

Dash – Problem solving_00

Summary: this document is the subject for the dash @ 42Seoul.

Contents

1 Foreword

2 Instructions

3 Exercise 00 : forward_print

4 Exercise 01 : backward_print

5 Exercise 02 : dynamic_programming

Chapter 1

Foreword

이 프로젝트는 문제를 해결하는 것에 초점을 맞추었으며, 문제를 다양하게 바라보는 관점을 기르는 것을 목표로 합니다.



이번 서브젝트에서 반복문 사용을 제한하여 재귀적인 함수 사용 방법을 습득해 봅니다.


Chapter 2

Instructions

- 빌드 옵션은 -Wall -Wextra -Werror 을 넣어주세요.
- 전역변수 사용을 강권합니다.
- 각 문제 별 제한 사항이 있으니 RedBox를 잘 읽어 주세요.
- norm은 지키지 않습니다.
- <stdio.h>를 사용할 수 있습니다.

Chapter 3

Exercise 00 : forward_print

	Exercise 00
forward_print	
Turn-in directory : ex00/	
Files to turn in : forward_print.c	
Allowed function : write	

- 주어진 매개변수를 정방향으로 출력함수를 작성하세요.

```
void forward_solution(char *msg) {  
    // write code  
}
```

```
sgang — bash — bash — a.out — 65x6  
[bash-3.2$ ./a.out  
HelloWorld!
```

입력

```
sgang — bash — bash — bash — 65x6  
[bash-3.2$ ./a.out  
HelloWorld!  
HelloWorld!  
bash-3.2$
```


출력



반복문과 변수 선언을 금지합니다.

Chapter 4

Exercise 01 : backward_print

	Exercise 01
backward_print	
Turn-in directory : ex01/	
Files to turn in : backward_print.c	
Allowed function : write	

- 주어진 매개변수를 역방향으로 출력함수를 작성하세요.

```
void backward_solution(char *msg){  
    // write code  
}
```

```
sgang — bash — bash — a.out — 65×6  
[bash-3.2$ ./a.out  
!dlroWolleH]
```

입력

```
sgang — bash — bash — bash — 65×6  
[bash-3.2$ ./a.out  
!dlroWolleH  
HelloWorld!  
bash-3.2$ ]
```


출력



반복문과 변수 선언을 금지합니다.

Chapter 5

Exercise 02 : dynamic_programming

	Exercise 02
fibonacci	
Turn-in directory : ex02/	
Files to turn in : fibonacci.c	
Allowed function :	

- 매개변수로 넘어온 n번째 fibonacci수를 반환하세요.

https://en.wikipedia.org/wiki/Fibonacci_number

```
long long fibonacci(int N) {  
    // write code  
}
```

N의 범위: $1 \leq N \leq 90$

https://en.wikipedia.org/wiki/Dynamic_programming

이 챕터에서는 메모하는 기술을 배워 볼 수 있습니다. 그런데 dynamic_programming은 무슨 뜻 일까요? (별 뜻은 없고, memoization은 멍멍해.. 멋있는 이름을 붙인 것 이라 한다...)

리처드 E. 벨먼



Ps

- 반복문을 이용하여 Memoization을 해보세요.
- Memoization하지 않고 반복문만 이용하여 Fibonacci(N)을 $O(N)$ 만에 구해보세요.



https://en.wikipedia.org/wiki/Big_O_notation