Recursion – Part 3

Now after wrapping up the whole process, and framing it is a little difficult and a bit lengthy.

Hence take those examples again:

```cpp
#include<iostream>
using namespace std;

int print(int p,int q){
    int a= 10;
    int b =20;

    if(p==2 || q==2){
        return 2;
    }
    else{
        //recursion
        cout<<a<<" "<<b<<endl;
        return print(p-1,q-1);
    }

}

int main(){
    int n;
    cin>>n;
    print(n,n);
    return 0;
}
```
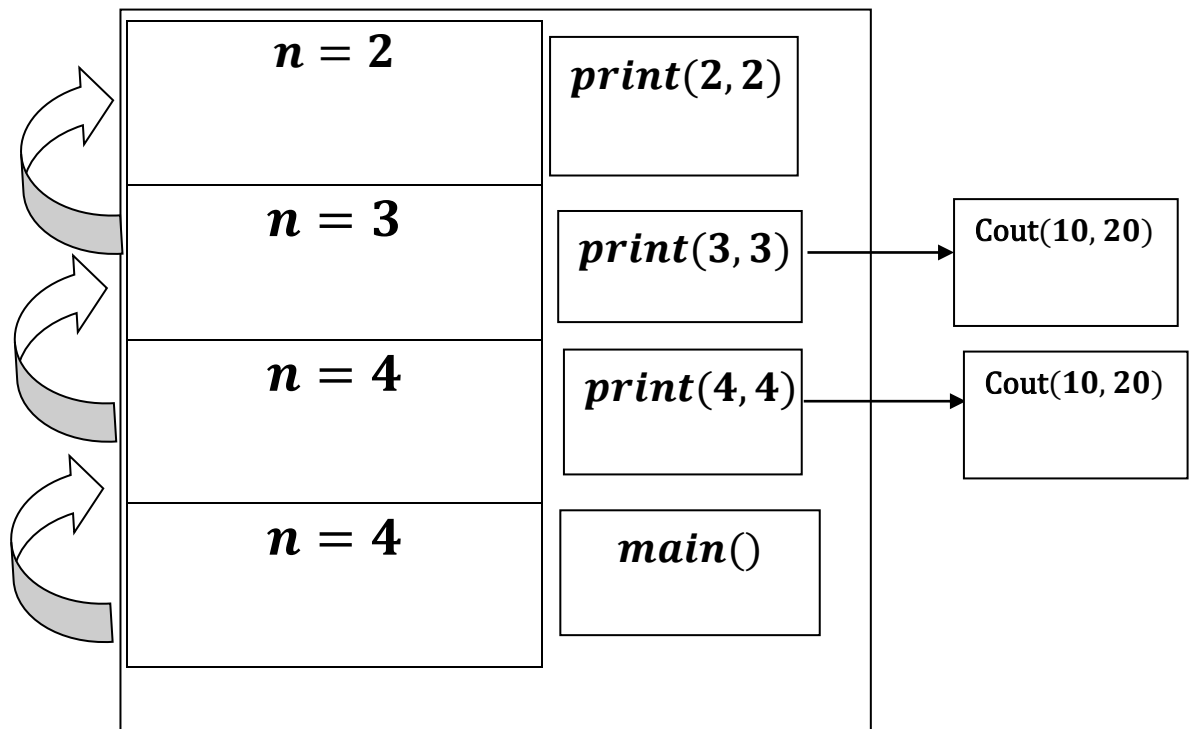
We can simplify the steps like:

$n = 4$

<u>Push Operation</u>:

$n = 2$	$print(2, 2)$
$n = 3$	$print(3, 3)$ → Cout(10, 20)
$n = 4$	$print(4, 4)$ → Cout(10, 20)
$n = 4$	$main()$

Note, how it will get excuted?

1st Part

1st $n = 4$ *and n will get pushed in main stack frame.*

2nd $p = 4$ *and* $q = 4$ *and* p, q *will get pushed in stack.*

3rd *it will check* $p = 2$ *or* $q = 2$ *is true or false, now it is false.*

4rth *it will go to else part and print a and b variable.*

5th *it will return i.e. call print* $(4 - 1 = 3, 4 - 1 = 3)$ *recursively.*

2nd Part

1st $p = 3$ *and* $q = 3$ *and* p, q *will get pushed in stack.*

**2nd** *it will check* $p = 2$ *or* $q = 2$ *is true or false, now it is false.*

3rd *it will go to else part and print a and b variable.*

4rth *it will return i.e. call print* $(3 - 1 = 2, 3 - 1 = 2)$ *recursively.*

As it calls $print(p = 2, q = 2), p, q$ *will be pushed in the stack frame and then will check* $p = 2$ *or* $q = 2$ *is true or false , now it is true , hence return* 2. *As it will not go to else part hence no `cout` statement runs.*

