CS & IT ENGINEERING

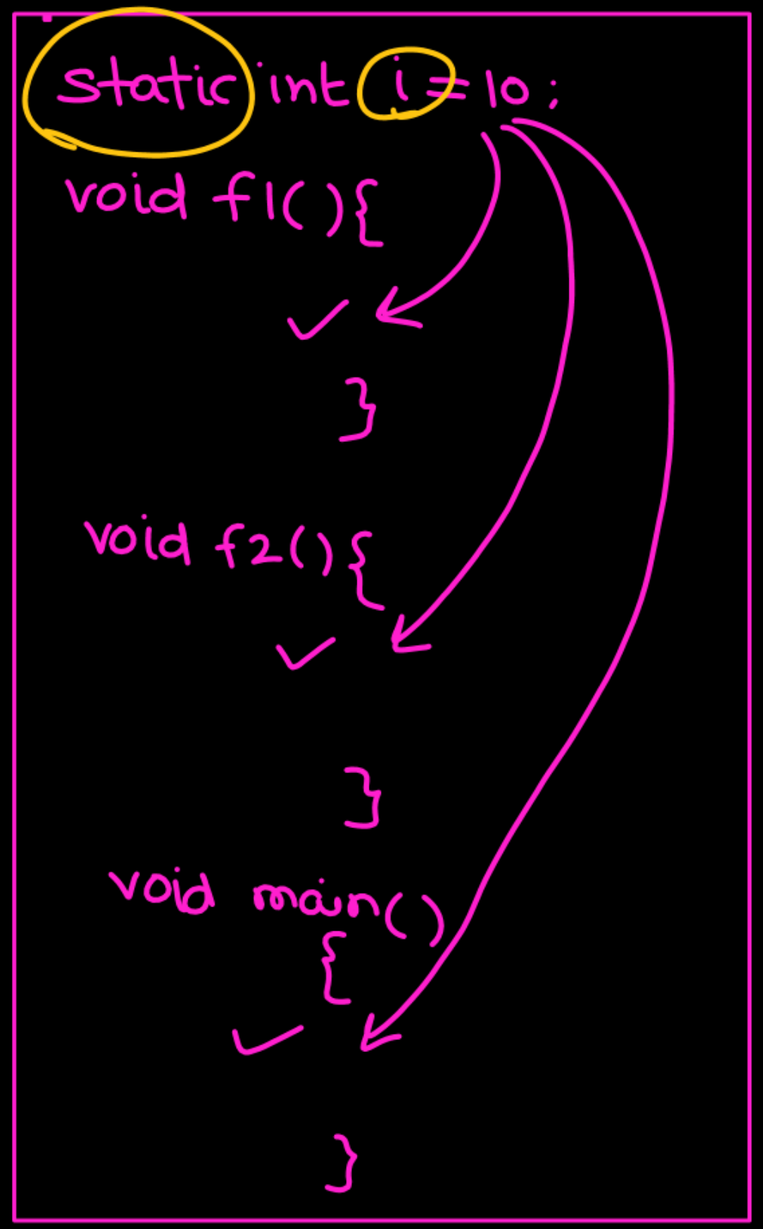
Programming in C

Functions and Storage Classes

Lec- 03

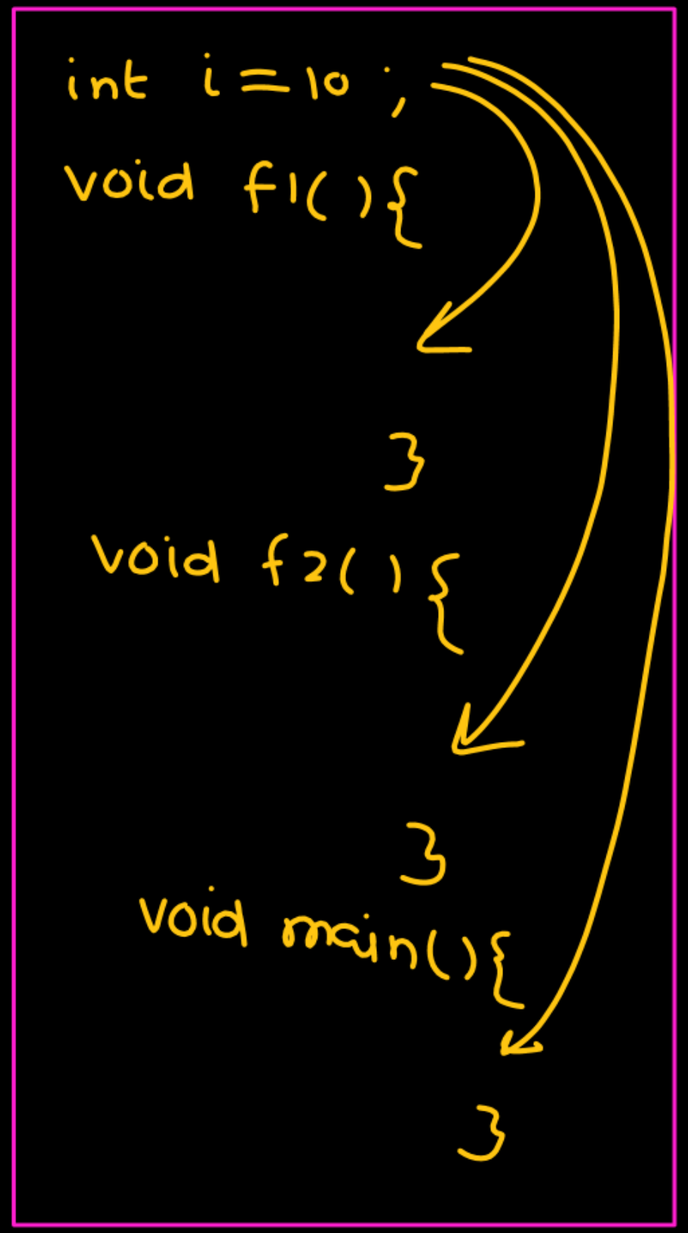
By- Pankaj Sharma sir

Storage Classes and Recursion

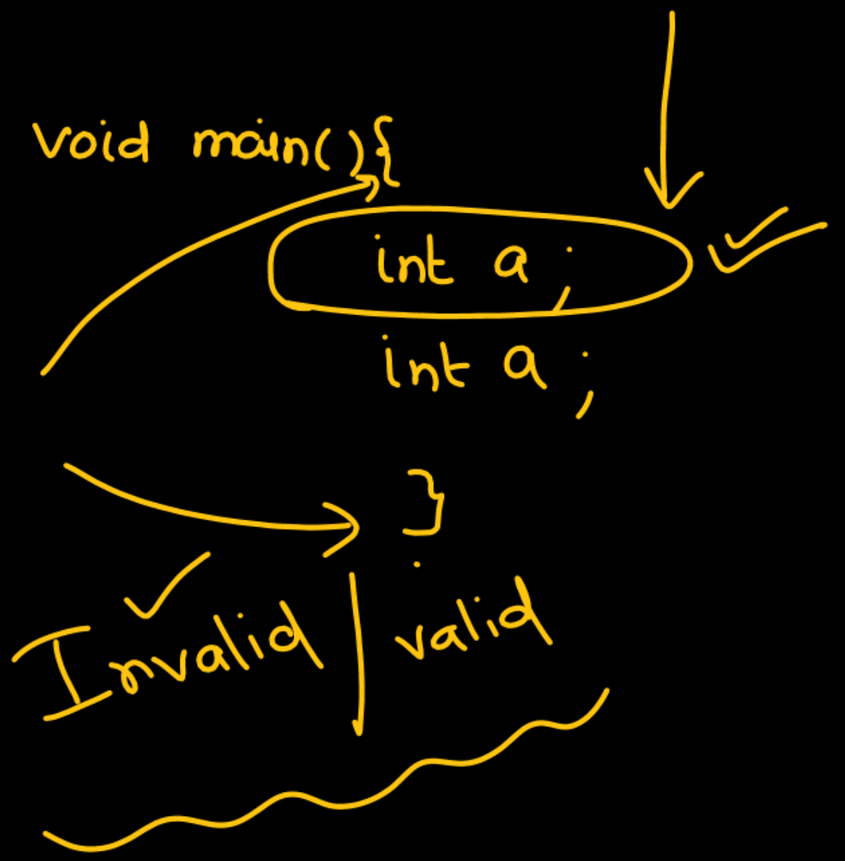


Static variable ✓

Static → global ✓



```
void main(){
    int a;
    int b;
}
```



(1)

```
void main(){
    int i=10;
    static int j = i ;
}
```

=

Invalid

(2)

```
Void main(){
    static int i = 10;
    static int j = i;
}
```

Invalid

⇒ using literal

(3)

```
void main(){
```

declaration

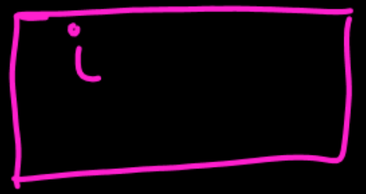
Compiler

```
static int i ;
static int i ;
}
```

