


Recursion in PHP



주영익 (사람인HR)

PHPSTORM BLOG

JETBRAINS.COMPHPSTORMFORUMEAPISSUE TRACKER


← PhpStorm 2017.1 EAP 171.2152VCS in Depth for PhpStorm →

PHP Annotated Monthly – January 2017

Posted on January 9, 2017 by Gary Hockin

Happy New Year! New year, new PHP Annotated Monthly, a roundup of all that's happened in the last month or so with [Gary Hockin](#), Developer Advocate for PhpStorm at JetBrains.

Don't forget: you can now get **PHP Annotated Monthly delivered to your inbox**, so you'll never miss the monthly roundup again. [Sign up here](#) and get next month's PHP Annotated Monthly delivered right to you.




PHP and Development

The latest versions of PHP are:


- 7.1.0
- 7.0.14

<https://blog.jetbrains.com/phpstorm/2017/01/php-annotated-monthly-january-2017/>

PHP can't jump? Thing about recursion.



Hubert
October 18, 2016



Let's get straight into the problem – assume we want to calculate nth Fibonacci number. Definition : $F(n) = F(n-1) + F(n-2)$ with seed values $F(1) = F(2) = 1$ so the most intuitive way to do this is just from the definition (recursive)

```
<?php
function fibonacci($n){
    if($n == 1 || $n == 2){
        return 1;
    }
    return fibonacci($n - 1) + fibonacci($n - 2);
}

echo fibonacci(7).PHP_EOL;
```

Search ...

RECENT POSTS

- PHP can't jump? Thing about recursion.
- Creating patient feedback system
- Commands and loops
- Approach at diet recommendations
- Finally some code

RECENT COMMENTS

ARCHIVES

- October 2016
- May 2016
- April 2016

<http://brylkowski.com/php-cant-jump-thing-about-recursion/>

피보나치 수(Fibonacci Numbers)

피보나치 수

위키백과, 우리 모두의 백과사전.

피보나치 수(Fibonacci Numbers)는 수학에서 아래의 점화식으로 정의되는 수열이다.

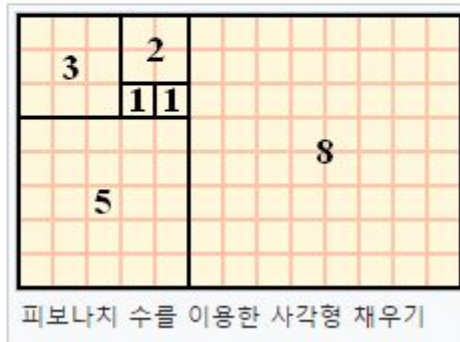
$$F_n := \begin{cases} 0 & \text{if } n = 0; \\ 1 & \text{if } n = 1; \\ F_{n-1} + F_{n-2} & \text{if } n > 1. \end{cases}$$

피보나치 수는 0과 1로 시작하며, 다음 피보나치 수는 바로 앞의 두 피보나치 수의 합이 된다.

$n = 0, 1, \dots$ 에 해당하는 피보나치 수는 (OEIS의 수열 A000045)

1, 1, 2, 3, 5, 8, 13, 21, 34, 55, 89, 144, 233, 377, 610, 987, 1597, 2584, 4181, 6765, 10946...

이다.



피보나치 수를 이용한 사각형 채우기

피보나치 수(Fibonacci Numbers)

$$\begin{aligned}s(x) &= \sum_{k=0}^{\infty} F_k x^k \\&= F_0 + F_1 x + \sum_{k=2}^{\infty} (F_{k-1} + F_{k-2}) x^k \\&= x + \sum_{k=2}^{\infty} F_{k-1} x^k + \sum_{k=2}^{\infty} F_{k-2} x^k \\&= x + x \sum_{k=0}^{\infty} F_k x^k + x^2 \sum_{k=0}^{\infty} F_k x^k \\&= x + x s(x) + x^2 s(x).\end{aligned}$$



피보나치 수(Fibonacci Numbers)

