C, C++ and DSA in depth

Recursion

Saurabh Shukla (MySirG)

Agenda

1. What is a recursion?
2. Recursion Tree | Tracing code
3. How to approach recursive solution?
4. Few examples

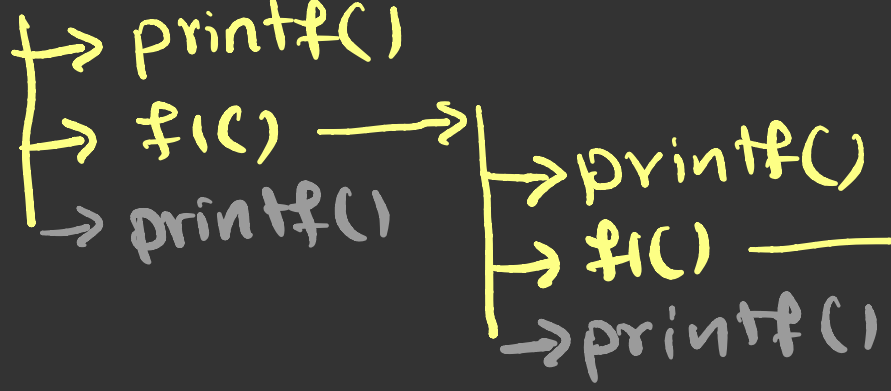
What is a recursion?

- Function calling itself is called recursion
- A recursive method solves a problem by calling a copy of itself to work on a smaller problem.
- It is important to ensure that the recursion terminates.

```
void f1( )
{
    printf("Hello");
    f1 ();
    printf("Bye");
}
```

```
void f2()
{
    printf("A");
}
```

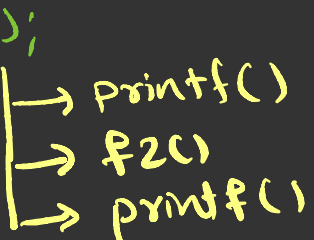
f1();



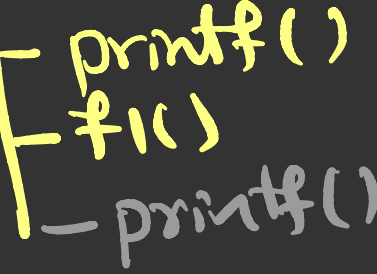
Infinite Recursion

HelloHelloHelloHello

f1();