×

(4)

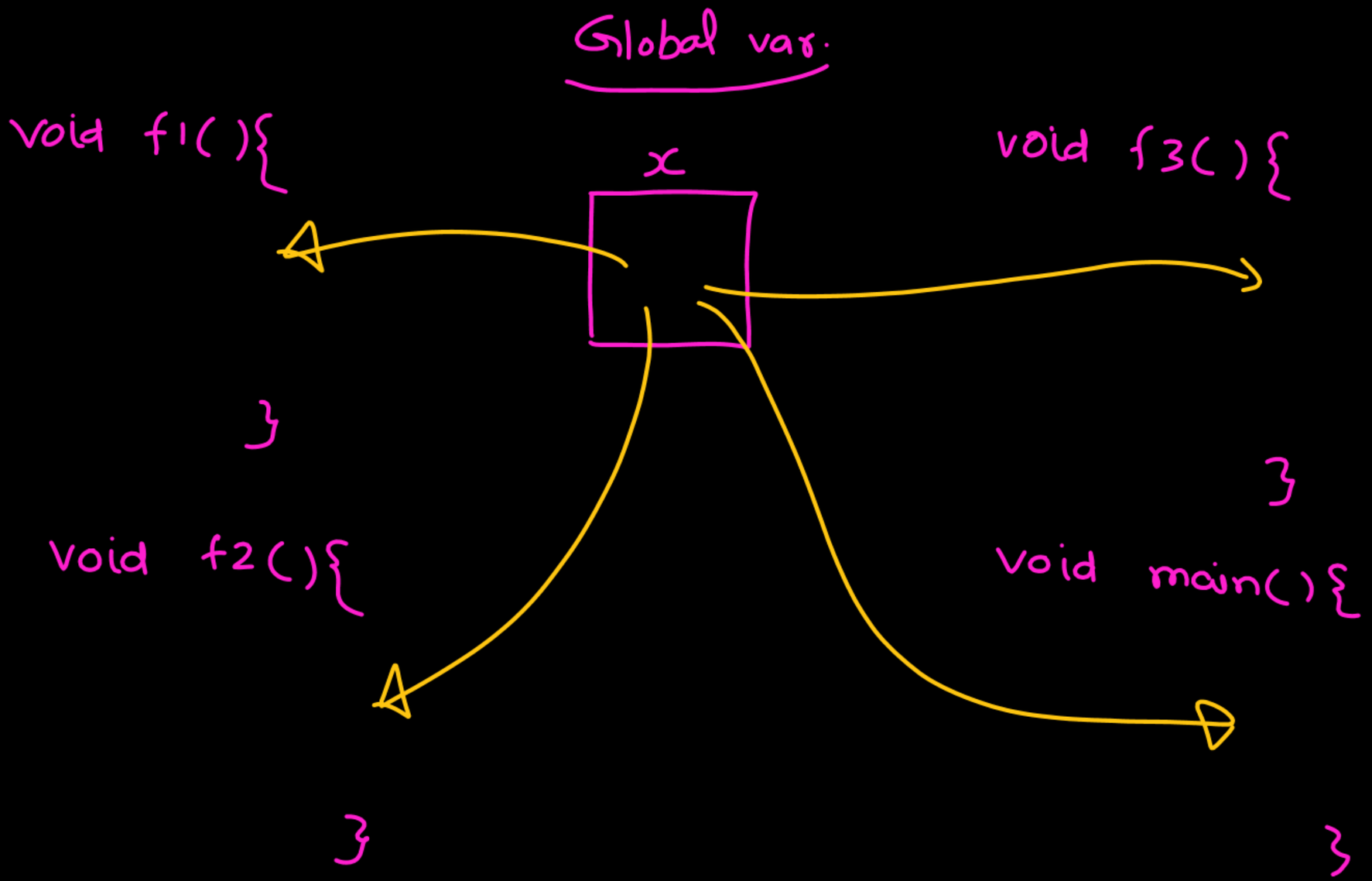
```
static int i ;
static int i ;
Void main(){
```

