Dash – Problem solving_00

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

Contents

Chapter 1

Foreword

This project focuses on solving problems and aims to develop a diverse perspective on problems.

In this subject, learn how to use recursive functions by limiting the use of repetitive statements.

Chapter 2

Instructions

- Include -Wall -Wextra -Werror for build options.
- I recommend using global variables
- There are limitations for each problem, so please read RedBox carefully.
- We don't keep norm.
- <stdio.h> is available.

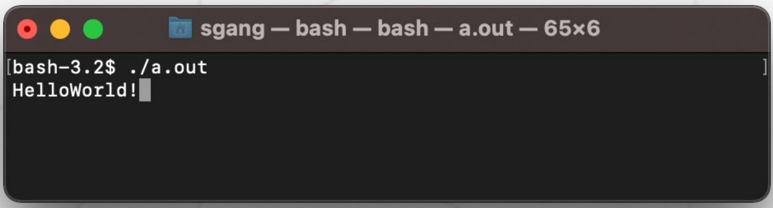
Chapter 3

Exercise 00 : forward_print

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

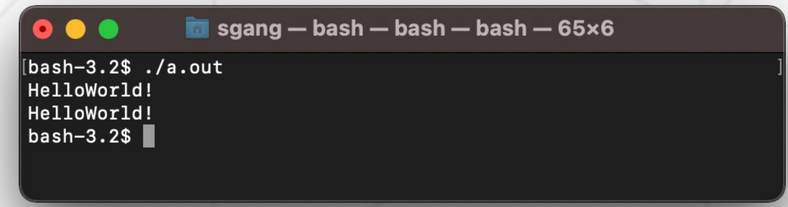
- Write the output function in the forward direction for the given.

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



```
bash-3.2$ ./a.out
HelloWorld!
```

input



```
bash-3.2$ ./a.out
HelloWorld!
HelloWorld!
bash-3.2$
```

output

Prohibit declaration of repeat statements and variables.

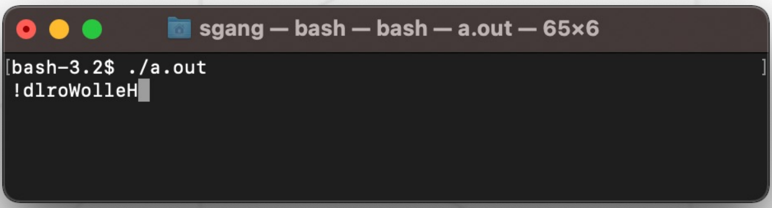
Chapter 4

Exercise 01 : backward_print

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

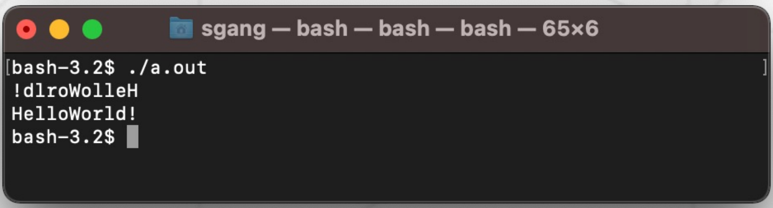
- Write an output function that reverses the given parameters.

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



```
bash-3.2$ ./a.out
!dlroWolleH
```

input



```
bash-3.2$ ./a.out
!dlroWolleH
HelloWorld!
bash-3.2$
```

output

Prohibit declaration of repeat statements and variables.

Chapter 5

Exercise 02 : dynamic_programming

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

- Return the nth number of fibonacci transferred to the parameter.
  https://en.wikipedia.org/wiki/Fibonacci_number

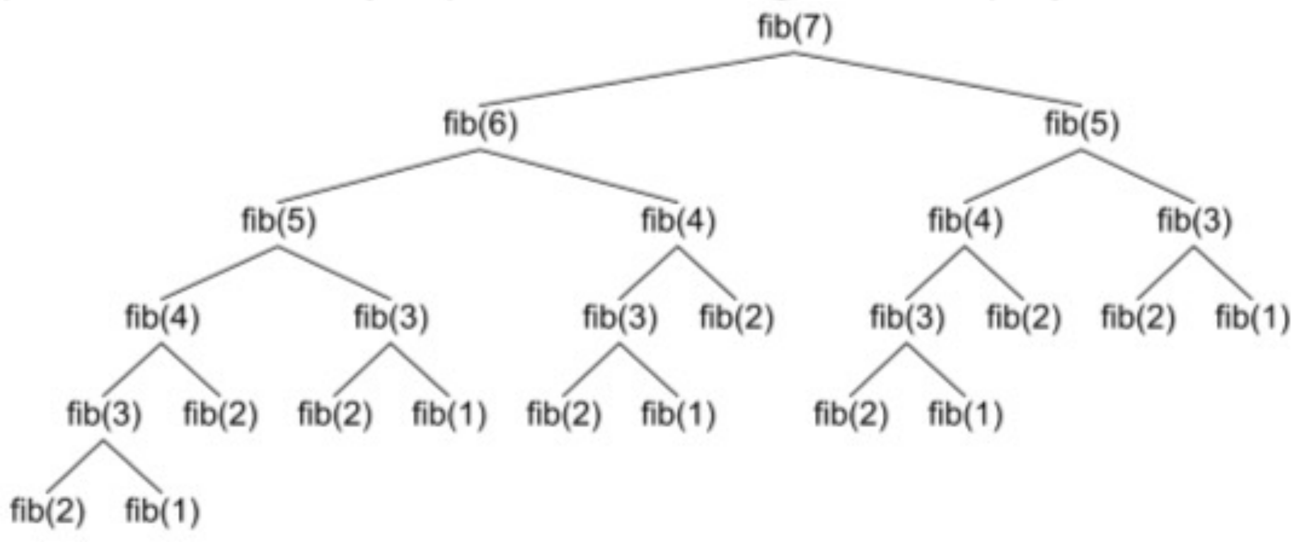
```
long long  fibonacci(int N) {
        // write code
}
```

N의 범위: 1 <= N <= 90

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

In this chapter, you can learn how to take notes
Yes, but dynamic_programming is not a problem
What does it mean? (It doesn't mean much,
memorization. It's bland... It's said to have given
a cool names...)

Richard E. Bellman



picture 1. Reasons to take notes

The reason for taking notes can be found in Figure 1. It can be seen that there are many overlapping parts in the composition called to obtain Fibonacci (7). This is to eliminate redundant (unnecessary) tasks by taking notes

Do not use a repeat statement.

Ps

- Memoize using a repetition sentence
- Find Fibonacci(N) in O(N) only by using a repetition sentence without Memoization.

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