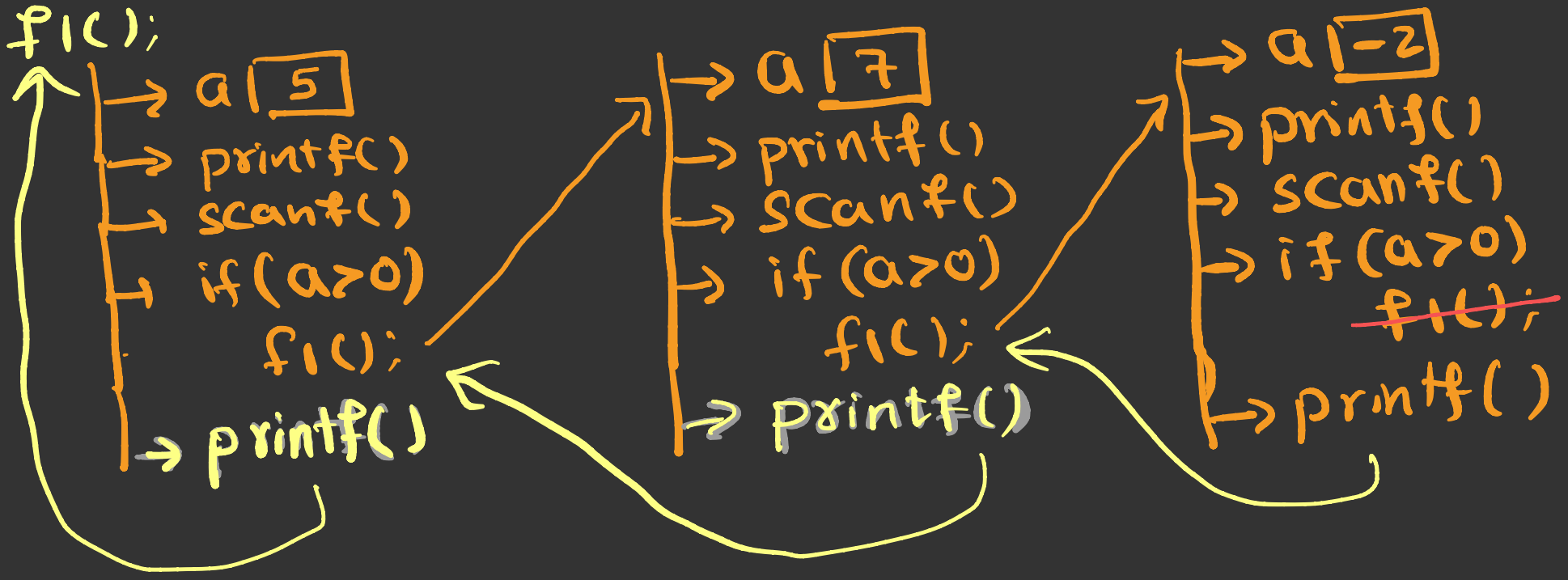


Hello A Bye



```
void f1()
{
    int a;
    printf("Enter a number");
    scanf("%d",&a);
    if(a>0)
        f1();
    printf("%d",a);
}
```

```
Enter a number 5
Enter a number 7
Enter a number -2