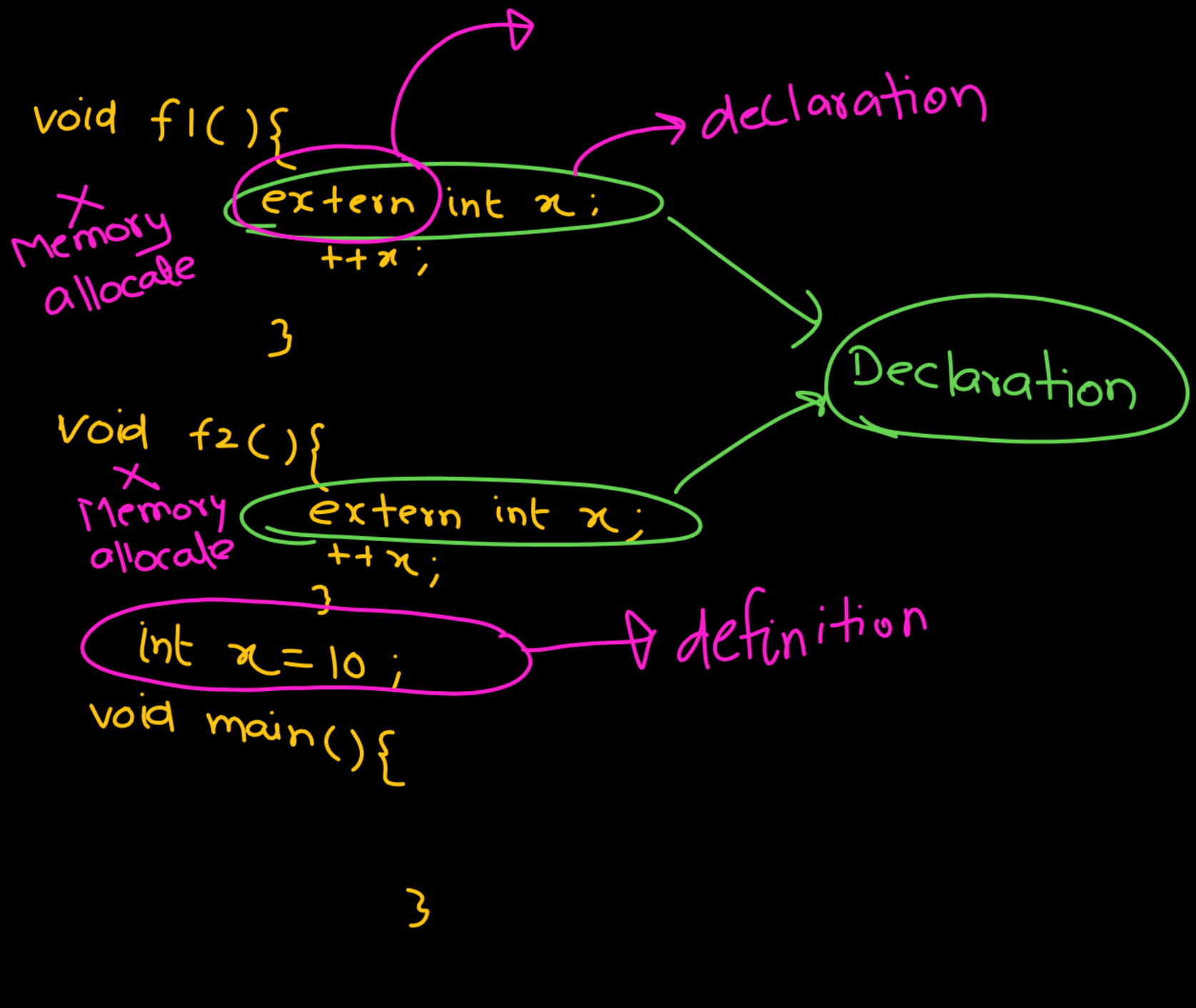


=

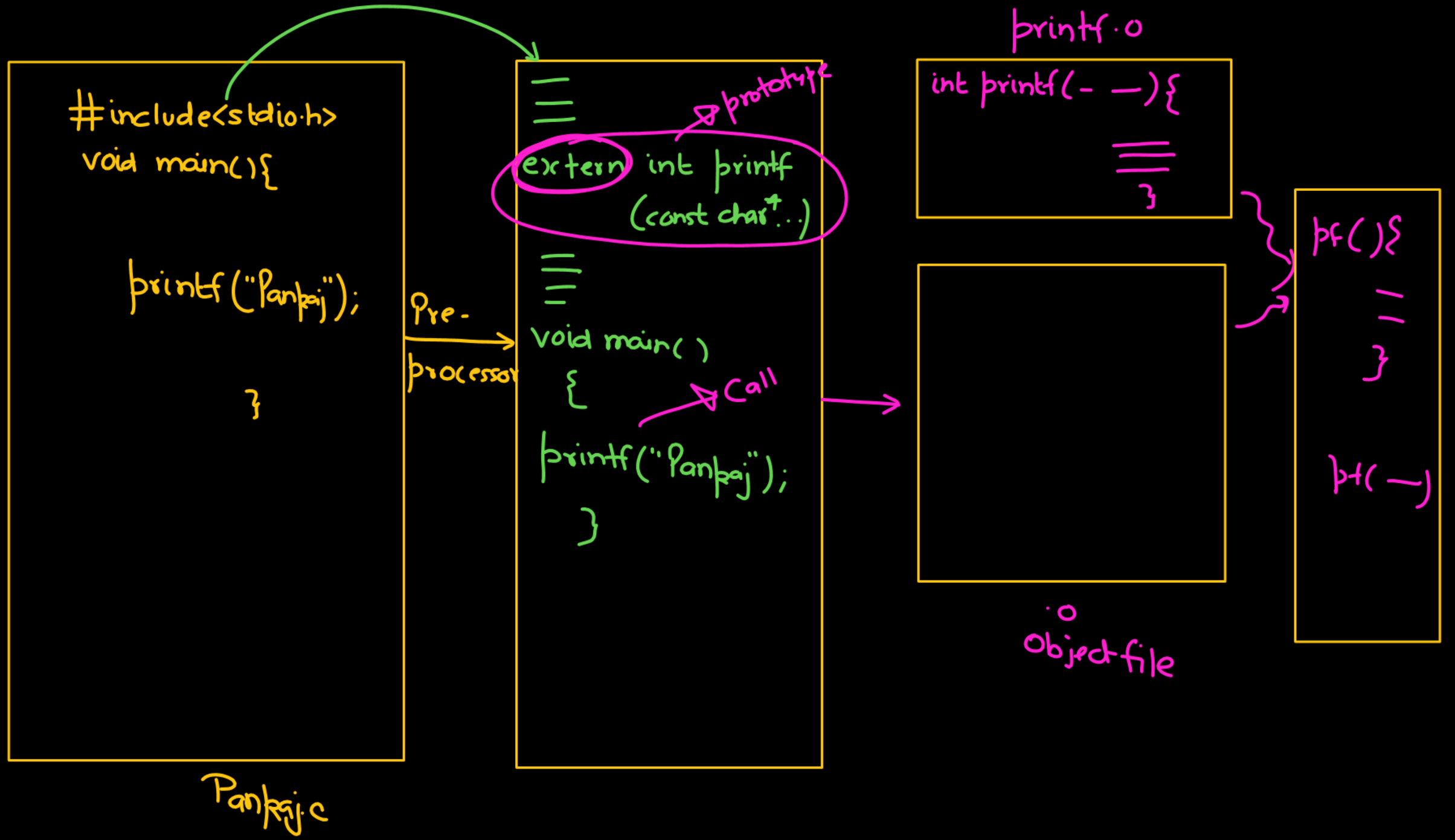
✓

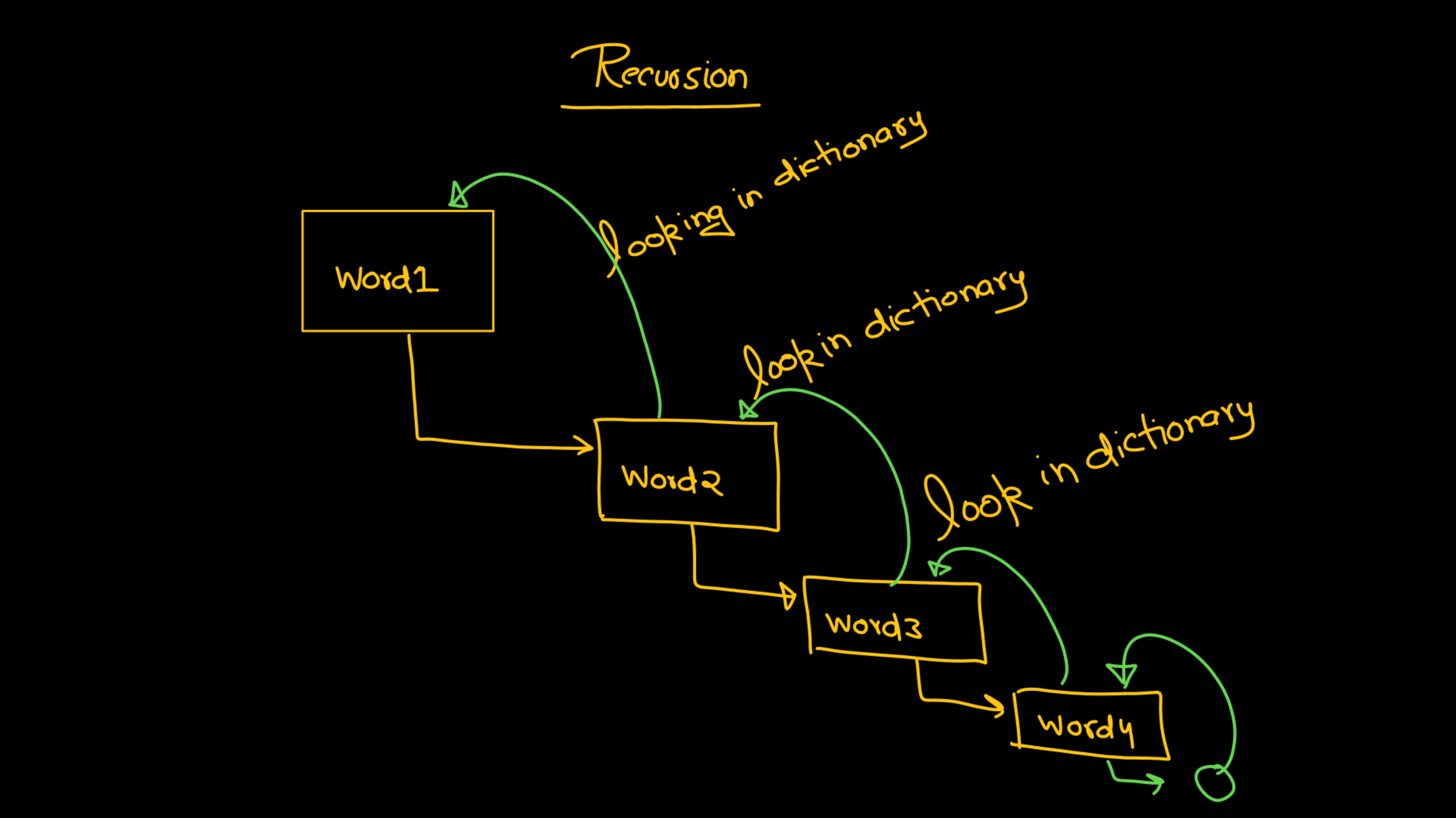
```
}
```

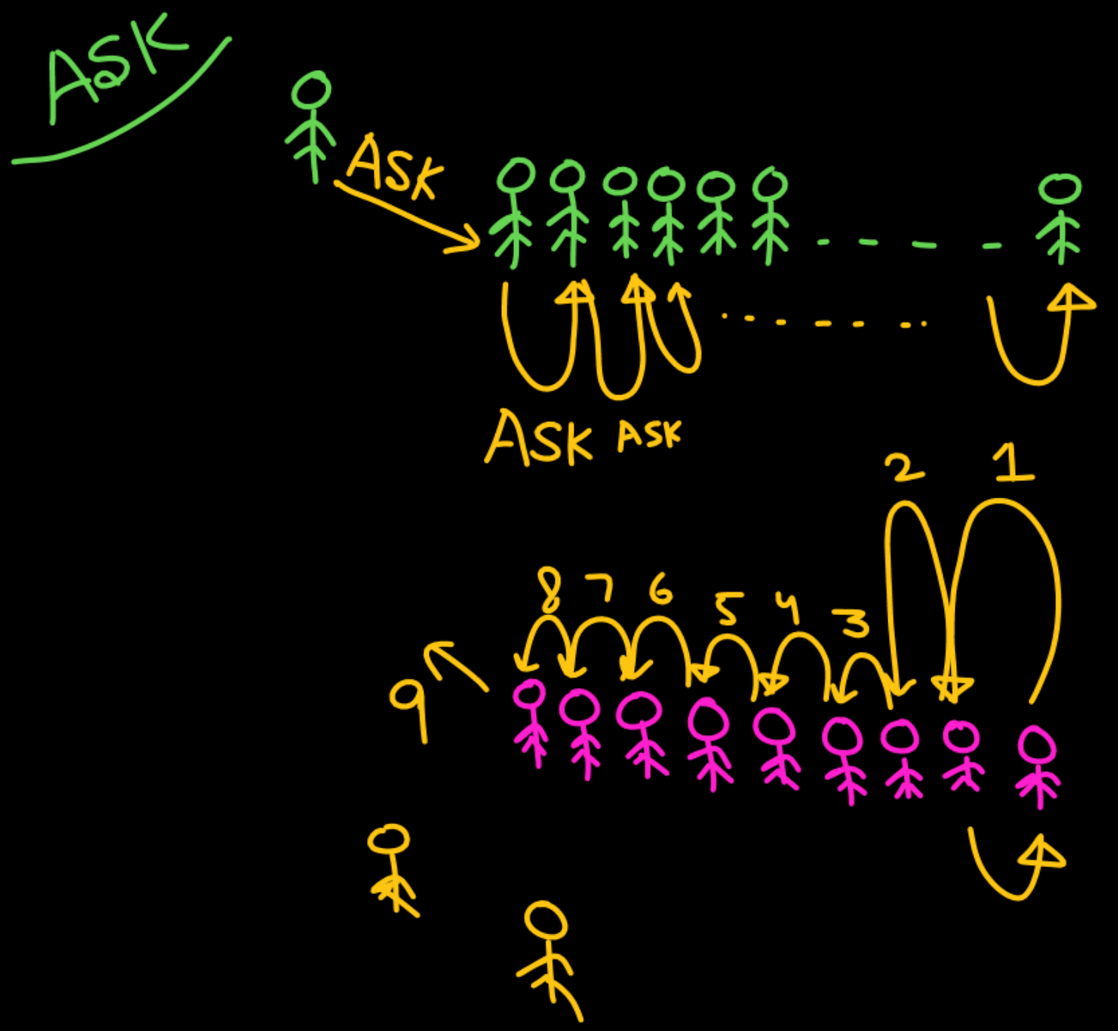

Global var.
x
Void f1(){
}
Void f2(){
}
void f3(){
}
Void main(){
}


void f1(){
extern int x;
++x;
declaration
x
Memory
allocate
}
Declaration
Void f2(){
x
Memory
allocate
extern int x;
++x;
}
int x=10;
definition
void main(){
}

Storage class, recursion, function


#include<stdio.h>
void main(){
printf("Pankaj");
}
Pankaj.c
Pre-
processor
extern int printf
(const char*...)
prototype
void main()
{
printf("Pankaj");
}
Call
printf.o
int printf(- -){
}
.o
object file
pf(){
}
pf(-)


Recursion
looking in dictionary
Word1
lookin dictionary
Word2
look in dictionary