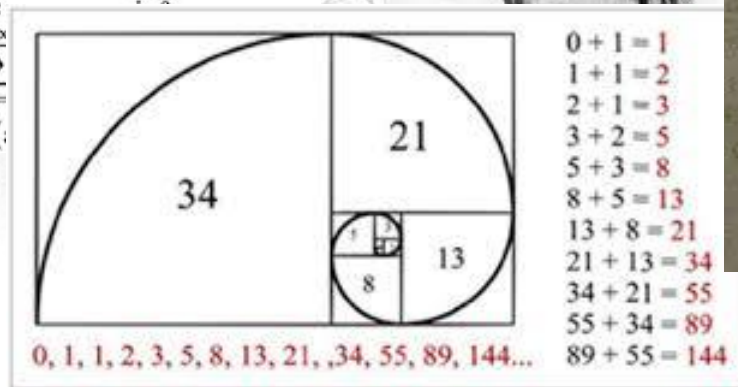
$$s(x) = \sum_{k=0}^{\infty} F_k x^k$$

$$= F_0 + F_1 x + \sum_{k=2}^{\infty} (F_{k-1} + F_{k-2}) x^k$$

$$= x + \sum_{k=2}^{\infty} F_{k-1} x^k + \sum_{k=2}^{\infty} F_{k-2} x^k$$

$$= x + x \sum_{k=1}^{\infty} F_k x^{k-1}$$

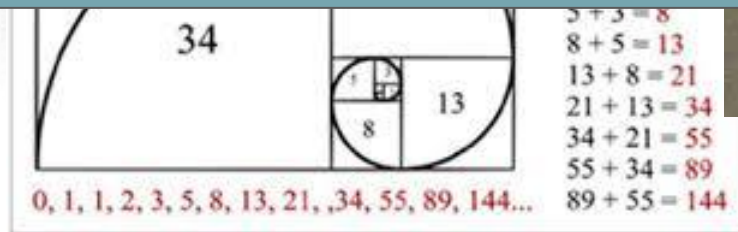
$$= x + x s(x)$$



피보나치 수(Fibonacci Numbers)

$$s(x) = \sum_{k=0}^{\infty} F_k x^k$$
$$= F_0 + F_1 x + \sum_{k=2}^{\infty} (F_{k-1} + F_{k-2}) x^k$$

이게 중요한 게 아님



발단 > 전개 > 위기 > 절망 > 결말

```
<?php
```

```
function fibonacci($nthNumber)
```

```
{
```

```
    // 코드를 짜 봅시다
```

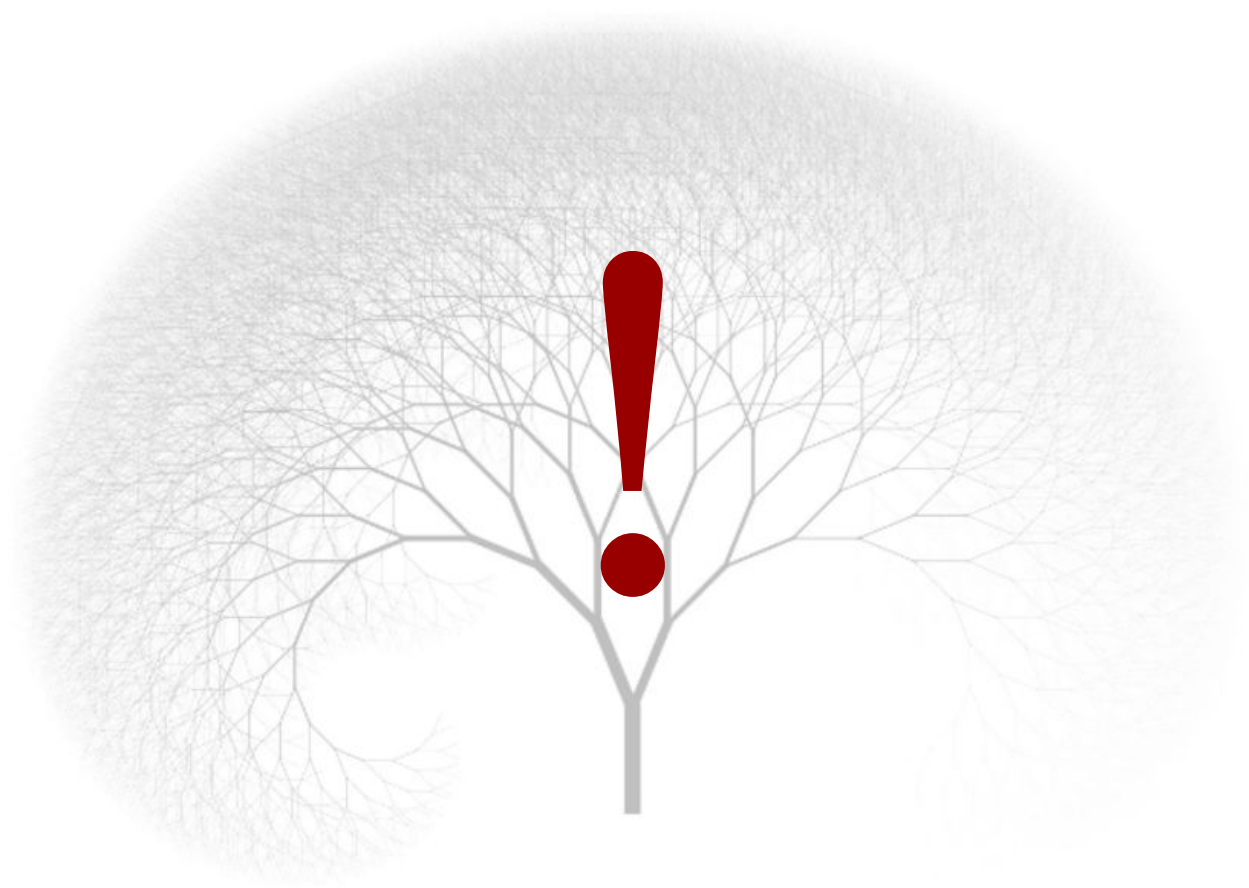
```
}
```

```
echo fibonacci(10) . PHP_EOL;
```

