(i) Recursion → Function calling itself

Fun calling itself

A() { B() ← method calling }

A() { A() ← method calling itself }

(i) function calling itself

① PMI → Mathematical proof / → using pmi

$\sum_{i=1}^{i=N} i = 1+2+3+4+\dots+N = \frac{n\times(n+1)}{2}$

① Smaller input ② Assume n=k correct ③ n=k+1 Proof

$\sum_{i=1}^{i=1} i = 1 = 1\times\frac{(1+1)}{2}$ | $\sum_{i=1}^{i=2} i = 1+2 = \frac{2\times(2+1)}{2}$

$\sum_{i=1}^{i=K} i = 1+2+3+4+\dots K = \frac{K\times(K+1)}{2}$

$\sum_{i=1}^{i=K+1} i = 1+2+3+4+\dots+K + K+1 = \frac{(K+1)\times(K+2)}{2}$

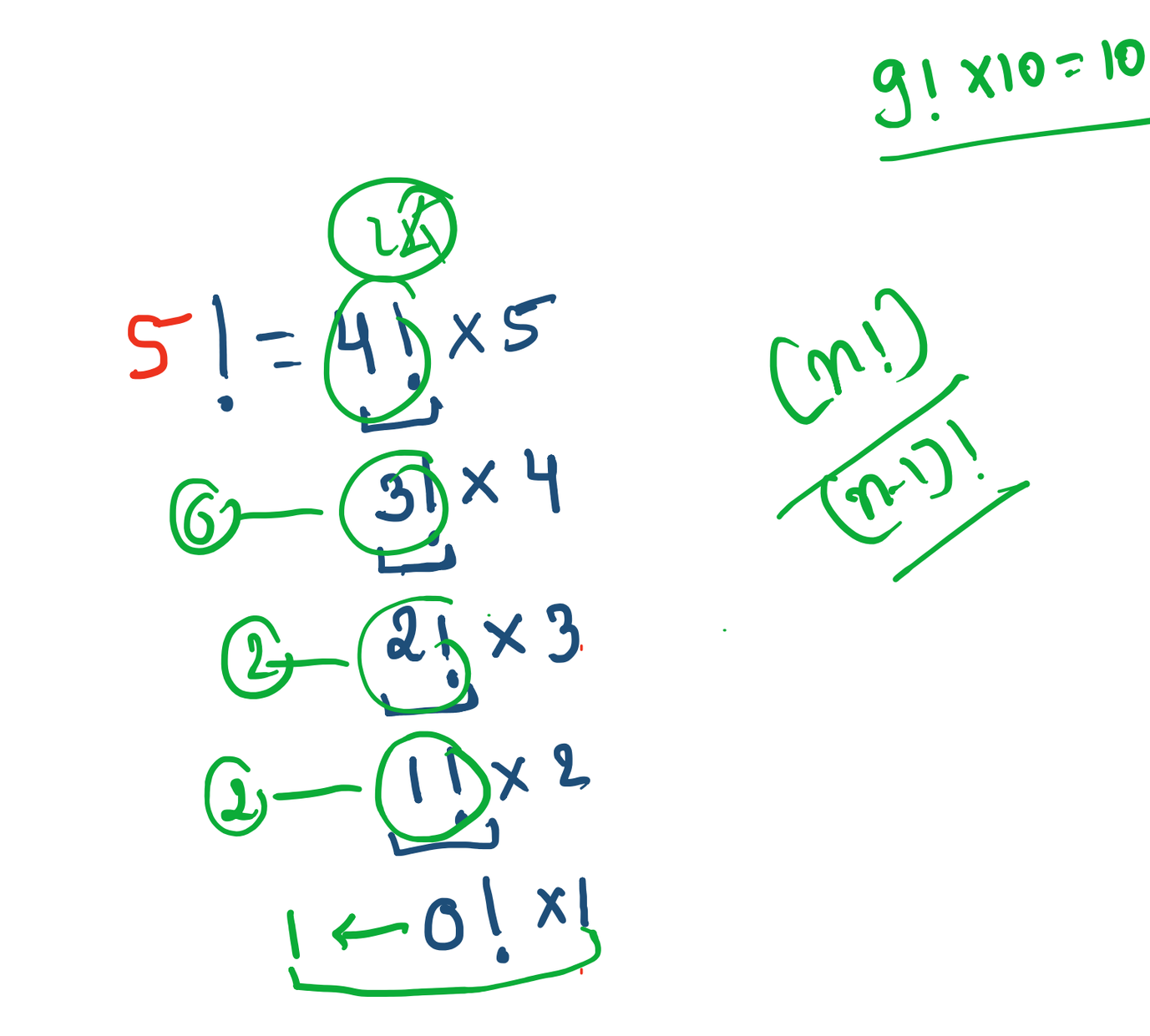
$\frac{K\times(K+1)}{2} + K+1 = (K+1)(\frac{K}{2}+1)$ → $\frac{(K+1)(K+2)}{2}$