Word3
Word4


ASK
Ask
Ask Ask
2 1
8 7 6 5 4 3
9

```
if ( n is small)
        {
            we can answer directly
            Easy to solve
            No recursion is needed
        }
else {
            input is large
            Hard case
            We can not answer directly
            recursion is needed
        }
```

Atleast

$n \geq 1$

i/p 2

o/p : PankajPankaj

i/p 3

o/p : PankajPankaj

i/p : 10

o/p : PankajPankajPankajPankajPankajPankajPankajPankajPankajPankaj

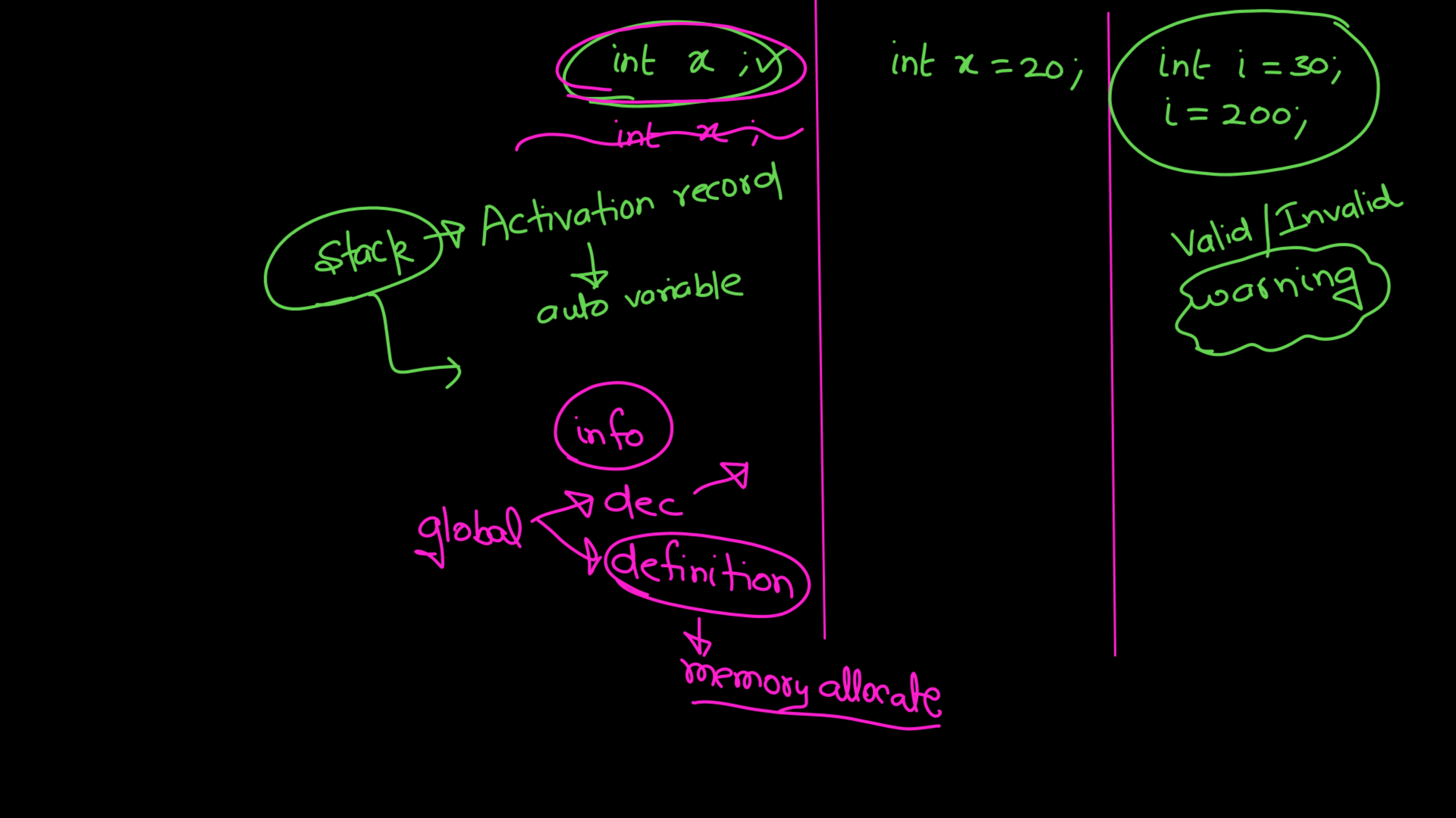
```
void fun(int n)
{
    if( ? )
    {
        Easy to solve
        direct answer
        No recursion
    }
    else {
        Hard to solve
        large input
        Can not answer directly
        rec. is needed
    }
}
```

