



C Piscine

C 13

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

Version: 4.2

Contents

I	Instructions	2
II	Foreword	4
III	Egzersiz 00 : btree_create_node	5
IV	Egzersiz 01 : btree_apply_prefix	6
V	Egzersiz 02 : btree_apply_infix	7
VI	Egzersiz 03 : btree_apply_suffix	8
VII	Egzersiz 04 : btree_insert_data	9
VIII	Egzersiz 05 : btree_search_item	10
IX	Exercise 06 : btree_level_count	11
X	Exercise 07 : btree_apply_by_level	12
XI	Submission and peer-evaluation	13

Chapter I

Instructions

- Lütfen sadece bu sayfayı referans alınız: söylentilere kulak asmayınız.
- Dikkat! Dokümanın gönderim öncesinde değişme ihtimali vardır.
- Lütfen dosyalarınız ve dizileriniz için gerekli yetkilere sahip olduğunuzdan emin olunuz.
- Bütün çalışmalarınız için gönderim talimatlarını takip ediniz.
- Çalışmalarınız sınıf arkadaşlarınız tarafından kontrol edilip notlandırılacaktır.
- Aynı zamanda, çalışmalarınız Moulinette adlı program tarafından da kontrol edilip notlandırılacaktır.
- Moulinette değerlendirmelerinde çok titiz ve katıdır. Otomatik bir program olmasından dolayı görüş alışverişi mümkün değildir. Sürpriz bir sonuçla karşılaşmak için çalışmalarınızı dikkatlice yapınız.
- Moulinette çok açık görüşlü değildir. Kodunuz Norm'a uymadığı takdirde onu anlamaya çalışmayacaktır. Moulinette dosyalarınızın norm'a uyup uymadığını kontrol etmek için **norminette** adında bir program kullanmaktadır. TL;DR: **norminette**'in kontrolünden geçemeyecek bir dosya teslim etmek akılsızca olacaktır.
- Çalışmalar en kolaydan en zora olacak şekilde zorluklarına göre sıralanmıştır. Daha zor bir çalışma başarıyla tamamlanmış bile olsa daha kolay bir çalışmanın tamamıyla fonksiyonel olmaması durumunda dikkate **alınmayacaktır**.
- Yasaklanmış bir fonksiyon kullanmak hile olarak görülmektedir. Bunu yapan kişiler -42 puan alacaktır, ve bu not pazarlığa tabi değildir.
- Sizden program istersek sadece bir main() fonksiyonu göndermeniz gerekir.
- Moulinette çalışmaları şu şekilde sınıflandırır: -Wall -Wextra -Werror ve gcc
- Eğer programınız sınıflandırılmazsa, 0 alırsınız.
- Dizinizde konunun başlığındakiler dışında hiçbir dosya bırakmayınız.
- Bir sorunuz mu var? Sağınızdaki arkadaşınıza sorun. Olmadı solunuzdakine...

- Başvuru kılavuzunuzun adı `Google / man / the Internet / ...` 'dır.
- Intranetteki forumun "C Piscine" kısmını ya da Slack'deki Piscine bölümünü kontrol edin.
- Konu içerisinde net bir şekilde belirtilmemiş detayları anlayabilmek için örnekleri dikkatlice inceleyiniz.
- Odin ve Thor adına ! Kafayı çalıştırın !!!
- Bu egzesiz için aşağıdaki structure kullanılacaktır :

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

- Bu yapıyı bir `ft_btree.h` dosyasına koyup, her çalışma için yollamanız gerekmektedir.
- Çalışma 01'den itibaren `ft_create_elem`'imizi kullanacağız, buna göre önleminizi alın. (Prototipini `ft_btree.h` dosyasında bulundurmak işe yarayabilir...).

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

Egzersiz 00 : btree_create_node


	Exercise 00
btree_create_node	
Turn-in directory : <i>ex00/</i>	
Files to turn in : btree_create_node.c , ft_btree.h	
Allowed functions : malloc	

- Yeni bir eleman için yeri ayrılan **btree_create_node** fonksiyonu oluşturun. Fonksiyon **item** değişkenine verilen argümanı ve diğer değişkenleri ise 0'a eşitler.
- Yaratılan elemanın adresi döndürülür.
- Prototipi şu şekilde olmalıdır :

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

Chapter IV

Egzersiz 01 : btree_apply_prefix


	Exercise 01
btree_apply_prefix	
Turn-in directory : <i>ex01/</i>	
Files to turn in : btree_apply_prefix.c , ft_btree.h	
Allowed functions : None	