(iii) Base → case

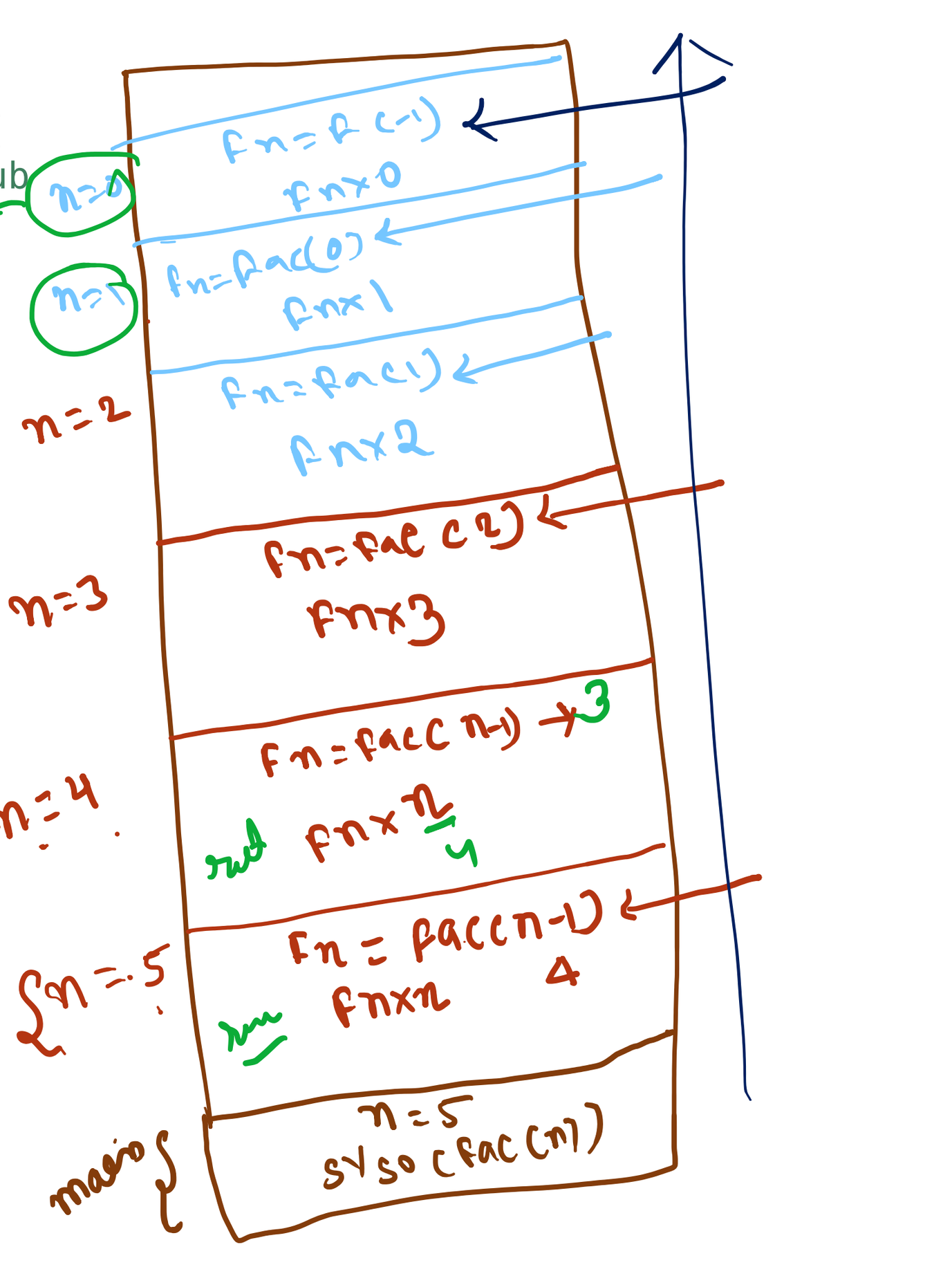
① 9! ② S.P selfwork B.P

Recursion → Recursion पहला प्यार है

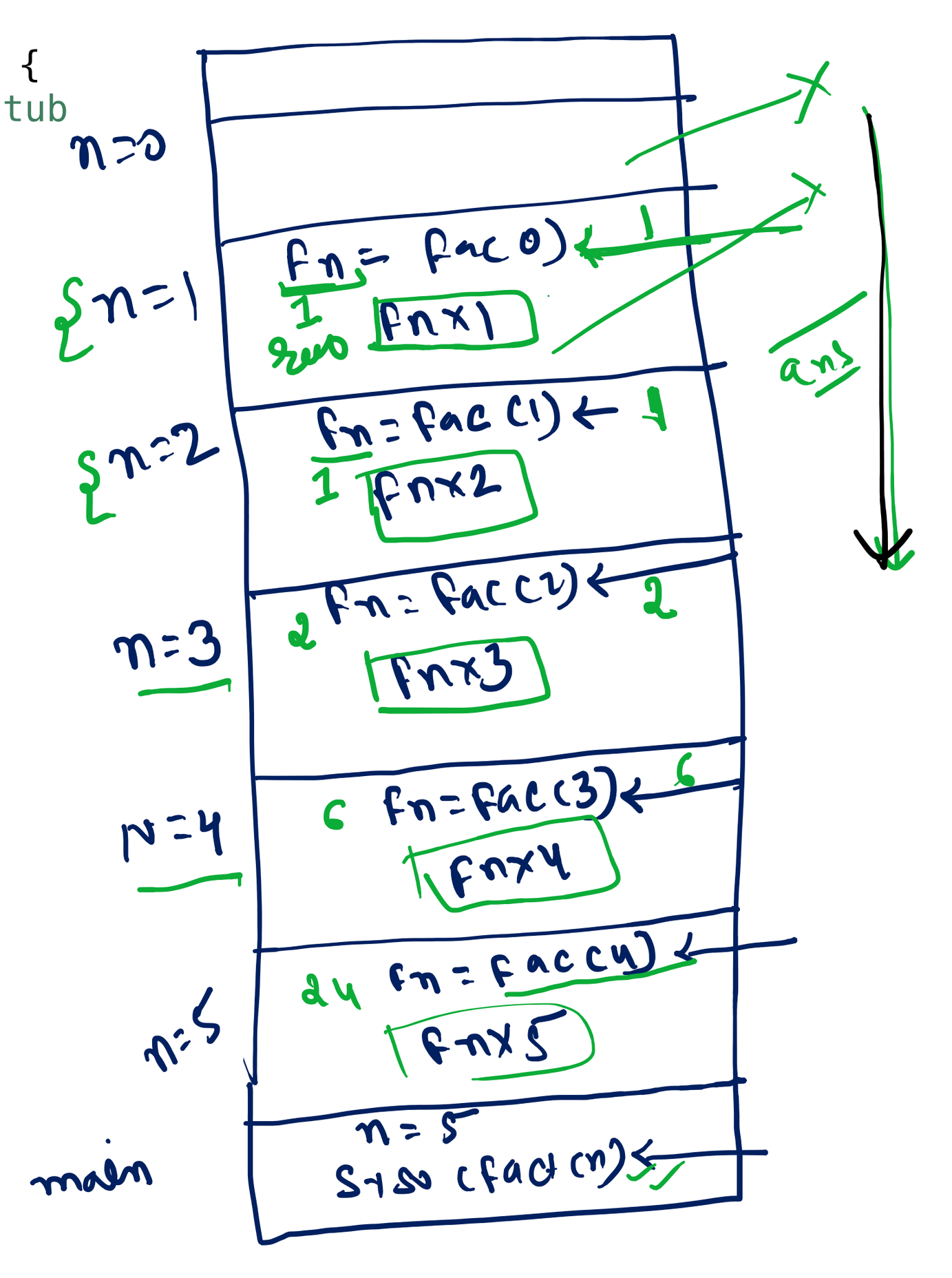
9! x10 = 10!



```java
public static void main(String[] args) {
	// TODO Auto-generated method stub
	int n = 5;
	System.out.println(fact(n));

}

public static int fact(int n) {

	int fn = fact(n - 1);// sp
	return fn * n;
}
```