Pop operation

2	$print(2, 2)$
2	$print(3, 3)$
2	$print(4, 4)$
2	$main()$
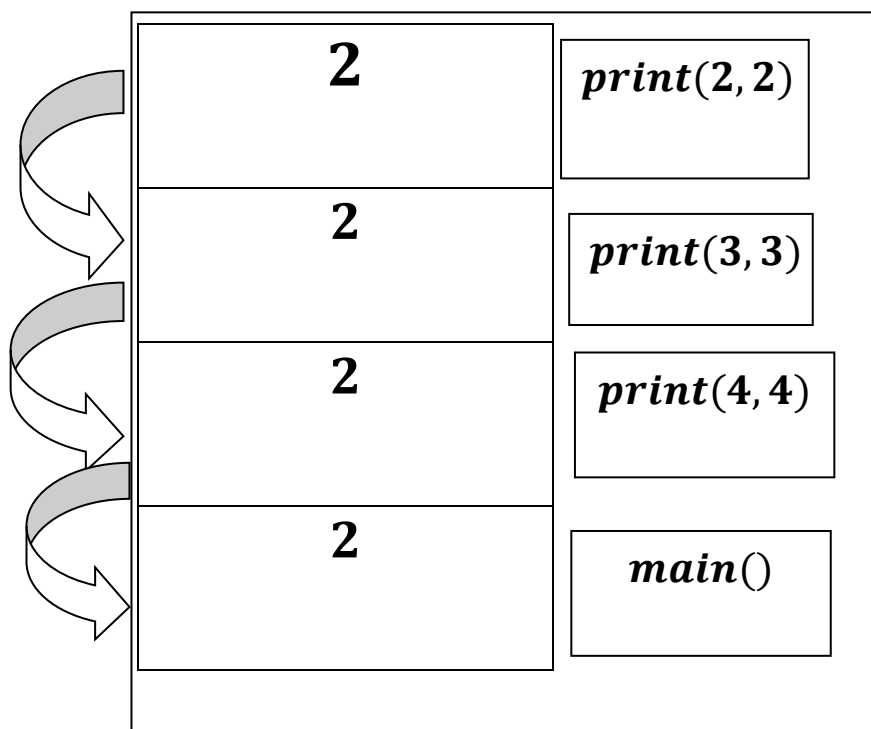
2 *will be popped to main function and from main function or main function's stack frame it will again get popped out so that we can see the return value.*