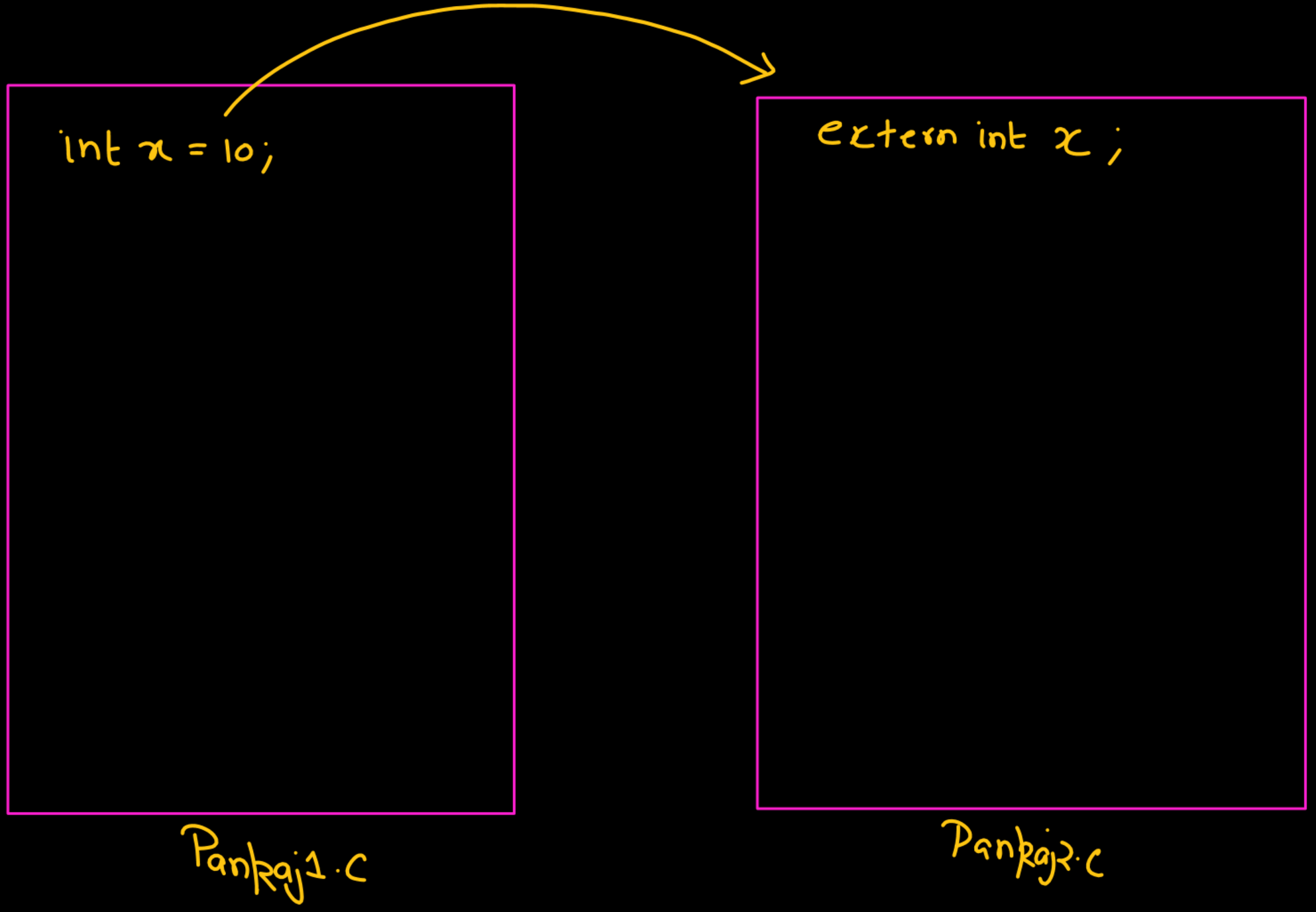
1000

⟹

Rec

```
void fun(int n)
{
    if(n==1)
    {
        printf("Pankaj");
        return;
    }
    else {
⟸       printf("Pankaj");
        fun(n-1);

    }
}
```


int x ;
int x ;
int x = 20;
int i = 30;
i = 200;
Valid | Invalid
warning
Stack
Activation record
auto variable
info
global
dec
definition
memory allocate