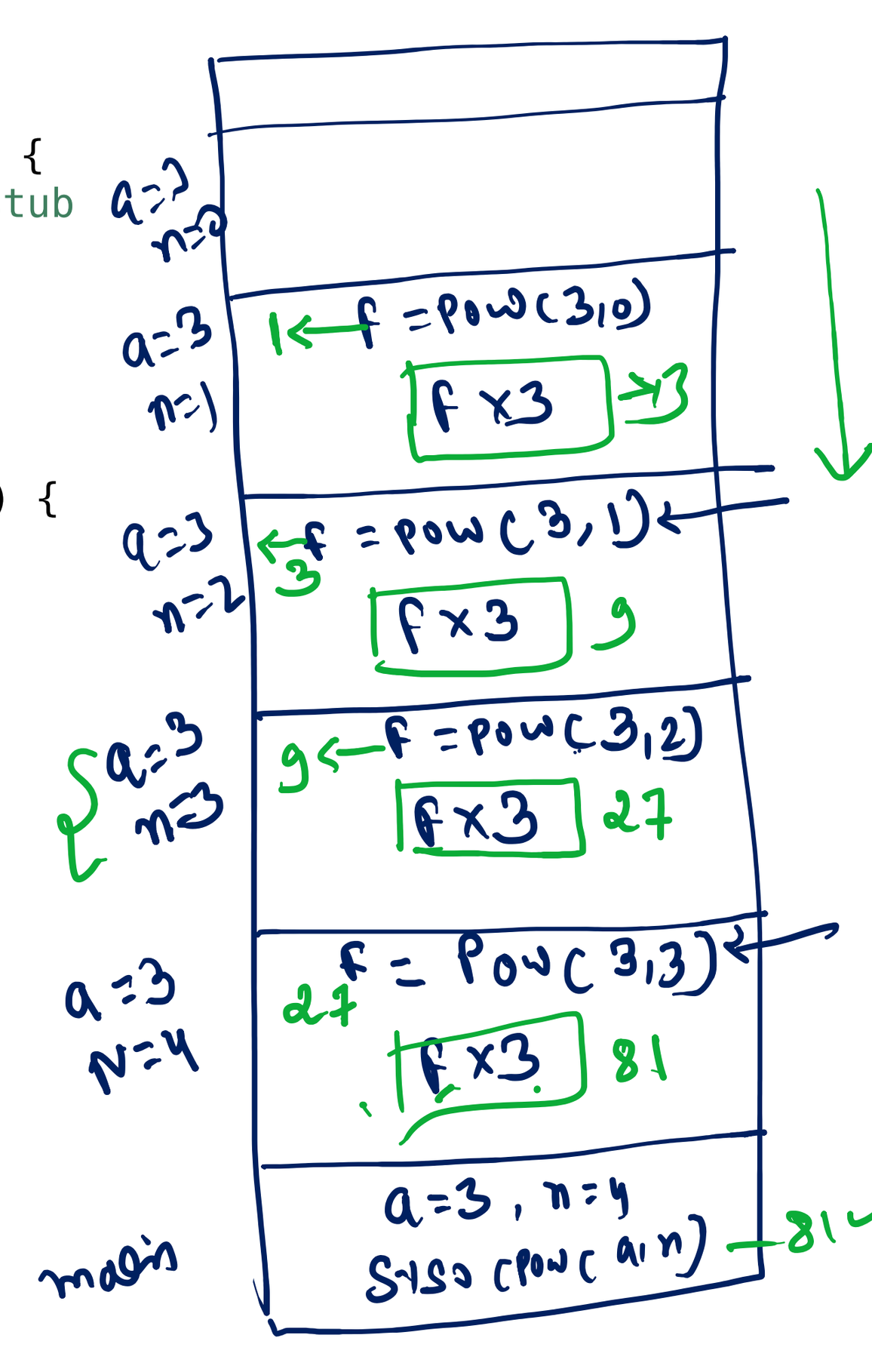


```java
public static void main(String[] args) {
	// TODO Auto-generated method stub
	int n = 5;
	System.out.println(fact(n));

}

public static int fact(int n) {
	// Base case
	if (n == 0) {
		return 1;
	}
	int fn = fact(n - 1);// sp
	return fn * n;
}
```