2nd Example:

```cpp
#include<iostream>
using namespace std;

int print(int n){

    if(n==0){
        return 0;
    }

    cout<<n<<endl;
    return print (n-1);
}

int main (){

    int n;
    cin>>n;
     print(n);
    return 0;
}
```
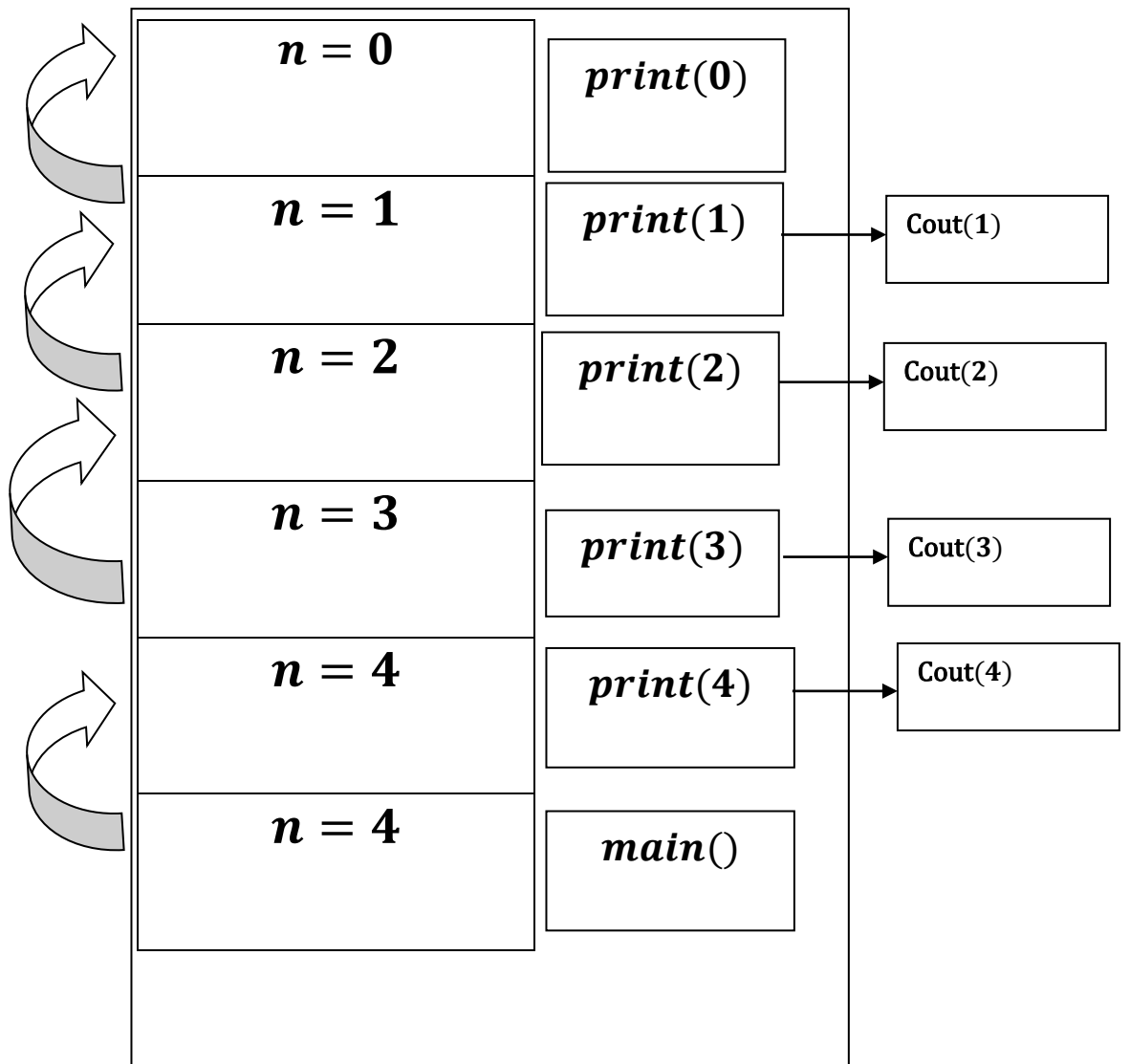
$let, n = 4$

Push Operation:

$n = 0$	$print(0)$	
$n = 1$	$print(1)$	Cout(1)
$n = 2$	$print(2)$	Cout(2)
$n = 3$	$print(3)$	Cout(3)
$n = 4$	$print(4)$	Cout(4)
$n = 4$	$main()$	

Pop Operation:

0	*print*(**0**)
0	*print*(**1**)
0	*print*(**2**)
0	*print*(**3**)
0	*print*(**4**)
0	*main*()