int x = 10;
extern int x;
Pankaj1.c
Pankaj2.c

Recursion

$n \geq 0$

i/p : 125 $\Rightarrow$ 1+2+5

o/p : 8

i/p : 9

o/p : 9

i/p : 2536 $\Rightarrow$ 2+5+3+6

o/p : 16

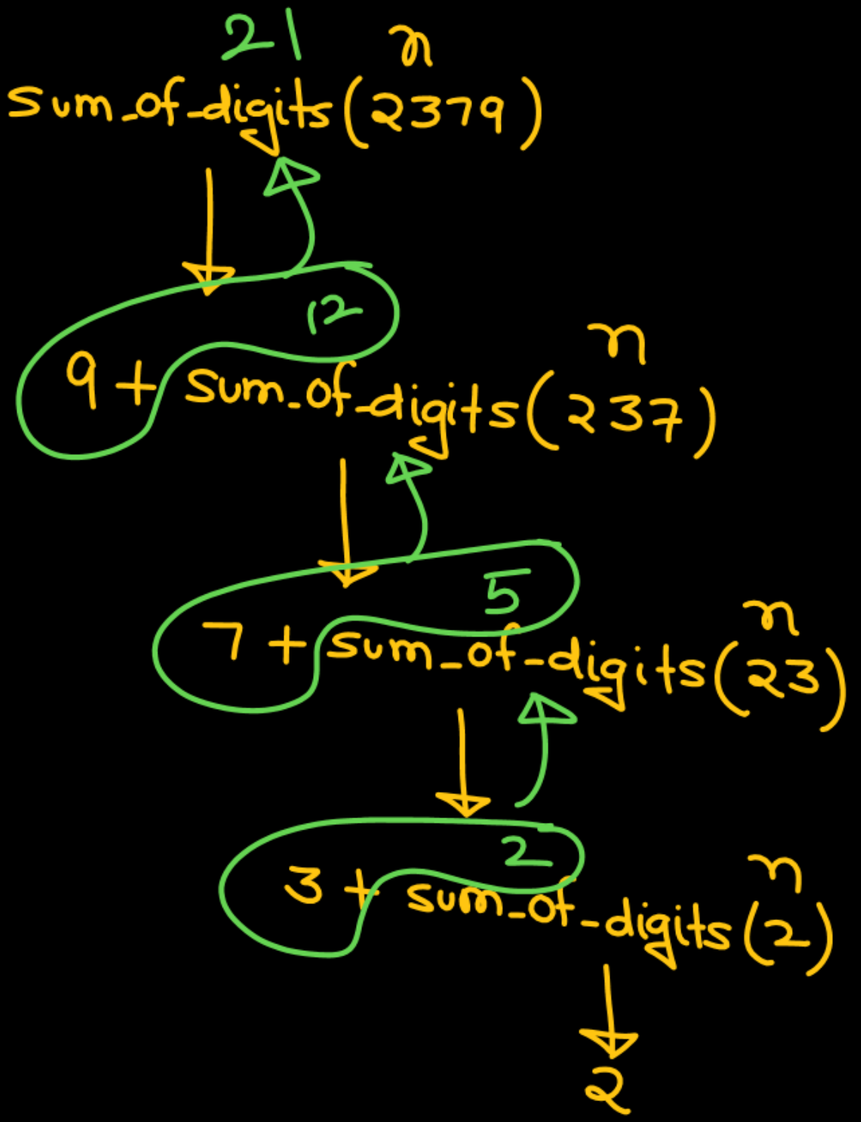
i/p : 1

o/p : 1

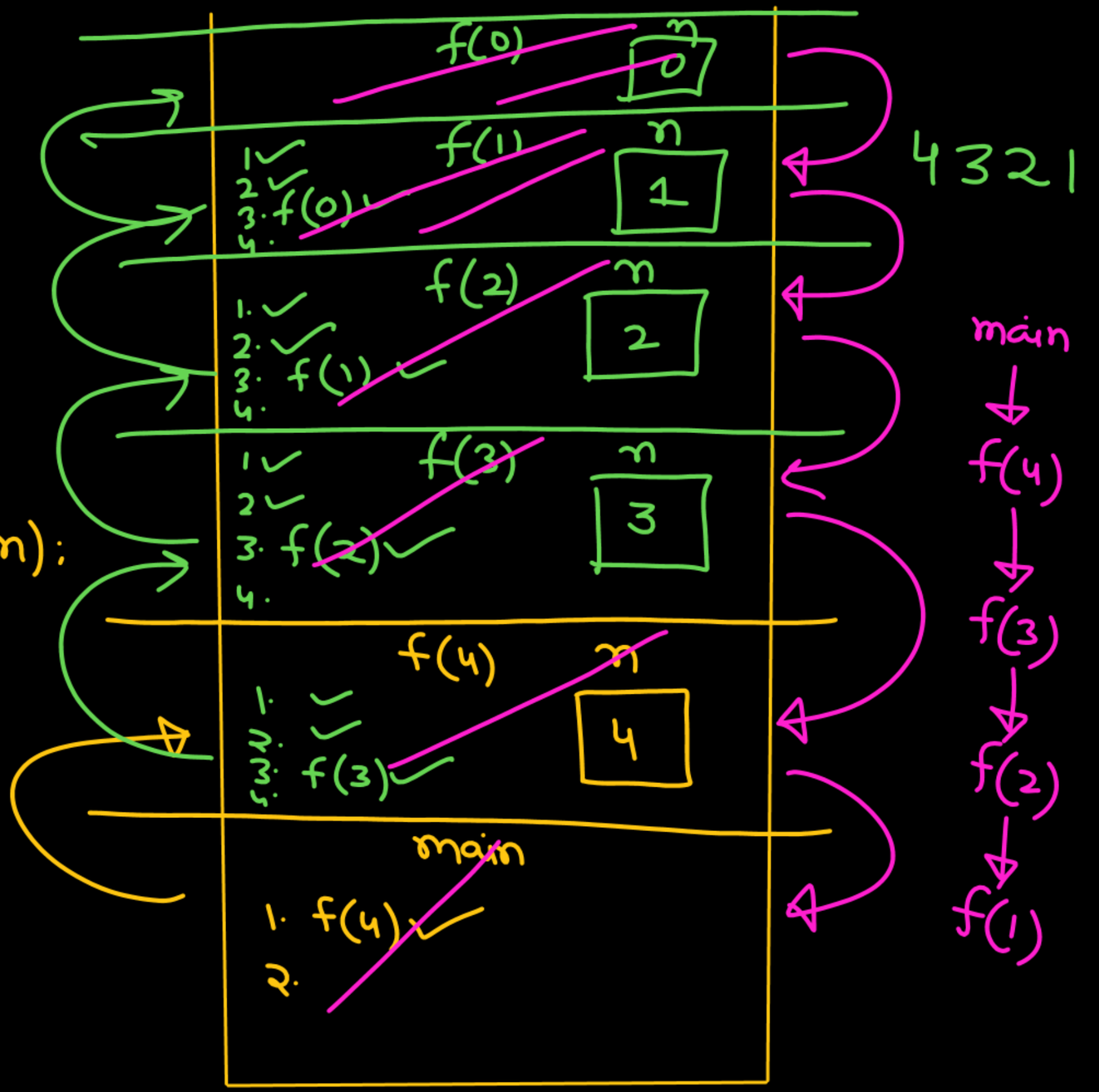
```
if (input is small)
 {
 directly answer
 No rec. is needed
 Easy to solve

 }
else {


}
```

```
int sum_of_digits(int n)
{
  if(n<=9)
      return n;
  else {
  return   n%10 + sum_of_digits(n/10);
      }
}
```

sum_of_digits(2379) → Act. arg

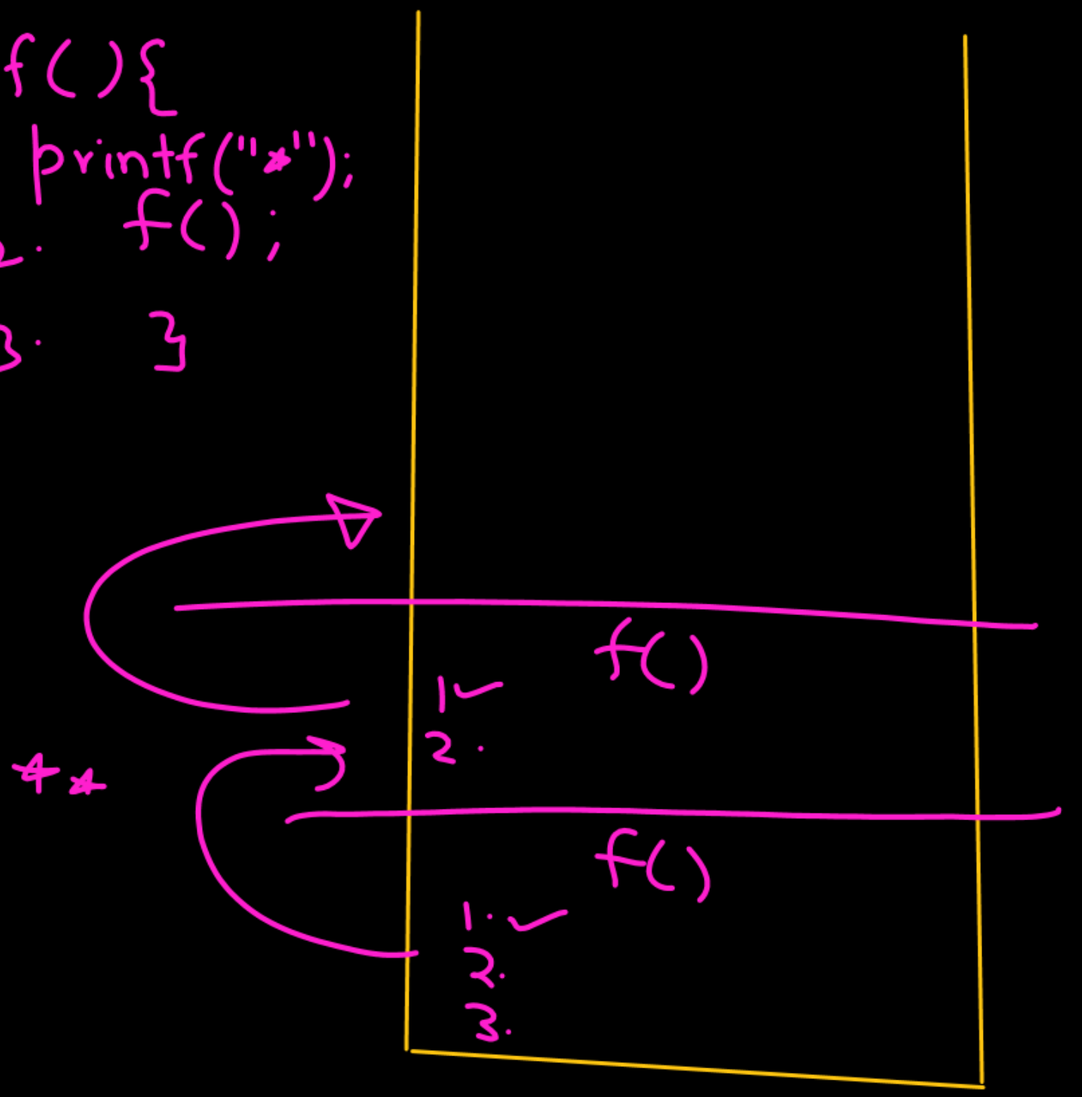


```
void f(int n)
    {
    1. if(n==0)
            return;
       else {
        2.   printf("%d", n);
        3.   f(n-1);
        4  }
    }
Void main(){
        1. f(4);
        2.   }
```