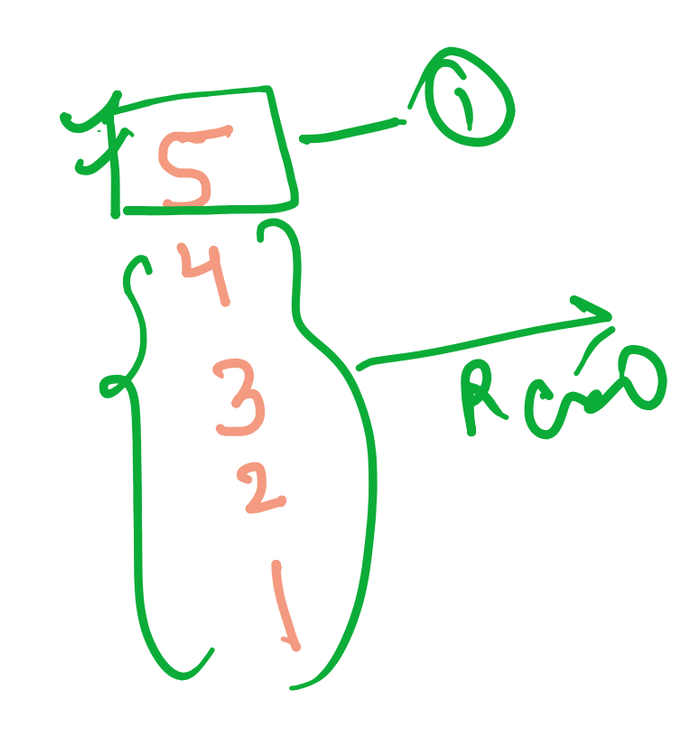
(3^10) → (3^5) ... (a,n) → smaller problem (a,n-1); (a^n) = a×a×a×...×a (n time)

```java
public static void main(String[] args) {
	// TODO Auto-generated method stub
	int a = 3;
	int n = 4;
	System.out.println(pow(a, n));

}

public static int pow(int a, int n) {
	if(n==0) {
		return 1;
	}

	int f = pow(a, n - 1);// sp
	return f * a;
}
```