-2 7 5
```



- Each time the function call itself with a slightly simpler version of the original problem.

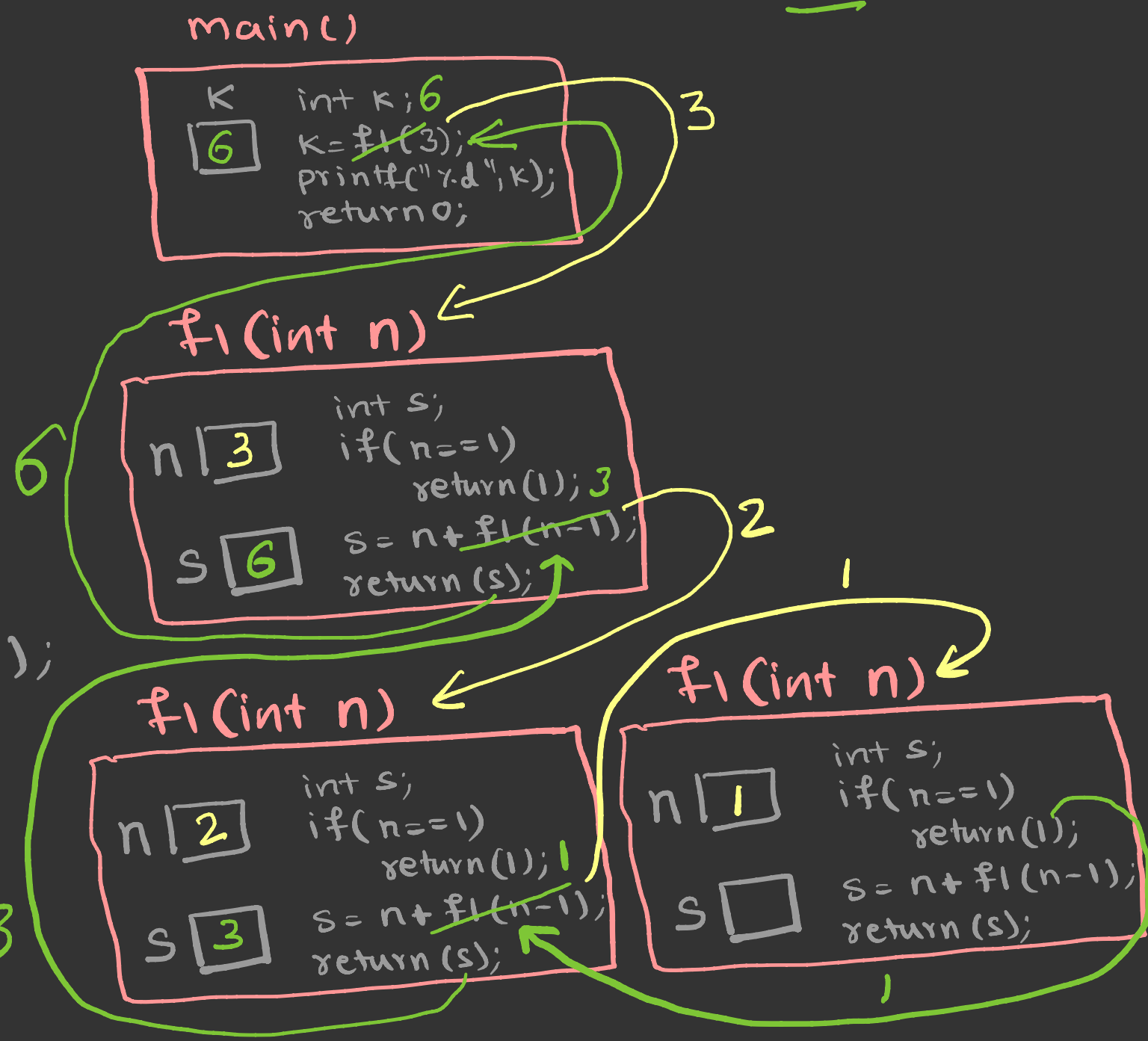
- Recursive code is generally shorter and easier to write than iterative code.