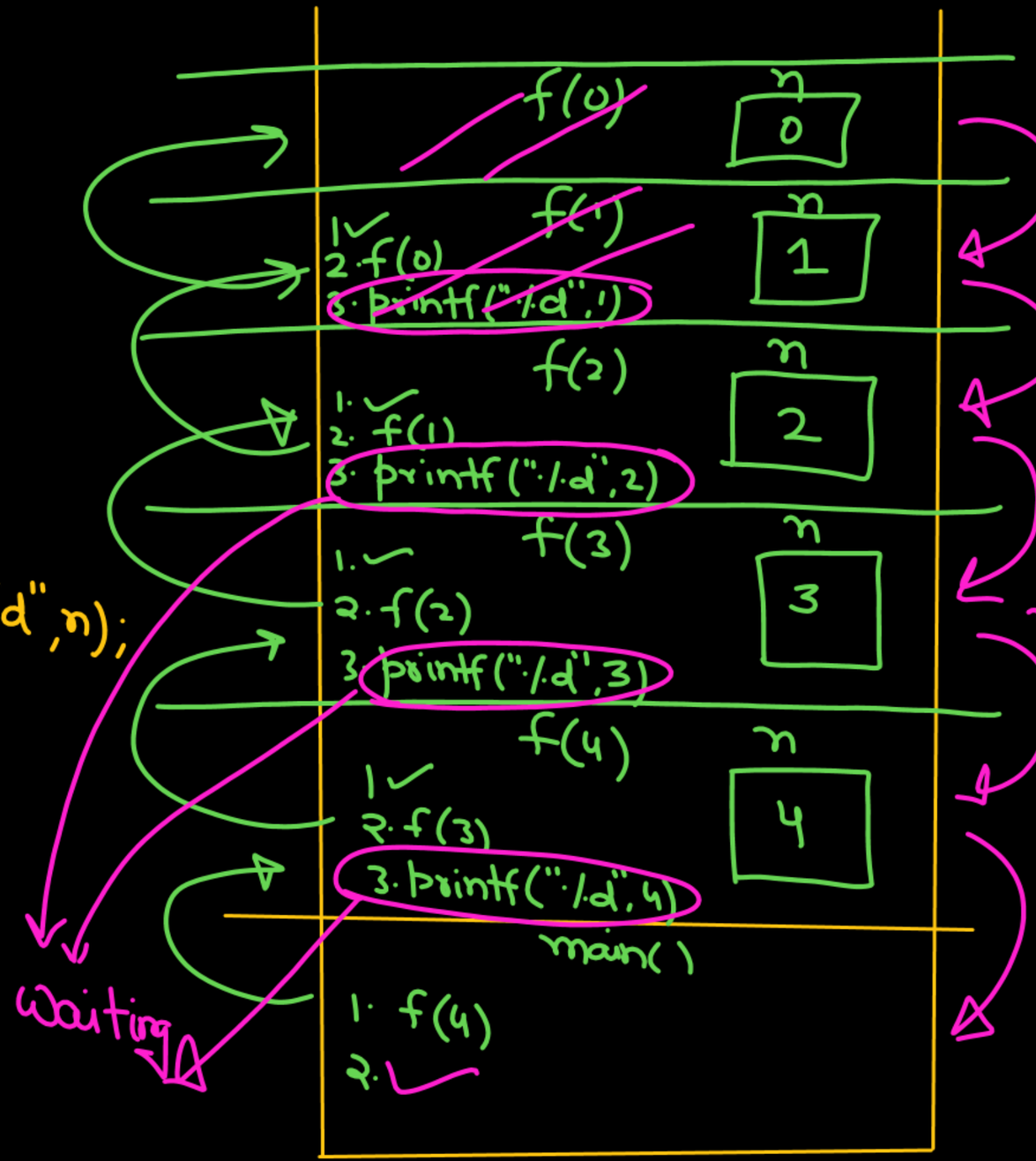


base case
base values
base parameter

```
void f(){
1.  printf("*");
2.  f();
3.  }
```

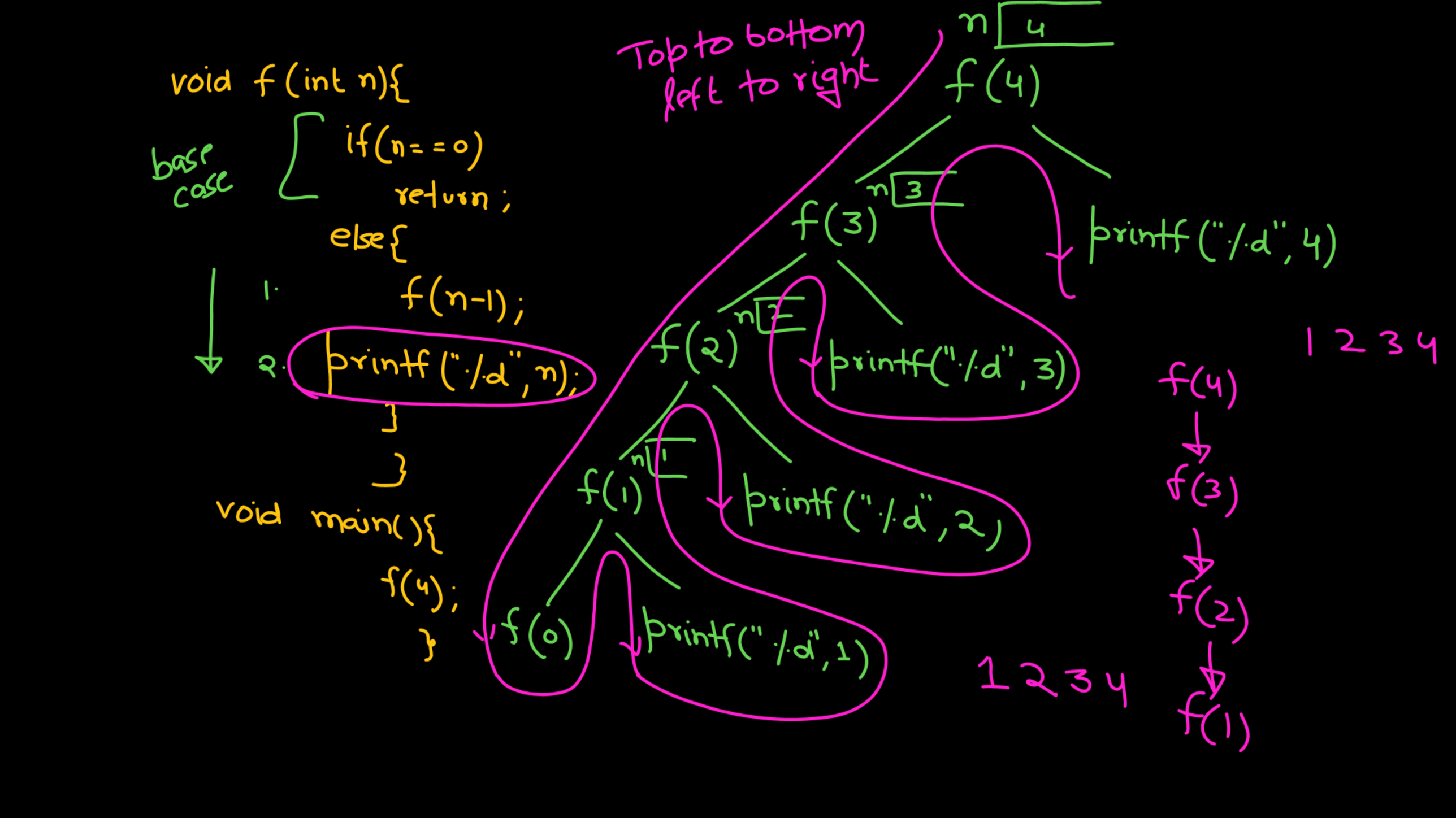


```
void f(int n){
1.   if(n==0)
         return;
     else{
2.       f(n-1);
3.       printf("%d",n);
     }
}
void main(){
1    f(4);
2.   }
```