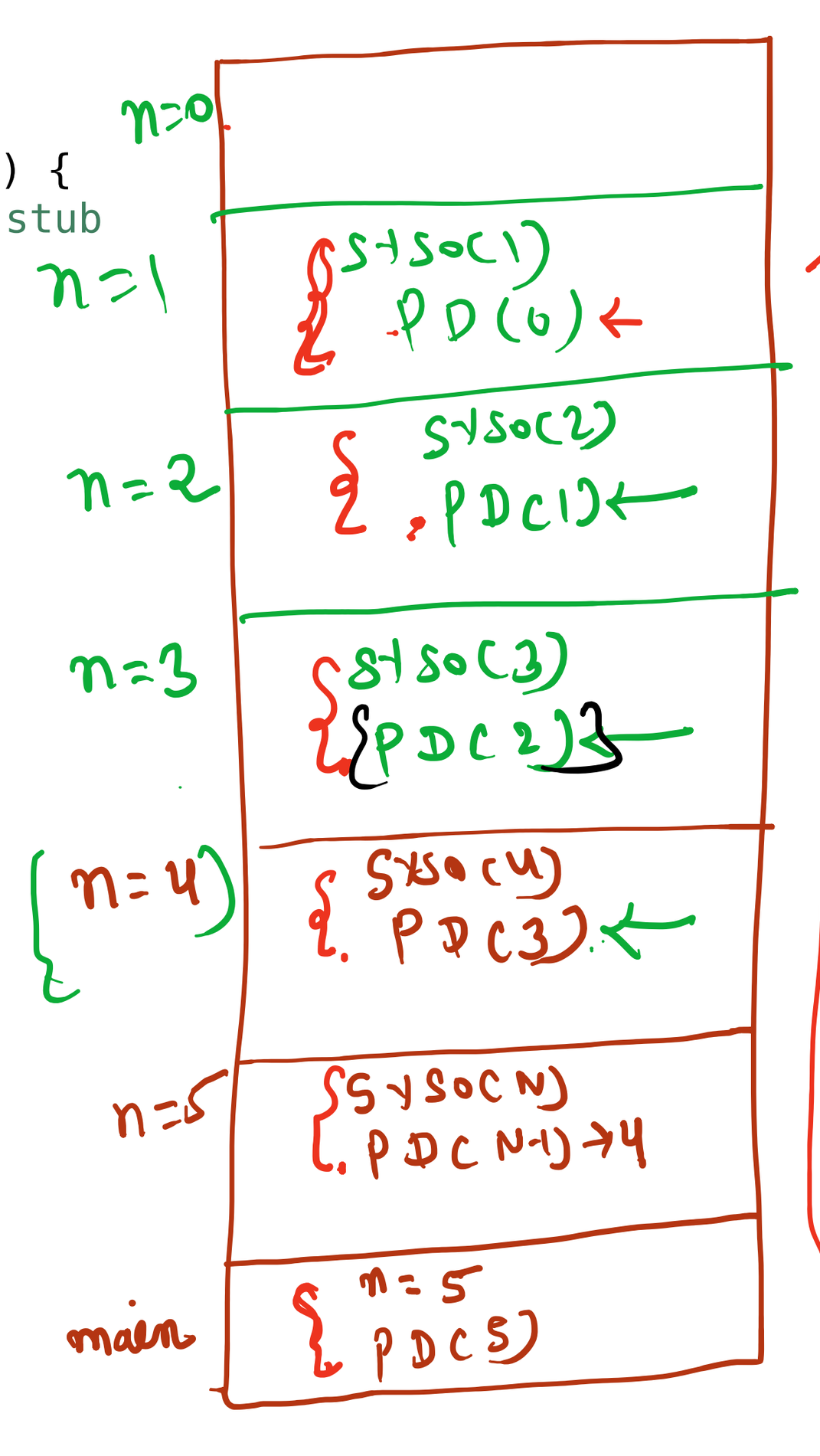


public static void PD(int n)

