita in ordine. Se non trova nessun elemento deve restituire NULL.
- Il prototipo è il seguente :

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

Capitolo IX

Esercizio 06: btree_level_count



- Creare la funzione btree_level_count che restituisce la dimensione del ramo più grande passato come argomento.
- Il prototipo è il seguente :

int btree_level_count(t_btree *root);

Capitolo X

Esercizio 07: btree_apply_by_level

4	Esercizio 07	
	btree_apply_by_level	/
Carte	ella per la consegna : $ex07/$	
File o	da consegnare: btree_apply_by_level.c, ft_btree.h	/
Funz	ioni permesse : malloc, free	/

- Creare la funzione btree_apply_by_level che applica la funzione passata come argomento ad ogni nodo dell'albero. L'albero deve essere attraversato un livello per volta. La funzione data come argomento avrà 3 argomenti:
 - o Il primo, di tipo void *, corrisponde all'elemento item del nodo ;
 - Il secondo, di tipo int, rappresenta il livello in cui siamo: 0 per il root, 1 per il figlio, 2 per i nipoti, etc.;
 - o Il terzo, di tipo int, sarà 1 se è il primo nodo del livello, 0 negli altri casi.
- Il prototipo è il seguente :

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

Capitolo XI

Consegna e valutazione tra pari

Consegna gli esercizi nella tuo repository Git come al solito. Durante la difesa verrà considerato unicamente ciò che si trova all'interno della repository. Assicurati di controllare che i nomi dei tuoi file siano corretti.



Devi consegnare solo i file richiesti da questo documento.