Chapter III

Exercise 00 : btree_create_node

	Exercice : 00
btree_create_node	
Turn-in directory : *ex*00/	
Files to turn in : `btree_create_node.c, ft_btree.h`	
Allowed functions : `malloc`	
Remarks : `n/a`	

- Create the function `btree_create_node` which allocates a new element. It should initialise its `item` to the argument's value, and all other elements to 0.
- The created node's address is returned.
- Here's how it should be prototyped :

```
t_btree *btree_create_node(void *item);
```

Chapter IV

Exercise 01 : btree_apply_prefix

	Exercice : 01
btree_apply_prefix	
Turn-in directory : $ex01/$	
Files to turn in : `btree_apply_prefix.c, ft_btree.h`	
Allowed functions : `Nothing`	
Remarks : `n/a`	

- Create a function `btree_apply_prefix` which applies the function given as argument to the `item` of each node, using `prefix traversal` to search the tree.
- Here's how it should be prototyped :

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

Chapter VIII

Exercise 05 : btree_search_item

	Exercice : 05
btree_search_item	
Turn-in directory : $ex05/$	
Files to turn in : `btree_search_item.c, ft_btree.h`	
Allowed functions : `Nothing`	
Remarks : `n/a`	

- Create a function `btree_search_item` which returns the first element related to the reference data given as argument. The tree should be browsed using `infix traversal` . If the element isn't found, the function should return `NULL`.

- Here's how it should be prototyped :

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

Chapter IX

Exercise 06 : btree_level_count

	Exercice : 06
btree_level_count	
Turn-in directory : $ex06/$	
Files to turn in : `btree_level_count.c, ft_btree.h`	
Allowed functions : `Nothing`	
Remarks : `n/a`	

- Create a function `btree_level_count` which returns the size of the largest branch passed as argument.
- Here's how it should be prototyped :

```
int btree_level_count(t_btree *root);
```