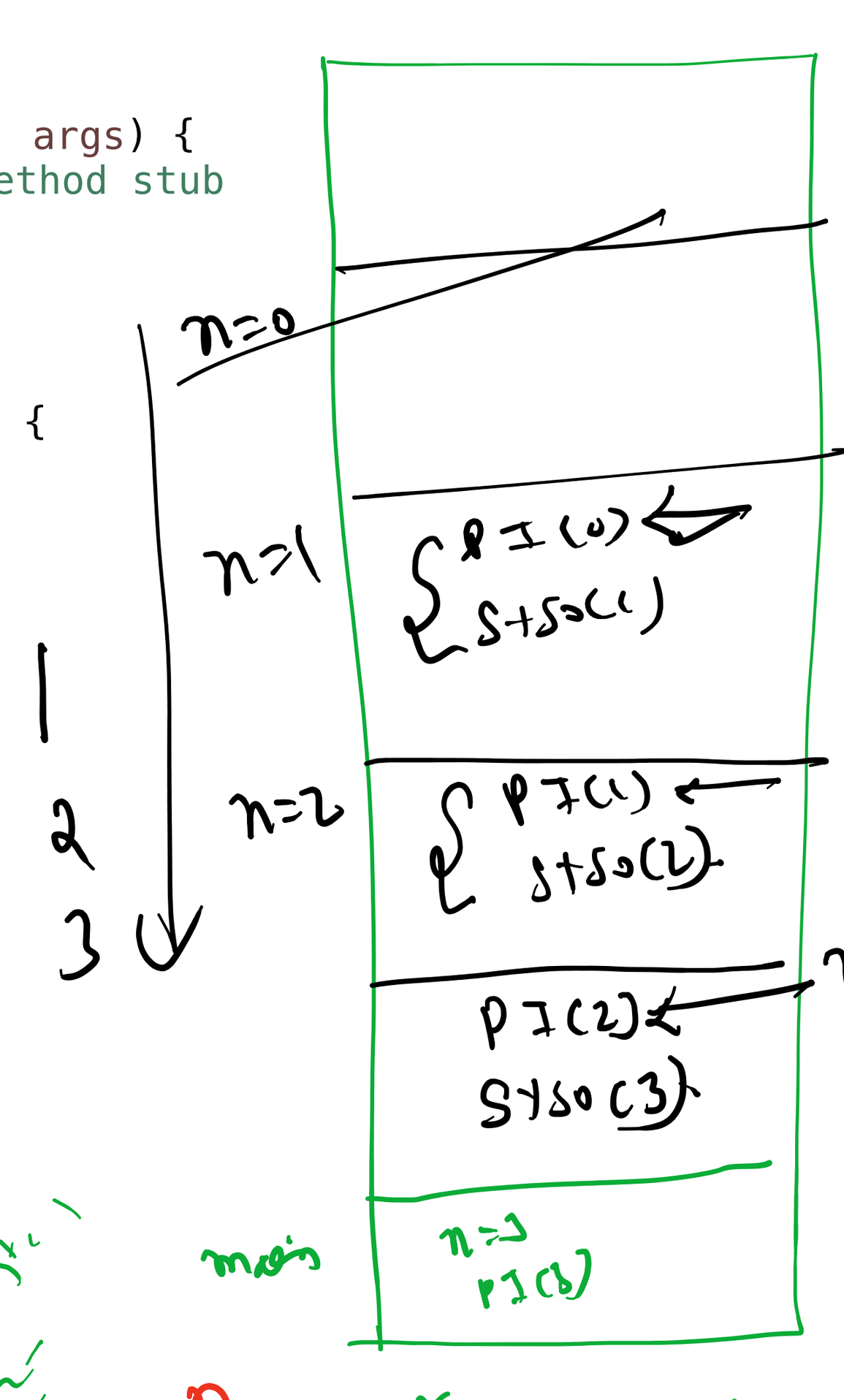
N=5



```java
public static void main(String[] args) {
	// TODO Auto-generated method stub
	int n = 5;
	PD(n);

}

public static void PD(int n) {
	if (n == 0) {
		return;
	}

	System.out.println(n);
	PD(n - 1);
}
```