Babo fibonacci

```
function fibonacci($n)
{
    if ($n == 1 || $n == 2) {
        return 1;
    }
    return fibonacci($n - 1) + fibonacci($n - 2);
}

echo fibonacci(34) . PHP_EOL;
```

Better fibonacci

```
function fibonacci($iterationsLeft, $prePrevious = 0, $previous = 1)
{
    if ($iterationsLeft == 0) {
        return $previous;
    }
    return fibonacci(
        $iterationsLeft - 1,
        $previous,
        bcadd($prePrevious, $previous)
    );
}


echo fibonacci(250) . PHP_EOL;
```

Better fibonacci

```
function fibonacci($iterationsLeft, $prePrevious = 0, $previous = 1)
{
    if ($iterationsLeft == 0) {
        return $previous;
    }
    return fibonacci(
        $iterationsLeft - 1,
        $previous,
        bcadd($prePrevious, $previous)
    );
}

echo fibonacci(12345) . PHP_EOL;
```

BUT



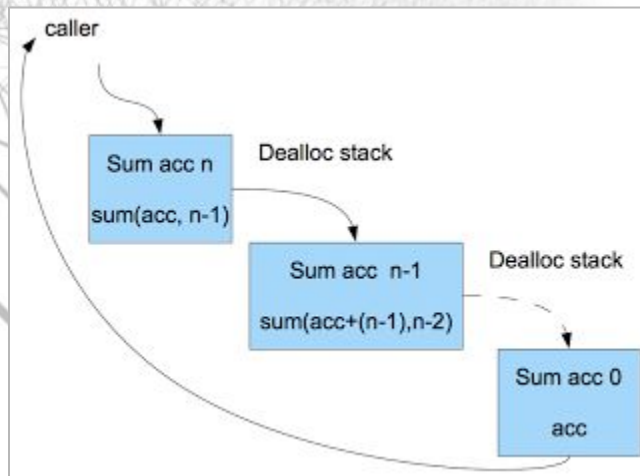
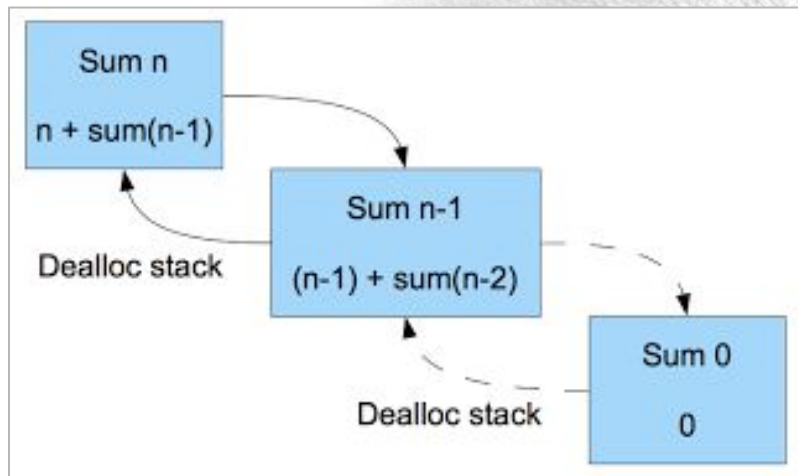
Fatal! Fatal!

