C Piscine

C 05

Summary: 이 문서는 C Piscine@42의 C 05 모듈용 과제입니다.

Version: 6.2

Contents

I	Instructions	2
II	Foreword	4
III	Exercise 00 : ft_iterative_factorial	6
IV	Exercise 01 : ft_recursive_factorial	7
V	Exercise 02 : ft_iterative_power	8
VI	Exercise 03 : ft_recursive_power	9
VII	Exercise 04 : ft_fibonacci	10
VIII	Exercise 05 : ft_sqrt	11
IX	Exercise 06 : ft_is_prime	12
X	Exercise 07 : ft_find_next_prime	13
XI	Exercise 08 : The Ten Queens	14
XII	제출 및 동료 평가	15

Chapter I

Instructions

- 오직 이 문서만 참고해야 합니다. 소문은 믿지 마세요.
- 파일 제출 직전까지도 이 문서가 변경될 수도 있으니 주의하세요!
- Exercise(이하 연습문제)들은 쉬운 것부터 어려운 것까지 난이도순으로 나열되어 있습니다. 만약 앞의 과제에 대한 답이 정확하지 않다면 뒤의 연습문제에 대한 답이 아무리 완벽하다 하더라도 신경조차 쓰지 마세요.
- 파일과 디렉토리에 대한 적절한 권한이 부여되어 있는지 확인하세요.
- 모든 연습문제의 제출은 제출 절차에 따라 진행되어야 합니다.
- 제출한 과제물은 동료들끼리 서로 확인하고 평가하게 됩니다.
- 추가로, Moulinette라는 프로그램도 과제물을 확인하고 평가합니다.
- Moulinette은 매우 세밀한 평가를 실시합니다. Moulinette은 완전히 자동화되어 있고 당신의 협상 요청에 응하지 않을 것입니다. Moulinette에게 마음의 상처를 받고 싶지 않다면 과제를 철저하게 진행하세요.
- Moulinette에게 타협이란 없습니다. 만약 당신의 코드가 Norm을 준수하지 않는다면 Moulinette은 컴파일조차 진행하지 않을 것 입니다. Moulinette은 `norminette`라는 프로그램을 활용하여 제출물의 Norm 준수 여부를 판단합니다. TL;DR: `norminette`의 검사조차 통과하지 못한 작업물을 제출하는 것은 어리석은 일입니다.
- 사용금지된 함수(또는 기능)을 사용하는 것은 부정 행위로 간주됩니다. 부정 행위자는 `-42`점을 부여 받으며 이 점수는 임의로 변경하지 못합니다.
- program을 요청하는 경우에만 main() 함수를 포함시키세요.
- Moulinette은 다음 명령어와 플래그로 컴파일을 합니다: `cc` -Wall -Wextra -Werror
- 만약 당신의 프로그램이 컴파일되지 않는다면 `0`점을 부여받습니다.
- 연습문제에 명시된 파일 이외에는 그 어떠한 파일도 남겨두어서는 안 됩니다.
- 질문이 있나요? 오른쪽에 있는 동료에게 물어보세요. 없다구요? 그렇다면 왼쪽에 있는 동료에게 물어보세요.
- 우리는 참고 자료의 이름을 `Google / man page / 인터넷 / ...`이라고 부르기로 했어요. 단, `42` 교육철학은 지켜주셔야 해요.

- Check out the "C Piscine" part of the forum on the intranet, or the slack Piscine.
- 예제를 꼼꼼히 보세요. 과제에서 명시적으로 언급하지 않은 세부 요구사항을 포함하고 있을지도..
- By Odin, by Thor ! Use your brain !!!

`Norminette`는 *`-R CheckForbiddenSourceHeader`* 플래그와 함께 실행되어야 합니다. `Moulinette`도 `Norminette`을 이렇게 사용할 것입니다.