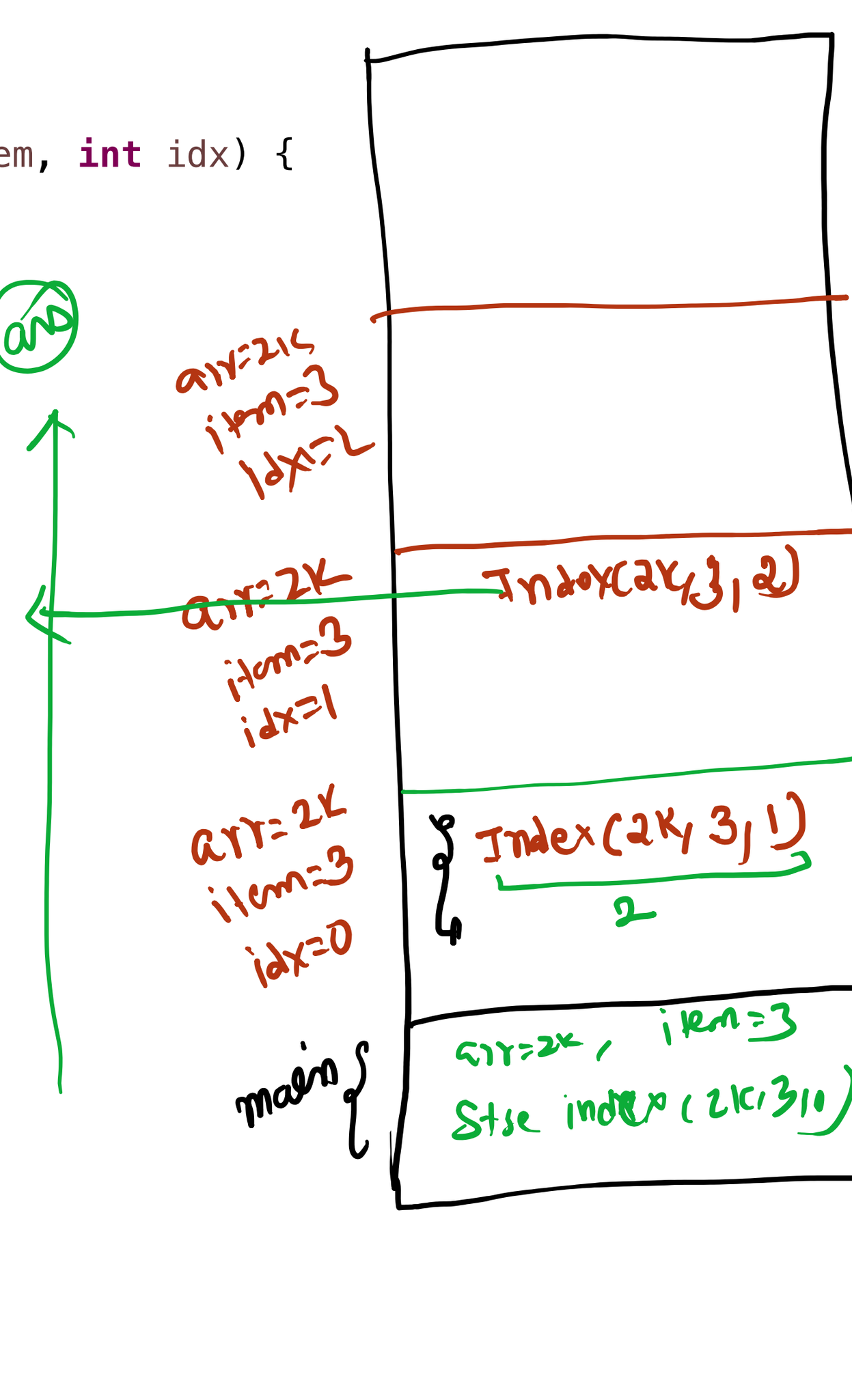


```java
public static void main(String[] args) {
	// TODO Auto-generated method stub
	int n = 5;
	PI(n);

}

public static void PI(int n) {

	if (n == 0) {
		return;
	}
	PI(n - 1);
	System.out.println(n);
}
```



```java
int[] arr = { 1, 2, 5, 4, 3, 4, 7, 4, 3, 6 };
int item = 4;
```

```java
public static int Index(int[] arr, int item, int idx) {
	if (idx == arr.length) {
		return -1;
	}

	if (arr[idx] == item) {
		return idx;
	}
	return Index(arr, item, idx + 1);
}
```



Stack se upar (Stack zero) ↑, Stack se down ↓, Tail / head

5! = 5×4×3×2×1

```java
public static void main(String[] args) {
	// TODO Auto-generated method stub
	int n = 5;
	System.out.println(fact(n, 1));

}

public static int fact(int n, int ans) {
	// Base case
	if (n == 0) {
		return ans;
	}

	return fact(n - 1, ans * n);

}
```