- Argüman olarak verilen fonksiyonu bütün nodeların **item** değişkenine uygulayan **btree_apply_prefix** fonksiyonu oluşturun. Tree'de gezerken **prefix traversal** kullanılmalı.
- Prototipi şu şekilde olmalıdır :

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

Chapter V

Egzersiz 02 : btree_apply_infix


	Exercise 02
btree_apply_infix	
Turn-in directory : <i>ex02/</i>	
Files to turn in : btree_apply_infix.c , ft_btree.h	
Allowed functions : None	

- Argüman olarak verilen fonksiyonu bütün nodeların **item** değişkenine uygulayan **btree_apply_infix** fonksiyonu oluşturun. Tree'de gezerken **infix traversal** kullanılmalı.
- Prototipi şu şekilde olmalıdır :

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


Chapter VI

Egzersiz 03 : btree_apply_suffix


	Exercise 03
btree_apply_suffix	
Turn-in directory : <i>ex03/</i>	
Files to turn in : btree_apply_suffix.c , ft_btree.h	
Allowed functions : None	

- Argüman olarak verilen fonksiyonu bütün nodeların **item** değişkenine uygulayan **btree_apply_suffix** fonksiyonu oluşturun. Tree'de gezerken **suffix traversal** kullanılmalı.
- Prototipi şu şekilde olmalıdır :

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

Chapter VII

Egzersiz 04 : btree_insert_data


	Exercise 04
btree_insert_data	
Turn-in directory : <i>ex04/</i>	
Files to turn in : btree_insert_data.c , ft_btree.h	
Allowed functions : btree_create_node	

- Argüman olarak verilen **item** kullanarak yeni bir node oluşturup oluşturulan node u argüman olarak verilen tree'ye yerleştiren **btree_insert_data** fonksiyonu oluşturun. Argüman olarak verilen tree sıralanmıştır. Herbir **node** sol tarafında kendisinden küçük sağ tarafında ise kendisinden büyük veya eşit bir değer olacak şekilde. Argüman olarak nodeları aralarında karşılaştırmamız için **strcmp** gibi bir fonksiyon verilecek
- The **root** parametresi treenin kökünü işaret eder. İlk çağırılışında **NULL**'a eşittir.
- Prototipi şu şekilde olmalıdır :

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

Chapter VIII

Egzersiz 05 : btree_search_item


	Exercise 05
btree_search_item	
Turn-in directory : <i>ex05/</i>	
Files to turn in : btree_search_item.c , ft_btree.h	
Allowed functions : None	

- Treenin içinde **infix traversal** yöntemi ile gezerek data referansı ile aynı olan node u döndüren **btree_search_item** fonksiyonu oluşturun. Eğer data referansı tree'de bulunmuyorsa fonksiyon NULL döndürmelidir.
- Prototipi şu şekilde olmalıdır :

```
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 : <i>ex06/</i>	
Files to turn in : btree_level_count.c, ft_btree.h	
Allowed functions : None	

- Argüman olarak verilen treenin en büyük dalının boyutunu döndüren **btree_level_count** fonksiyonu oluşturun.
- Prototipi şu şekilde olmalıdır :

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

Chapter X

Exercise 07 : btree_apply_by_level

	Exercise 07
btree_apply_by_level	
Turn-in directory : <i>ex07/</i>	
Files to turn in : btree_apply_by_level.c , ft_btree.h	
Allowed functions : malloc , free	

- Argüman olarak verilen fonksiyonu treenin bütün nodelarına uygulayan **btree_apply_by_level** fonksiyonu oluşturun. Tree seviye seviye gezilmelidir. Fonksiyon üç argüman alır.
 - İlk argüman, **void *** tipinde, node'un değeridir.
 - İkinci argüman, **int** tipinde, bulunan seviyedir: 0 kök, 1 çocuk, 2 torun, vb. ;
 - Üçüncü argüman, **int** tipinde, seviyenin ilk node u ise 1, değil ise 0.
- Prototipi şu şekilde olmalıdır :

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

Chapter XI

Submission and peer-evaluation

Egzersizlerinizi yollarken Git deponuzu kullanıcaksınız her zamanki gibi. Savunmada sadece deponuzun içindekiler değerlendirilicektir. Dosya ve klasör isimlerini bir daha kontrol etmekten çekinmeyin. Doğru olduklarına emin olun.



Yalnızca talep edilen dosyaları teslim etmeniz gerekir.