C Language

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.

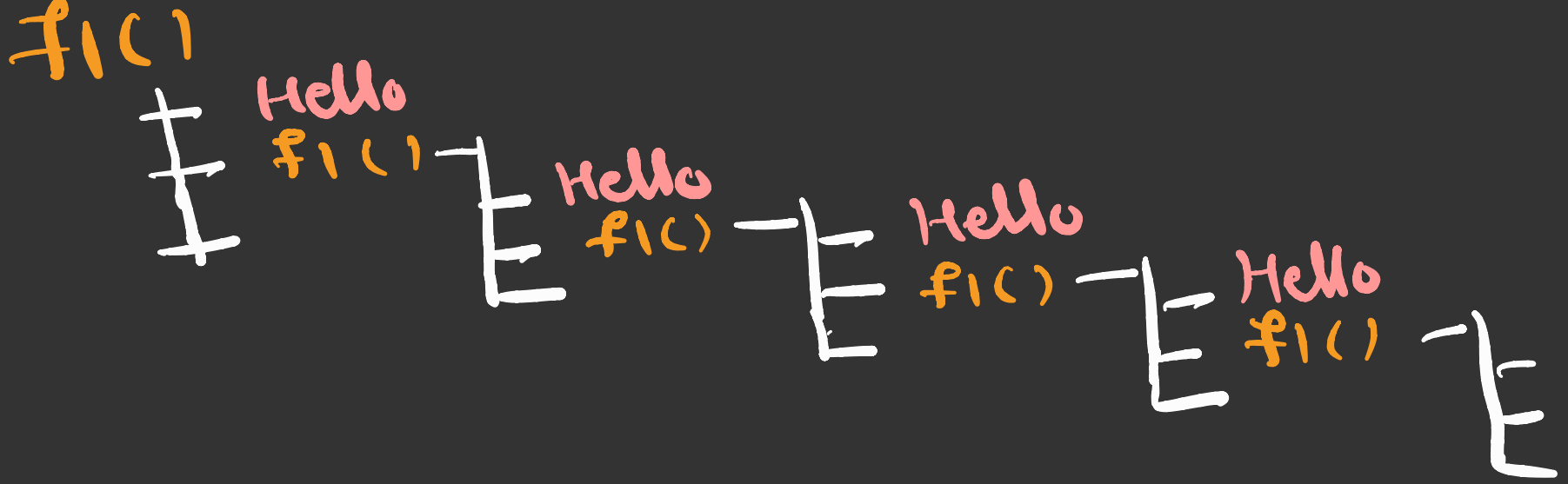
```
void f1()
{
    ≡
    f1();
    ≡
}
```

```
Main()
{
    main();
}
```

