

C Piscine C 13

Summary: このドキュメントは、 $C\ Piscine\ @\ 42$ の  $C\ 13$ モジュール用の課題である。

# Contents

1	mstructions	4
II	Foreword	4
III	Exercise 00 : btree_create_node	5
IV	Exercise 01 : btree_apply_prefix	6
V	Exercise 02 : btree_apply_infix	7
VI	Exercise 03 : btree_apply_suffix	8
VII	Exercise 04 : btree_insert_data	9
VIII	Exercise 05 : btree_search_item	10
IX	Exercise 06 : btree_level_count	11
$\mathbf{X}$	Exercise 07 : btree_apply_by_level	12

## Chapter I

#### Instructions

- 課題に関する噂に惑わされないよう気をつけ、信用しないこと。
- この書類は、提出前に変更になる可能性があるため、気をつけること。
- ファイルとディレクトリへの権限があることを、あらかじめ確認すること。
- すべての課題は、提出手順に従い行うこと。
- 課題の確認と評価は、あなたの周りにいるPiscine受験者により行われる。
- 課題の確認と評価は、Piscine受験者に加えて、Moulinetteと呼ばれるプログラムによっても行われる。
- Moulinetteは、大変細かい評価を行う。これはすべて自動で行われるため、交 渉の余地はない。
- Moulinetteは、コーディング規範(Norm)を遵守しないコードを解読することができない。そのため、Moulinetteはnorminetteと呼ばれるプログラムを使用し、あなたのファイルがコーディング規範を遵守しているか確認を行う。せっかくの取り組みが、norminetteの確認により無駄にならないよう、気をつけること。
- 問題は、簡単なものから徐々に難しくなるように並べられている。簡単な問題が解けていない場合は、難しい問題が解けていたとしても 加点されることはない。
- 使用が禁止されている関数を使用した場合は、不正とみなされる。不正者は-42の評価をつけられ、この評価に対する交渉の余地はない。
- 課題が<u>プログラム</u>の提出を要求する際は、main()関数のみを提出すること。
- Moulinetteは以下のフラッグを用いて、gccでコンパイルする。 -Wall -Wextra -Werror
- プログラムがコンパイルされなかった場合、評価は0になる。
- 課題で指定されていないものは、 $\underline{\underline{V}}$ んなファイルもディレクトリ内に置かないこと。

- 質問がある場合は、隣の人に聞くこと。それでも分からない場合は、反対側の 席の人に聞くこと。
- 助けてくれるのは、Google / 人間 / インターネット / ...と呼ばれているものたちである。
- 出力例には、問題文に明記されていない細部まで表示されている場合があるため、入念に確認すること。
- この課題では、以下の構造体を使用すること。

- 上記の構造体がft\_btree.hファイルの中に含まれていることを確認し、各問題を提出すること。
- Exercise 01以降は、btree\_create\_nodeを使用する。ft\_list.hファイルにプロトタイプを入れておくと、便利な場合がある。

#### Chapter II

#### Foreword

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.

# Chapter III

Exercise 00: btree\_create\_node



#### Exercise 00

btree\_create\_node

提出するディレクトリ: *ex*00/

提出するファイル: btree\_create\_node.c, ft\_btree.h

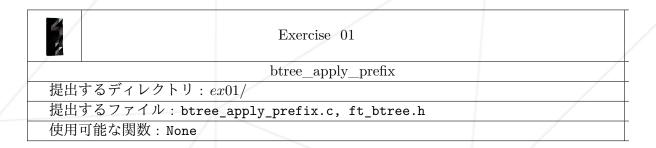
使用可能な関数: malloc

- 新たな要素を作成するために、動的にメモリを確保する関数btree\_create\_nodeを作成せよ。要素のitemを引数の値に初期化し、それ以外のすべての要素を0に初期化すること。
- 作成したノードのアドレスが返される。
- プロトタイプ例)

t\_btree \*btree\_create\_node(void \*item);

# Chapter IV

Exercise 01: btree\_apply\_prefix

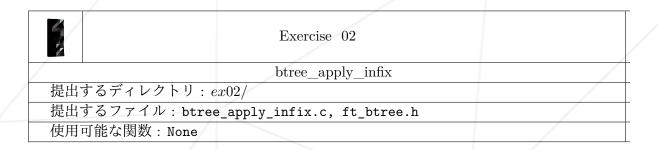


- prefix traversalを利用してツリーを検索し、引数として与えられた関数を、各ノードのitemに適用する関数btree\_apply\_prefixを作成せよ。
- プロトタイプ例)

void btree\_apply\_prefix(t\_btree \*root, void (\*applyf)(void \*));