1234

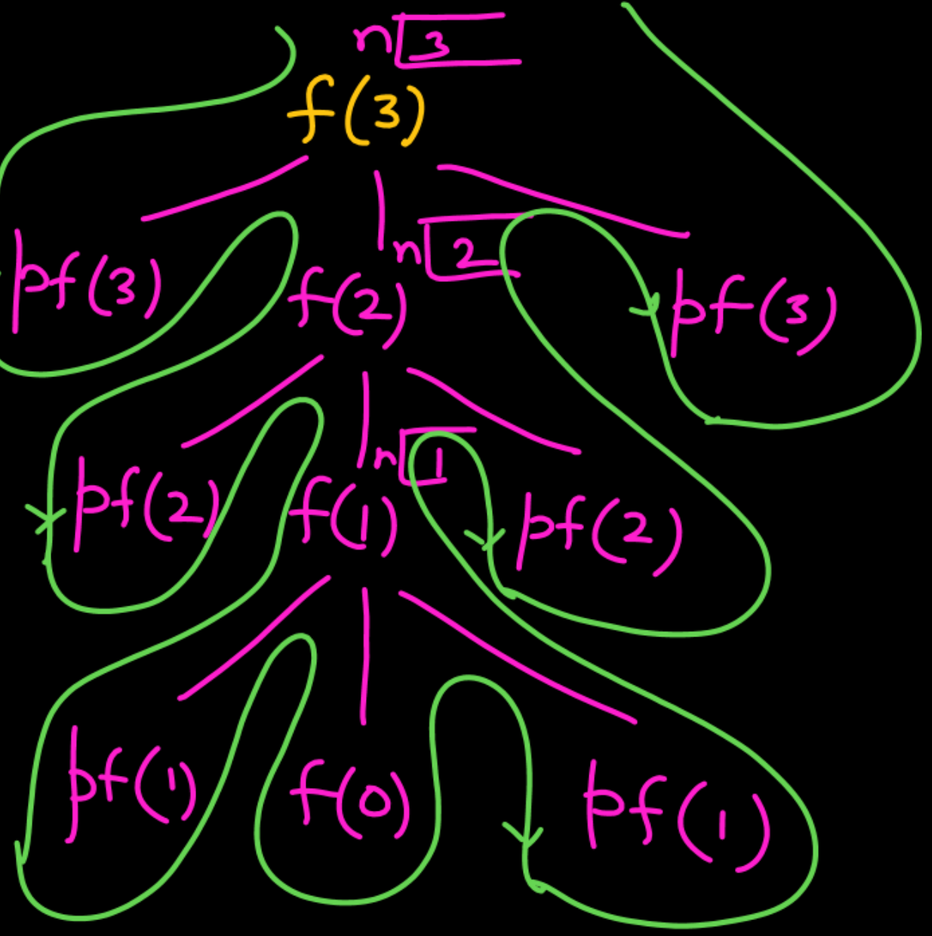
main → f(4) → f(3) → f(2) → f(1)


void f(int n){
if(n==0)
return;
else{
1. f(n-1);
2. printf("%d", n);
}
}
void main(){
f(4);
}
base case
Top to bottom
left to right
n 4
f(4)
n 3
f(3)
printf("%d", 4)
n 2
f(2)
printf("%d", 3)
n 1
f(1)
printf("%d", 2)
f(0)
printf("%d", 1)
1 2 3 4
f(4)
f(3)
f(2)
f(1)
1 2 3 4

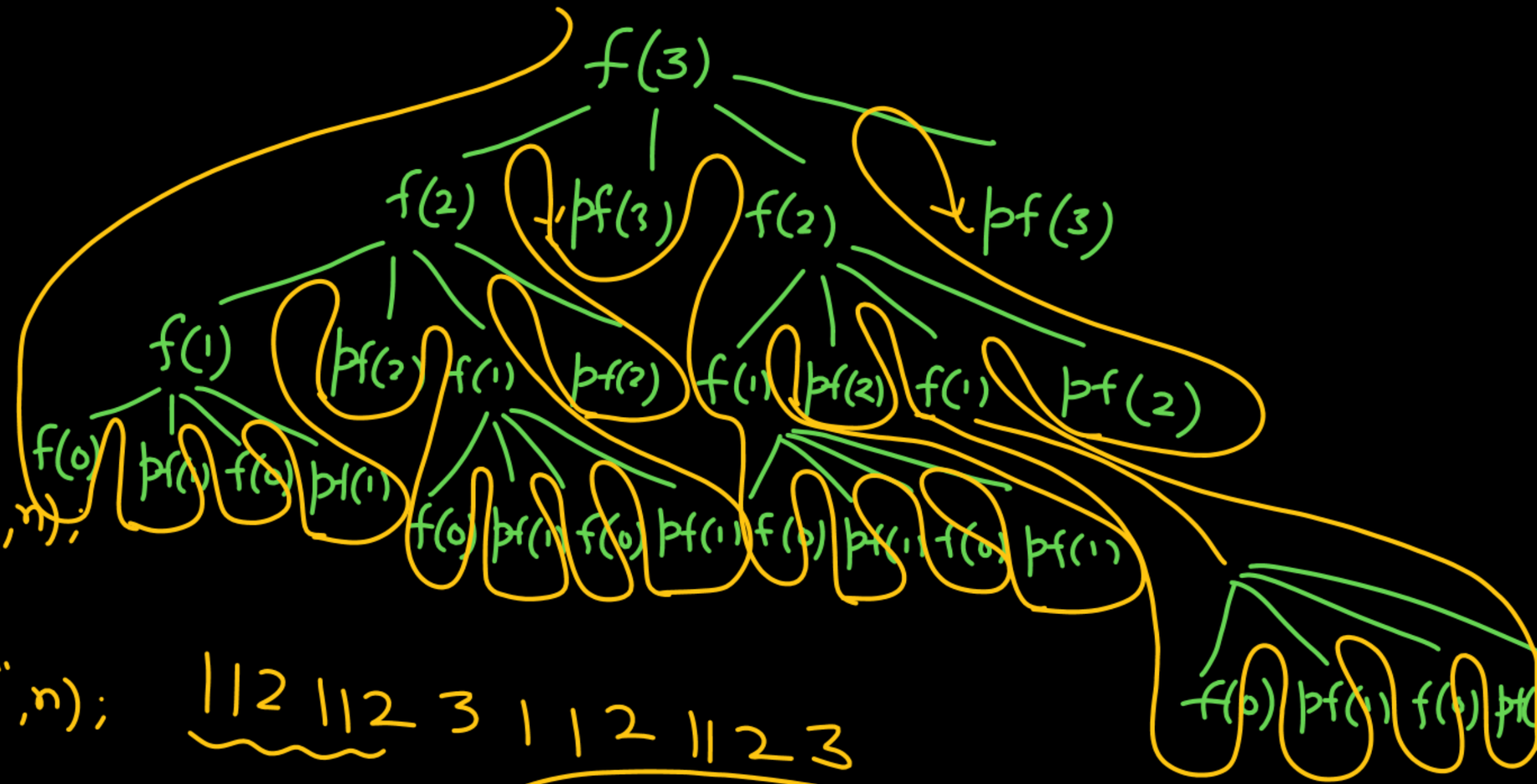
```
void f(int n)
{
  if(n==0)
      return;
  else{
1.  printf("%d",n);
2.  f(n-1);
3.  printf("%d",n);
  }
}
void main(){
      f(3);
  }
```

321|23

321123



```
void f(int n){
    if(n==0)
        return;
    else {
1.      f(n-1);
2.  printf("%d",n);
3.      f(n-1);
4.  printf("%d",n);
    }
}
void main(){
    f(3);
}
```



1 1 2 1 1 2 3 1 1 2 1 1 2 3

Rec-tree method

static variable → data segment

HW $a, b > 0$

$a \times b \Rightarrow$ using recursion

$a + b \Rightarrow$ using recursion

$a^b \Rightarrow$ using recursion

decimal to binary $\Rightarrow$ using recursion

1st year / 1st time Rec.


THANK YOU GW
SOLDIERS !