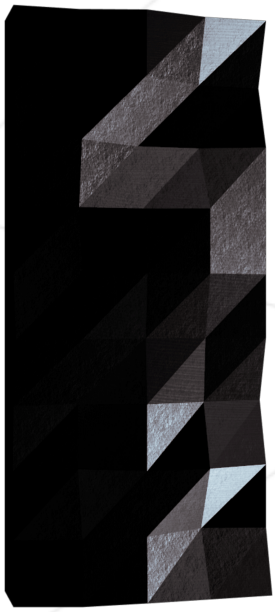

C Piscine

C 13

Summary: This document is the subject for the module C 13 of the C Piscine @ 42.

Contents

Chapter I

Instructions

- 오직 이 페이지만 참고해야 합니다. 소문은 믿지 마세요.
- 파일 제출 전에 이 문서가 변경될 수도 있으니 주의하세요!
- 파일과 디렉토리에 대해 적절한 권한을 갖고 있는지 확인하세요.
- 모든 과제물을 제출할 때는 제출 절차를 따라야 합니다.
- 제출하신 과제물은 동료들끼리 서로 확인하고 평가하게 됩니다.
- 추가로, Moulinette라는 프로그램도 과제물을 확인하고 평가합니다.
- Moulinette는 아주 꼼꼼하고 깐깐하게 과제물을 평가합니다. 완전히 자동화된 프로그램이기때문에 일체의 협상은 불가능합니다. 그러니 좋지 않은 평가를 받고 실망하고 싶지 않다면 최선을 다해 철저하게 과제를 수행하세요.
- Moulinette는 그다지 마음이 너그럽지 못하답니다. 표준을 따르지 않는 코드는 이해하려고 노력조차 하지 않을 겁니다. Moulinette은 `norminette` 는 프로그램으로 파일이 표준을 따랐는지 확인합니다. 그러니까 `norminette` 검사를 통과하지 못하는 과제물을 제출한다는 건 어리석은 일이겠죠?
- Exercise는 난이도에 따라 쉬운 문제에서 어려운 문제 순으로 짜여 있습니다. 앞 단계 문제의 과제물이 완벽하게 작동하지 않으면 난이도가 더 높은 문제는 아무리 잘 완료했다 하더라도 평가에 반영되지않습니다.
- 사용이 금지된 함수를 사용하는 것은 부정 행위로 간주됩니다. 부정 행위는 -42점을 받게 되며, 받은 점수는 절대 조정이 불가능합니다.
- 프로그램을 제출해야 하는 문제의 경우 main() 함수만 제출하면 됩니다.
- Moulinette은 -Wall -Wextra -Werror 플래그를 지정하여 컴파일하며 `gcc`를 사용합니다.
- 프로그램이 컴파일되지 않으면 0점을 받게 됩니다.
- Exercise에서 정한 파일 이외의 어떠한 파일도 디렉토리에 남겨 두어서는 안 됩니다.
- 질문이 있으신가요? 오른쪽 동료에게 물어보세요. 아니면 왼쪽 동료에게 물어보세요.
- 참고 가이드는 `Google / man / the Internet / ...`입니다.

- 인트라넷의 포럼에서 'C Piscine' 파트를 참조하거나 Slack의 Piscine 채널을 확인해 보세요.
- 예시를 꼼꼼히 살펴보세요. Exercise에서 명시적으로 언급되지 않은 세부적인 사항에 대한 힌트를 얻을 수도 있습니다...
- 오딘의 힘으로, 토르의 힘으로! 열심히 고민해 보세요!!!
- 이번 과제에서는 다음 struct를 사용하게 됩니다. :

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

- 이 struct를 `ft_btree.h` 파일에 넣은 다음 각 과제물을 제출할 때 함께 제출해야합니다.
- Exercise 01 부터는 제공된 btree_create_node를 사용하게 되므로 준비하세요. (ft_btree.h 파일에 프로토타입을 넣어 두면 편할 수도 있습니다...).

Chapter II

Foreword

다음은 헤비메탈 밴드 Venom의 앨범목록입니다 :

- 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)

Venom의 노래를 들으신다면 오늘 과제가 좀 쉬워 보일 겁니다.