# Chapter V

Exercise 02: btree\_apply\_infix

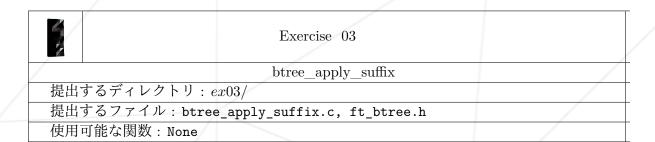


- infix traversalを利用してツリーを検索し、引数として与えられた関数を、各ノードのitemに適用する関数btree\_apply\_prefixを作成せよ。
- プロトタイプ例)

void btree\_apply\_infix(t\_btree \*root, void (\*applyf)(void \*));

# Chapter VI

Exercise 03: btree\_apply\_suffix

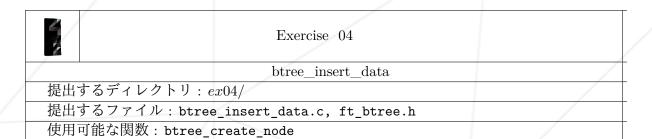


- suffix traversalを利用してツリーを検索し、引数として与えられた関数を、各ノードのitemに適用する関数btree\_apply\_prefixを作成せよ。
- プロトタイプ例)

void btree\_apply\_suffix(t\_btree \*root, void (\*applyf)(void \*));

### Chapter VII

Exercise 04: btree\_insert\_data



- 要素のitemをツリーに挿入する関数btree\_insert\_dataを作成せよ。引数として渡されたツリーは、既に並べ替えられている。nodeごとに、すべての下位の要素は左側に配置され、すべての上位、または、同等の要素は右側に配置される。strcmpと同様の比較関数を、引数として渡す。
- 引数rootは、ツリーのルートノードを指す。最初に呼び出される際は、NULLを 指していること。
- プロトタイプ例)

void btree\_insert\_data(t\_btree \*\*root, void \*item, int (\*cmpf)(void \*, void \*));

# Chapter VIII

Exercise 05: btree\_search\_item



Exercise 05

btree\_search\_item

提出するディレクトリ: *ex*05/

提出するファイル: btree\_search\_item.c, ft\_btree.h

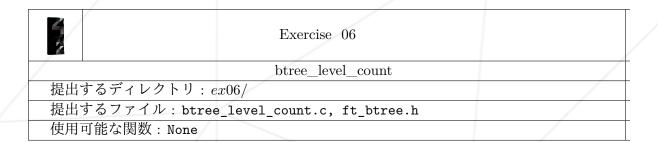
使用可能な関数: None

- 引数として与えられた参照データに対応する、最初の要素を返す関数btree\_search\_itemを作成せよ。ツリーは infix traversalを使用して参照すること。要素が見つからない場合は、NULLを返すこと。
- プロトタイプ例)

void \*btree\_search\_item(t\_btree \*root, void \*data\_ref, int (\*cmpf)(void \*, void \*));

# Chapter IX

Exercise 06: btree\_level\_count

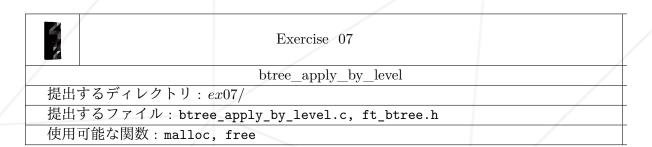


- 引数として渡された二分木の深さの最大値を返す関数btree\_level\_countを作成せよ。
- プロトタイプ例)

int btree\_level\_count(t\_btree \*root);

# Chapter X

Exercise 07: btree\_apply\_by\_level



- 引数として渡された関数を、ツリーの各ノードに適用する関数btree\_apply\_by\_levelを作成せよ。ツリーはレベルごとに参照すること。呼び出される関数は、3つの引数を取る。
  - void \*型の1つ目の引数は、ノードのアイテムである。
  - o int型の2つ目の引数は、検出された際のレベル(ルートである場合は0、子である場合は1 など)を表す。
  - int型の3つ目の引数は、レベルの最初のnodeである場合は1を表し、それ 以外の場合は0を表す。
- プロトタイプ例)

void btree\_apply\_by\_level(t\_btree \*root, void (\*applyf)(void \*item, int current\_level, int is\_first