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

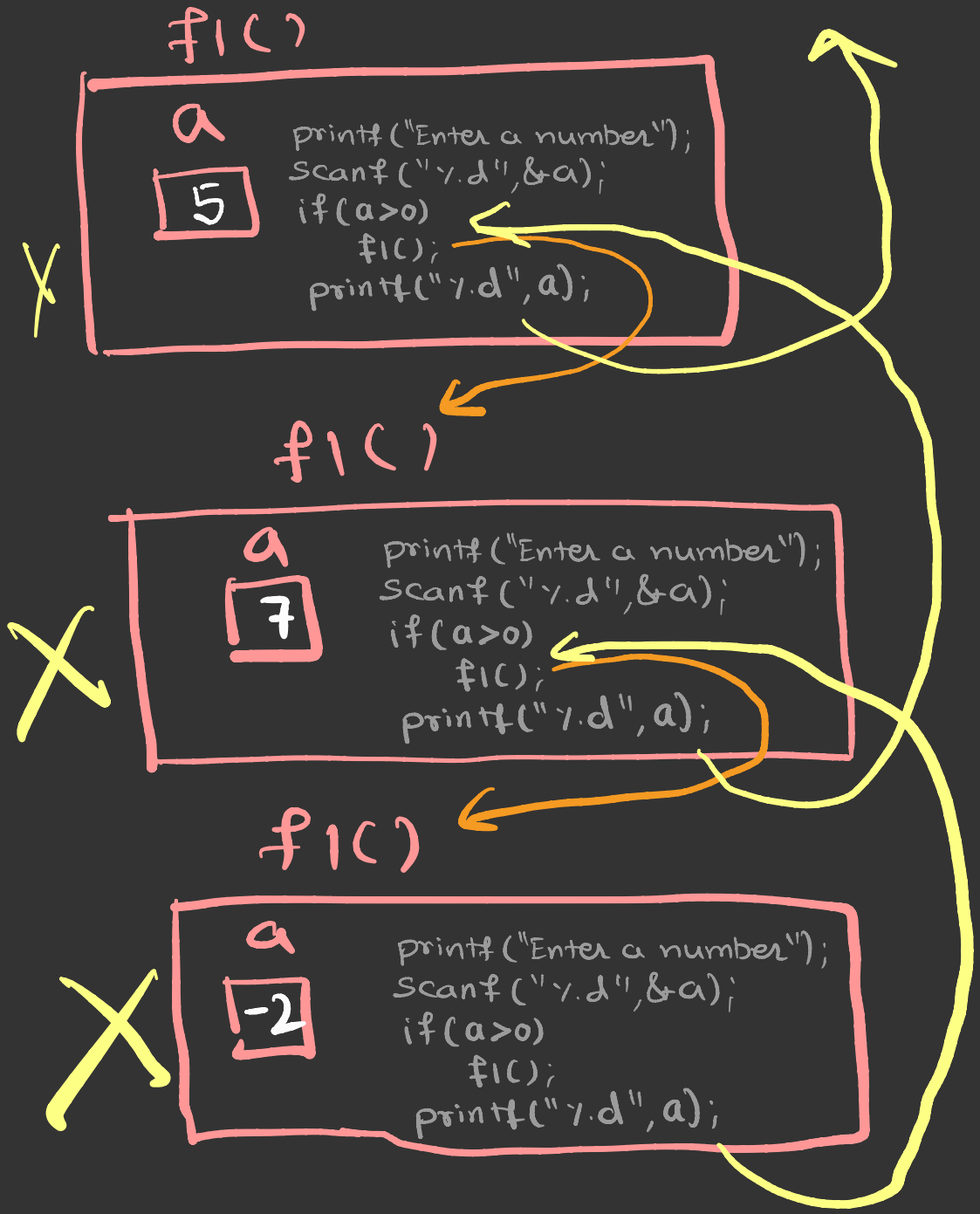
```
void f2()
{
    printf("A");
}
```

HelloABye



```
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



-275

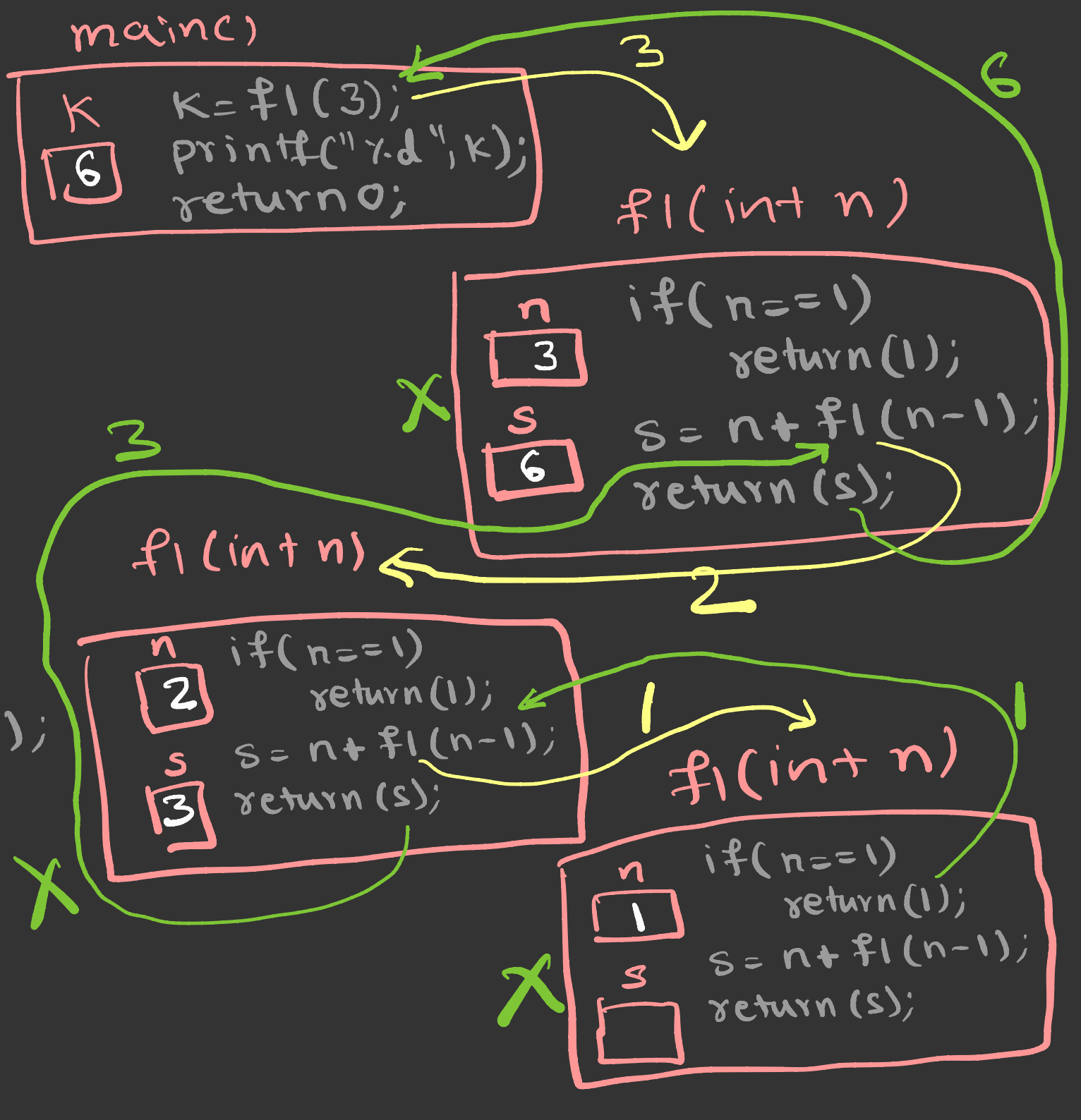
- Each time the function call itself with a slightly simpler version of the original problem.

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

- Solution to some problems are easier to formulate recursively.

```c
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);
}
```

6



f1(n) n + f1(n-1)

f1(4) 10

f1(3) 6

f1(2) 3

f1(1) 1

sum(100)

100 + sum(99)

99 + sum(98)

98 + sum(97

3 + sum(2)

2 + sum(1)

1

How to approach a Recursive Problem?

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

1. Sum(n) — $1+2+3+4+\ldots+n$
2. RC: n + Sum(n-1) — $1+2+3+\ldots+n-1$
3. BC: $n==1$ — 1

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