Similarly,

3rd Example:

```cpp
#include<iostream>
using namespace std;

int print(int n){
    if(n==0){
        return 0;
    }
    return 1+ print(n-1);
}

int main(){

    int n;
    cin>>n;
    int a =  print(n);
    return 0;
}
```
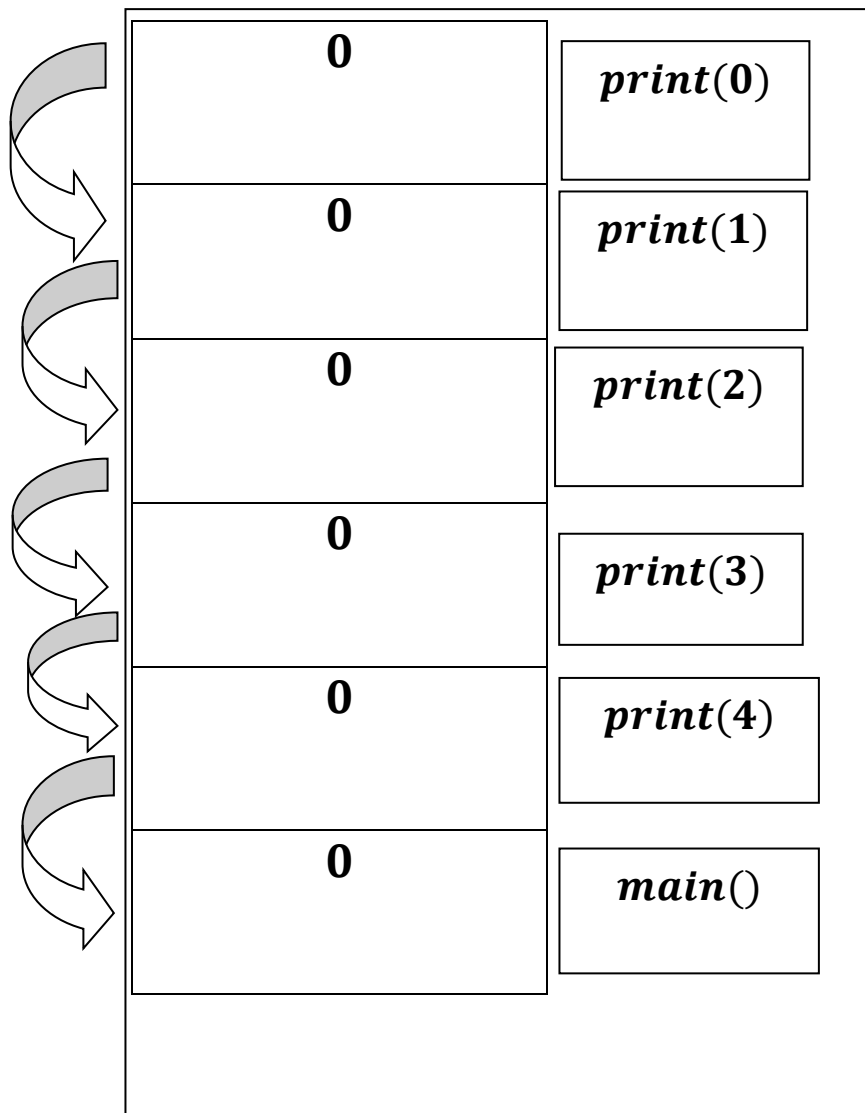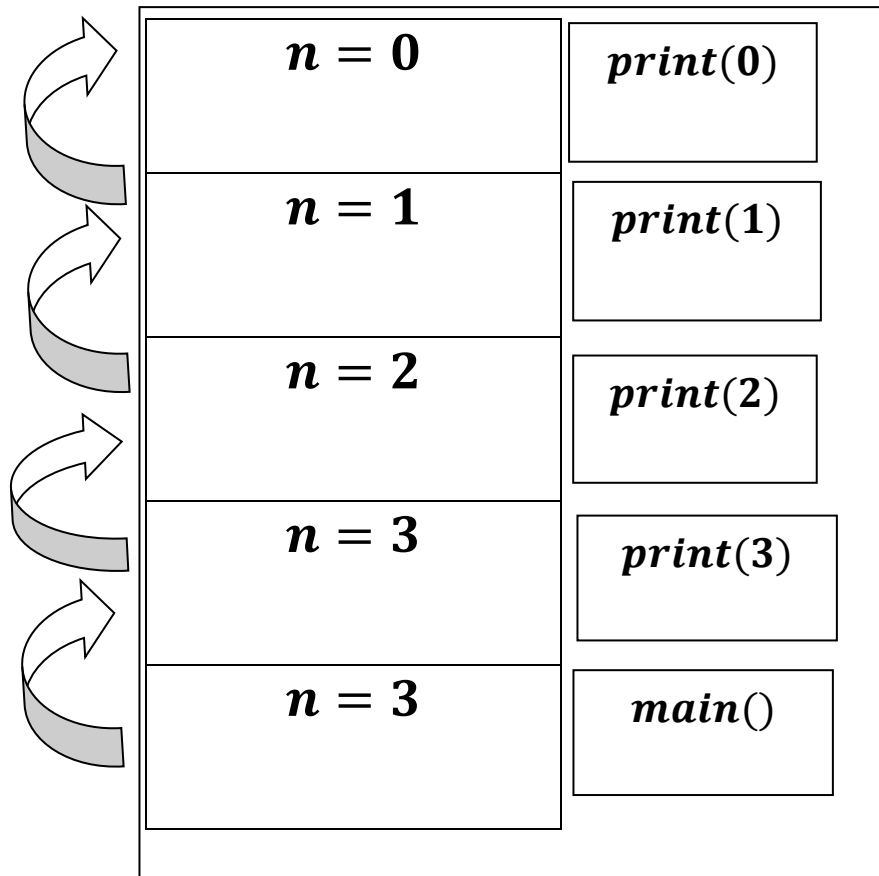
let n = 4,

Push Operation:

$n = 0$	$print(0)$	
$n = 1$	$print(1)$	
$n = 2$	$print(2)$	
$n = 3$	$print(3)$	
$n = 3$	$main()$	

Pop Operation:

0	$print(0)$
0 + 1 = 1	$print(1)$
1 + 1 = 2	$print(2)$
1 + 2 = 3	$print(3)$
3	$main()$

****