Chapter II

Foreword

해리 포터에는 다음과 같은 노래가 나옵니다. :

```
Oh you may not think me pretty,
But don't judge on what you see,
I'll eat myself if you can find
A smarter hat than me.

You can keep your bowlers black,
Your top hats sleek and tall,
For I'm the Hogwarts Sorting Hat
And I can cap them all.

The Sorting Hat, stored in the Headmaster's Office.
There's nothing hidden in your head
The Sorting Hat can't see,
So try me on and I will tell you
Where you ought to be.

You might belong in Gryffindor,
Where dwell the brave at heart,
Their daring, nerve, and chivalry
Set Gryffindors apart;

You might belong in Hufflepuff,
Where they are just and loyal,
Those patient Hufflepuffs are true
And unafraid of toil;

Or yet in wise old Ravenclaw,
If you've a ready mind,
Where those of wit and learning,
Will always find their kind;

Or perhaps in Slytherin
You'll make your real friends,
Those cunning folks use any means
```

```
To achieve their ends.

So put me on! Don't be afraid!
And don't get in a flap!
You're in safe hands (though I have none)
For I'm a Thinking Cap!
```

안타깝지만 이번 과제는 해리 포터와는 아무 관련이 없습니다. 연습 문제는 마법으로 풀 수는 없기 때문이죠. 아쉬운 일이네요.

Chapter III

Exercise 00 : ft_iterative_factorial

	Exercise 00
	ft_iterative_factorial
Turn-in directory : *ex*00/	
Files to turn in : `ft_iterative_factorial.c`	
Allowed functions : `None`	

- 숫자를 반환하는 반복 함수(iterated function)를 만드세요. 반환값은 인수로 주어진 숫자가 밑수(base)인 계승(factorial)의 결과입니다.
- 인수가 올바르지 않으면 함수는 0을 반환해야 합니다.
- 오버플로우는 고려하지 마세요. 함수 반환은 정의되지 않습니다(undefined).
- 프로토타입은 다음과 같이 선언합니다. :

```
int ft_iterative_factorial(int nb);
```

`undefined behavior`

Chapter IV

Exercise 01 : ft_recursive_factorial

	Exercise 01
ft_recursive_factorial	
Turn-in directory : $ex01/$	
Files to turn in : `ft_recursive_factorial.c`	
Allowed functions : `None`	

- 매개 변수로 지정된 수의 계수를 반환하는 재귀 함수(recursive function)를 만드세요.
- 인수가 올바르지 않으면 함수는 0을 반환해야 합니다.
- 오버플로우는 고려하지 마세요. 함수 반환은 정의되지 않습니다.
- 프로토타입은 다음과 같이 선언합니다. :

```
int ft_recursive_factorial(int nb);
```

Chapter V

Exercise 02 : ft_iterative_power

	Exercise 02
ft_iterative_power	
Turn-in directory : $ex02/$	
Files to turn in : `ft_iterative_power.c`	
Allowed functions : `None`	

- nb의 거듭제곱 값을 반환하는 반복 함수를 만드세요. power가 0보다 작을 경우 0을 반환합니다. 역시 오버플로는 처리되지 않아야 합니다.
- 0의 0제곱은 1을 반환하기로 합시다.
- 프로토타입은 다음과 같이 선언합니다. :

```
int ft_iterative_power(int nb, int power);
```

Chapter VI

Exercise 03 : ft_recursive_power

	Exercise 03
ft_recursive_power	
Turn-in directory : $ex03/$	
Files to turn in : `ft_recursive_power.c`	
Allowed functions : `None`	

- nb의 거듭제곱 값을 반환하는 재귀 함수를 만드세요.
- 오버플로우는 고려하지 마세요. 함수 반환은 정의되지 않습니다.
- 0의 0제곱은 1을 반환하기로 합시다.
- 프로토타입은 다음과 같이 선언합니다. :

```
int ft_recursive_power(int nb, int power);
```