Fatal error: Allowed memory size of 134217728 bytes exhausted ...

Fatal error: Maximum **function** nesting level of '100' reached, aborting! in ...

꼬리 재귀 최적화

TAIL CALL(RECURSION) OPTIMIZATION



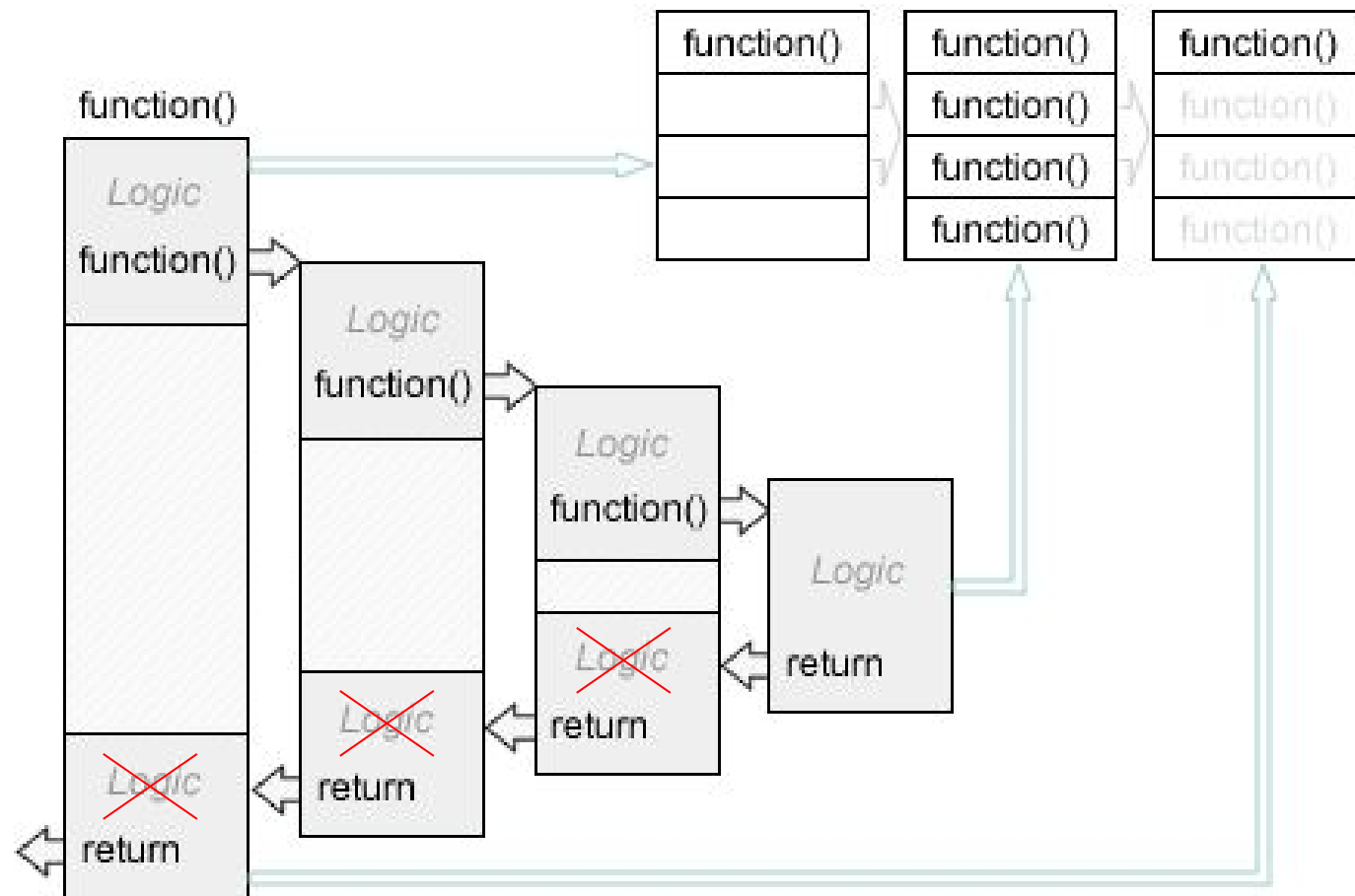
꼬리 재귀 최적화

TAIL CALL(RECURSION) OPTIMIZATION

- 꼬리 재귀 : 재귀 호출이 끝난 후 현재 함수에서 추가 연산을 요구하지 않도록 구현하는 것
 - 이때 컴파일러에 따라 스택을 효율적으로 관리해주는 언어‘도’ 있다
 - 꼬리재귀 최적화는 더 이상 필요없는 스택을 유지하지 않고 재사용함으로써 발생한다
-
- 결론부터 말하자면, **PHP에선 꼬리 재귀 최적화를 지원하지 않는다**

Recursive Call

Call Stack



```
function factorial_recursive($number) {  
    if ($number < 0) {  
        throw new InvalidArgumentException('Number cannot be less than zero');  
    }  
    if ($number == 0) {  
        return 1;  
    }  
    return $number * factorial_recursive($number - 1);  
}
```

```
function factorial ($number, $factorial = 1) {  
    if ($number < 0) {  
        throw new InvalidArgumentException('Number cannot be less than zero (0)');  
    }  
    if ($number == 0) {  
        return $factorial;  
    }  
    else {  
        return factorial($number - 1, $factorial * $number);  
    }  
}
```



```
function factorial_recursive($number) {  
    if ($number < 0) {  
        throw new InvalidArgumentException('Number cannot be less than zero');  
    }  
    if ($number == 0) {  
        return 1;  
    }  
    return $number * factorial_recursive($number - 1);  
}
```

다 부질없고...PHP는 똑같다는 거...

```
function factorial($number) {  
    if ($number == 0) {  
        return 1;  
    }  
    return $number * factorial($number - 1);  
}
```

Why does php not support tail call optimization?

Quora

Recursion Mathematical Optimization PHP (programming language)

Why does php not support tail call optimization?

With rise of interest in functional programming in dev community and tail calls being talked since almost 5 years(<http://bit.ly/2az5AFE>), it's strange that PHP does not have tail call optimization yet.

Tl;dr:

There's no technical reason that PHP couldn't implement tail recursion optimization at the moment, but the PHP core team is more concerned about security and interoperability than performance.