Chapter III

Exercise 00 : btree_create_node

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

- 새로운 요소를 할당하는 함수 `btree_create_node`를 작성하세요. 이 함수는 `item`을 인자의 값으로 초기화하고 다른 모든 멤버는 0으로 초기화해야 합니다.
- 생선된 노드의 주소를 반환합니다.
- 프로토타입은 다음과 같이 선언합니다 :

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

Chapter IV

Exercise 01 : btree_apply_prefix

	Exercise 01
btree_apply_prefix	
Turn-in directory : *ex*01/	
Files to turn in : `btree_apply_prefix.c, ft_btree.h`	
Allowed functions : `None`	

- 인자로 주어진 함수를 각 노드의 `item`에 적용하는 함수 `btree_apply_prefix`를 작성하세요. 이때 전위 순회를 사용하여 트리를 검색합니다.
- 프로토타입은 다음과 같이 선언합니다. :

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

Chapter V

Exercise 02 : btree_apply_infix

	Exercise 02
btree_apply_infix	
Turn-in directory : *ex*02/	
Files to turn in : `btree_apply_infix.c, ft_btree.h`	
Allowed functions : `None`	

- 인자로 주어진 함수를 각 노드의 `item`에 적용하는 함수 `btree_apply_infix`를 작성하세요. 이때 **중위 순회**를 사용하여 트리를 검색합니다.
- 프로토타입은 다음과 같이 선언합니다. :

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

Chapter VI

Exercise 03 : btree_apply_suffix

	Exercise 03
btree_apply_suffix	
Turn-in directory : *ex*03/	
Files to turn in : `btree_apply_suffix.c, ft_btree.h`	
Allowed functions : `None`	

- 인자로 주어진 함수를 각 노드의 `item`에 적용하는 함수 `btree_apply_suffix`를 작성하세요. 이때 후위 순회를 사용하여 트리를 검색합니다.
- 프로토타입은 다음과 같이 선언합니다. :

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

Chapter VII

Exercise 04 : btree_insert_data

	Exercise 04
btree_insert_data	
Turn-in directory : *ex*04/	
Files to turn in : `btree_insert_data.c, ft_btree.h`	
Allowed functions : `btree_create_node`	

- `item`을 트리에 삽입하는 함수 `btree_insert_data`를 작성하세요. 인자로 전달되는 트리는 다음과 같이 정렬됩니다: 각 노드의 값보다 작은 요소는 왼쪽에 위치하고 높거나 같은 요소는 오른쪽에 위치합니다. 또한 `strcmp`와 비슷한 비교 함수가 인자로 전달됩니다.
- `root` 매개변수는 트리의 루트 노드를 가리킵니다. 최초 호출 시에는 `NULL`을 가리켜야 합니다. 회
- 프로토타입은 다음과 같이 선언합니다. :

```
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	
Turn-in directory : *ex*05/	
Files to turn in : `btree_search_item.c, ft_btree.h`	
Allowed functions : `None`	

- 인자로 주어진 참조 데이터와 관련 있는 첫 번째 요소를 반환하는 함수 `btree_search_item`를 작성하세요. 트리는 **전위 순회**를 사용하여 탐색되여야 합니다. 요소를 찾을 수 없다면 함수는 `NULL`을 반환해야 합니다.
- 프로토타입은 다음과 같이 선언합니다. :

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

Chapter IX

Exercise 06 : btree_level_count

	Exercise 06
btree_level_count	
Turn-in directory : *ex*06/	
Files to turn in : `btree_level_count.c, ft_btree.h`	
Allowed functions : `None`	

- 인자로 전달된 트리의 가장 큰 브랜치의 크기를 반환하는 함수 `btree_level_count`를 작성하세요.
- 프로토타입은 다음과 같이 선언합니다. :

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

Chapter X

Exercise 07 : btree_apply_by_level

	Exercise 07
btree_apply_by_level	
Turn-in directory : *ex*07/	
Files to turn in : `btree_apply_by_level.c, ft_btree.h`	
Allowed functions : `malloc, free`	

- 인자로 전달되는 함수를 트리의 각 노드에 적용하는 함수 `btree_apply_by_level`을 작성하세요. 트리는 레벨에서 레벨로 탐색되어야 합니다. 호출되는 함수는 다음 세 개의 인자를 취합니다.
  - 첫 번째 인자, `void *` 형은 노드의 item에 해당합니다;
  - 두 번째 인자, `int` 형은 지금 검색 중인 레벨에 해당합니다: 루트는 0, 자식은 1, 손자는 ....
  - 세 번째 인자, `int` 형은 레벨의 첫 번째 노드일 경우에는 1, 그 이외의 경우는 0이 됩니다.
- 프로토타입은 다음과 같이 선언합니다. :

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