- Solution to some problems are easier to formulate recursively.

Recursion Tree

6

```
int main()
{
    int k;
    k=f1(3);
    printf("%d",k);
    return 0;
}
int f1(int n)
{
    int s;
    if(n==1)
        return(1);
    s = n+f1(n-1);
    return(s);
}
```



How to approach a Recursive Problem?

Write a recursive function to calculate sum of first n natural numbers.

```
int sum(int n)
{
    if(n==1)
        return 1;          ] BC
    return n+sum(n-1);
}                          RC
```

1. Sum(n) 1+2+3+4+..... +(n-1)+n
2. RC — n+sum(n-1) 1+2+3+....+(n-1)
3. BC — n==1 return(1) 1

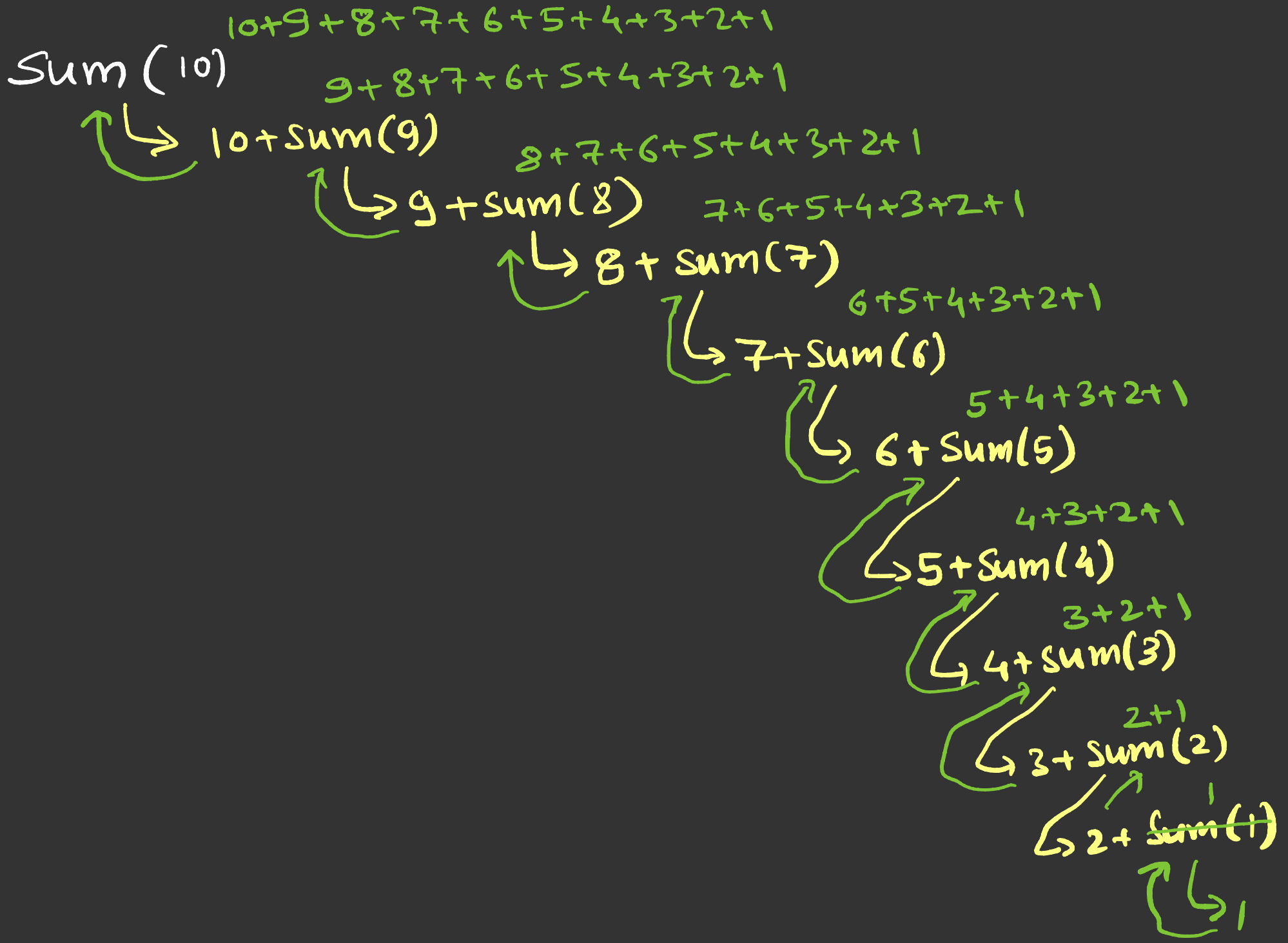
n = 10

sum(10) = (1+2+3+4+5+6+7+8+9) + 10 — 45

10 + sum(9) = (1+2+3+4+5+6+7+8+9) + 10 — 45

sum(2) = 1+2 + 10

sum(1) = 1


10+9+8+7+6+5+4+3+2+1
Sum(10)
9+8+7+6+5+4+3+2+1
10+Sum(9)
8+7+6+5+4+3+2+1
9+Sum(8)
7+6+5+4+3+2+1
8+Sum(7)
6+5+4+3+2+1
7+Sum(6)
5+4+3+2+1
6+Sum(5)
4+3+2+1
5+Sum(4)
3+2+1
4+Sum(3)
2+1
3+Sum(2)
1
2+Sum(1)
1

Write a recursive function to calculate factorial of n.

```
int fact(int n)
{
    if(n==0)
        return 1;
    return( n*fact(n-1));
}
```

(1)		fact(n)	$1 \times 2 \times 3 \times 4 \times \ldots \times (n-1) \times n$
RC	(2)	$n *$ fact(n-1)	$1 \times 2 \times 3 \times \ldots \times (n-1)$
BC	(3)	$n == 0$	1

Write a recursive function to print first n natural numbers.

```
void printN(int n)
{
    if(n>0)
    {
        printN(n-1);
        printf("%d",n);
    }
}
```

① printN(n) 1 2 3 4 5 ... (n-1) n

RC ② printN(n-1) 1 2 3 4 5 .... (n-1)
printf("%d",n); n

BC ③ n → 0