nothing. Opacache pre-compiles your code into intermediary bytecode format so that it can process faster.


This meant PHP implemented behavior like limiting how many times a function could call itself and throwing an error.

Since 5.5, there's no good technical reason to not have tail call optimization, particularly

<https://www.quora.com/Why-does-php-not-support-tail-call-optimization>

발단 > 전개 > 위기 > 절망 > 결말

재귀 함수를 최적화하기



Trampoline

발단 > 전개 > 위기 > 절망 > 결말

재귀 함수를 최적화하기

Trampoline

붕붕

방방, 방방이 : 수도권, 충청도 지방.

봉봉 : 대구, 경북 지방.

풍풍 : 인천과서울 일부지역, 부산, 울산, 경남 지방.

콩콩 : 경남 동남부, 전라도, 강원도 영동 지방.



```
function trampoline($function)
{
    $result = function () use ($function) {
        return $function();
    };
    while (is_callable($result)) {
        $result = $result();
    }
    return $result;
}

function fibonacci($n)
{
    return trampoline(
        function () use ($n) {
            return fibonacciInternal($n, 0, 1);
        }
    );
}

function fibonacciInternal($iterationsLeft, $prePrevious = 0, $previous = 1)
{
    if ($iterationsLeft == 0) {
        return $prePrevious;
    }
    return function () use ($iterationsLeft, $prePrevious, $previous) {
        return fibonacciInternal($iterationsLeft - 1, $previous, bcadd($prePrevious, $previous));
    };
}

echo fibonacci(12345) . PHP_EOL;
```



```
function trampoline($function)
{
    $result = $function();
    while (is_callable($result)) {
        $result = $result();
    }
    return $result;
}

function fibonacci($n)
{
    return trampoline(
        fibonacciInternal($n, 0, 1)
    );
}

function fibonacciInternal($iterationsLeft, $prePrevious = 0, $previous = 1)
{
    if ($iterationsLeft == 0) {
        return $prePrevious;
    }
    return function () use ($iterationsLeft, $prePrevious, $previous) {
        return fibonacciInternal($iterationsLeft - 1, $previous, bcadd($prePrevious, $previous));
    };
}

echo fibonacci(12345) . PHP_EOL;
```