Chapter VII

Exercise 04 : ft_fibonacci

	Exercise 04
	ft_fibonacci
Turn-in directory : *ex*04/	
Files to turn in : `ft_fibonacci.c`	
Allowed functions : `None`	

- 피보나치 수열의 `n`번째 항을 반환하는 함수 `ft_fibonacci`를 만드세요. 이때 첫번째 항을 0번 인덱스(index)로 간주합니다.
  우리의 피보나치 수열은 이렇게 시작합니다. : 0, 1, 1, 2
- 오버플로우는 고려하지 마세요. 함수 반환은 정의되지 않습니다.
- 프로토타입은 다음과 같이 선언합니다. :

```
int ft_fibonacci(int index);
```

- `ft_fibonacci`가 재귀 함수인 것을 제가 두눈으로 똑똑히 봤습니다.
- `index`가 0보다 작을 경우 -1을 반환합니다.

Chapter VIII

Exercise 05 : ft_sqrt

	Exercise 05
	ft_sqrt
Turn-in directory : $ex05/$	
Files to turn in : `ft_sqrt.c`	
Allowed functions : `None`	

- 어떤 수의 제곱근이 존재하면 그 제곱근을, 제곱근이 무리수면 0을 반환하는 함수를 만드세요.
- 프로토타입은 다음과 같이 선언합니다. :

```
int ft_sqrt(int nb);
```

Chapter IX

Exercise 06 : ft_is_prime

	Exercise 06
	ft_is_prime
Turn-in directory : *ex06/*	
Files to turn in : `ft_is_prime.c`	
Allowed functions : `None`	

- 매개변수로 주어진 수가 소수이면 1을, 그렇지 않으면 0을 반환하는 함수를 만드세요.
- 프로토타입은 다음과 같이 선언합니다. :

```
int ft_is_prime(int nb);
```

0과 1은 소수가 아닙니다.

Chapter X

Exercise 07 : ft_find_next_prime

	Exercise 07
ft_find_next_prime	
Turn-in directory : $ex07/$	
Files to turn in : `ft_find_next_prime.c`	
Allowed functions : `None`	

- 매개변수로 주어진 숫자보다 크거나 같은 다음 소수를 반환하는 함수를 만드세요.
- 프로토타입은 다음과 같이 선언합니다. :

```
int ft_find_next_prime(int nb);
```

Chapter XI

Exercise 08 : The Ten Queens

	Exercise 08
The Ten Queens	
Turn-in directory : $ex08/$	
Files to turn in : `ft_ten_queens_puzzle.c`	
Allowed functions : `write`	

- 10X10 체스판에서 퀸 10개가 위치할 수 있는 모든 경우의 수를 출력하는 함수를 만드세요. 단, 모든 퀸은 1회의 움직임으로 다른 퀸을 잡을 수 없어야 합니다.
- 이 문제를 해결하기 위해서는 재귀를 활용해야 할 것입니다.
- 프로토타입은 다음과 같이 선언합니다. :

```
int ft_ten_queens_puzzle(void);
```

- 다음과 같이 출력되어야 합니다. :

```
$>./a.out | cat -e
0257948136$
0258693147$
...
4605713829$
4609582731$
...
9742051863$
$>
```

- 순서는 왼쪽에서 오른쪽입니다. 첫 번째 숫자는 첫 번째 열(index는 0부터 시작)에서의 첫 번째 퀸의 위치를 나타냅니다. N번째 자릿수는 N번째 열에서 N번째 퀸의 위치를 나타냅니다.
- 반환값은 출력된 모든 방법의 총 개수여야 합니다.

Chapter XII

제출 및 동료 평가

평소처럼 `git` 저장소에 과제를 제출하세요. 디펜스 중에는 저장소 내부의 작업만 평가됩니다. 파일 이름이 올바른지 다시 한 번 확인하는 작업을 주저하지 마세요.

이 프로젝트의 문서에서 요구한 파일만 반환해야 합니다.