실행 가능한 전달 인자를 받아
더 이상 실행할 수 없을 때까지 실행한 후 리턴

trampoline 함수에 실행 가능한 인수를 넘긴다

1. 자신을 호출하는 익명함수를 리턴
2. 최종 단계라면 마지막 값을 리턴

재귀 함수를 최적화하기 - 2. Loop

그냥 Loop나 돌려라

재귀 함수를 최적화하기 - 2. Loop



재귀문 : 조건을 만족할 때까지
함수를 호출하라

반복문 : 조건을 만족할 때까지
블럭 안의 코드를 실행하라

재귀 함수를 최적화하기 - 2. Loop

```
function tail_factorial($n, $acc = 1)
{
    if ($n == 1) {
        return $acc;
    }
    return tail_factorial($n-1, $n * $acc);
}

function unrolled_factorial($n)
{
    $acc = 1;
    while ($n > 1) {
        $acc *= $n--;
    }
    return $acc;
}
```

재귀문 : 조건을 만족할 때까지
함수를 호출하라

반복문 : 조건을 만족할 때까지
블럭 안의 코드를 실행하라

재귀 함수를 최적화하기 - 3. GOTO

```
function goto_factorial($n)
{
    $acc = 1;
    f:
    if ($n == 1) {
        return $acc;
    }
    $acc *= $n--;
    goto f;
}
```

3. GOTO문의 해로움

[편집]

GOTO Statement Considered Harmful ^[1] - Edsger W. Dijkstra

고급 언어에서 GOTO문은 많은 커뮤니티에서 터부시되는 구문이다. 그 이유는 이게 너무 많이 쓰이다 보면 **구조가 무너지고 코드가 황폐화되면서** 이게 어째 실행은 잘 되긴 되는데 코드를 짰 프로그래머 자신도 "이게 뭘미?"하는 소리가 절로 나오며 읽고 유지보수하기가 힘들어지는 지경에 이르게 된다. 이러한 상태로 된 코드를 '**스파게티 코드**'라고 한다. 이런 스파게티 스타일을 반대하고 구조적 프로그래밍을 주창한 **네덜란드**의 컴퓨터 과학자 **에츠허르 다익스트라**(Edsger Wybe Dijkstra)는 GOTO문의 사용이 프로그램의 정확성 분석과 증명 등의 측면에 해가 된다고 한 바 있다. 실제 프로그래밍할 때도 GOTO문을 남용하면 **디버깅 불가+기능 추가 난해+다른 사람의 욕 처먹기+내가 이러려고 이 코드를 짰나 자괴감 들고 괴로운** 사단 콤보가 일어날 수 있다.

<https://namu.wiki/w/GOTO>

ALL YOU NEED

IS

~~LOVE~~

PHP 5.3+

홍보

1. 사람인 사람 뽑습니다.

- **PHP** 웹 개발 신입/경력
- **JAVA** 개발 신입/경력
- 퍼블리셔
- UI 디자인

2. 개발 커뮤니티 지원

- 장소 : 사람인 시설 제공 가능(구로디지털단지역)
- 다과비/식대 등 : 비용 일정액을 지원 가능
- 기타 : 커뮤니티 진행 중 필요한 지원 사항



Q & A

References

- <https://blog.jetbrains.com/phpstorm/2017/01/php-annotated-monthly-january-2017/>
- <http://brylkowski.com/php-cant-jump-thing-about-recursion/>
- <http://jpauli.github.io/2016/09/16/php-7-magic-function-call-trampoline.html>
- <https://www.sitepoint.com/understanding-recursion/>
- <http://php.net/manual/kr/control-structures.goto.php>
- <http://php.net/manual/kr/functions.anonymous.php>
- <https://www.quora.com/Why-does-php-not-support-tail-call-optimization>

