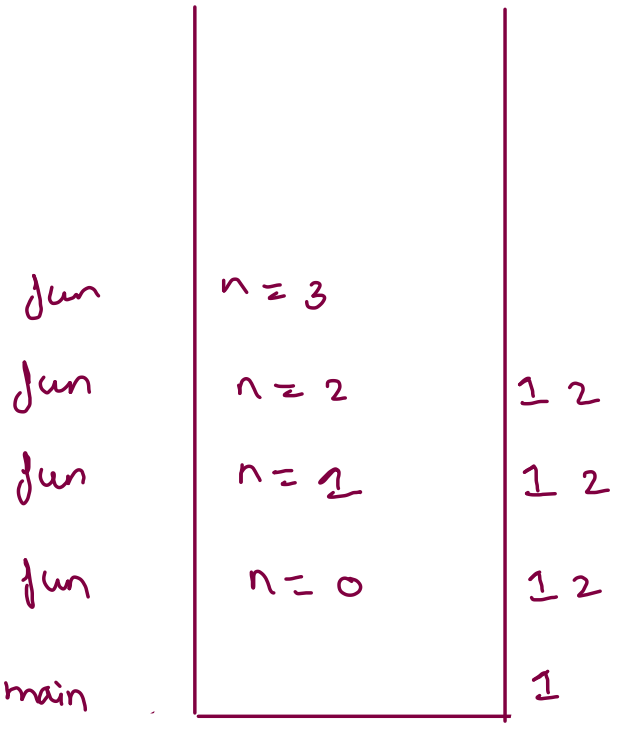
```
p s void fun(int n) {
    1. syso(n);
    2. fun(n+1);
}

p s void main( ) {
    1. fun(0);
}
```



PMI principal of mathematical Induction $n=k$ $k=1$

$$\sum_{i=1}^{n} i = \frac{n(n+1)}{2} \quad -(1) \qquad n \in N$$


✓ (i) assume (1) is true $n=k$

✓ (ii) prove (1) is true $n=k+1$

✓ (iii) prove (1) for $n=1$

$$1 = \frac{1(1+1)}{2}$$

$$1 = 1$$

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

(ii) $$\sum_{i=1}^{k+1} i = \frac{(k+1)(k+2)}{2}$$

$$\underbrace{1+2+3+4+\cdots+k}_{\frac{k(k+1)}{2}} + k+1 = \frac{(k+1)(k+2)}{2}$$

$$\frac{k(k+1)}{2} + k+1$$

$$= \frac{k(k+1) + 2(k+1)}{2} = \frac{(k+1)(k+2)}{2}$$

factorial (5) = 5 * 4 * 3 * 2 * 1

factorial (4) = 4 * 3 * 2 * 1

n = 4

factorial(5) = 5 * factorial(4)

n = 4+1

n = 4

expectation  factorial(n) = n * n-1 * n-2 .... 1

faith  factorial(n-1) = n-1 * n-2 * n-3 .... 1

factorial(n) = n * factorial(n-1);

1. decide expectation
2. freeze faith (assume 'just smaller')
3. meet your expectation using faith

expectation  pd(n) → n  n-1  n-2 ..... 1

faith  pd(n-1) → n-1  n-2 .... 1

pd(n) → syso(n) + pd(n-1)

n n-1 n-2 ... 1 → n  n-1 n-2 ... 1

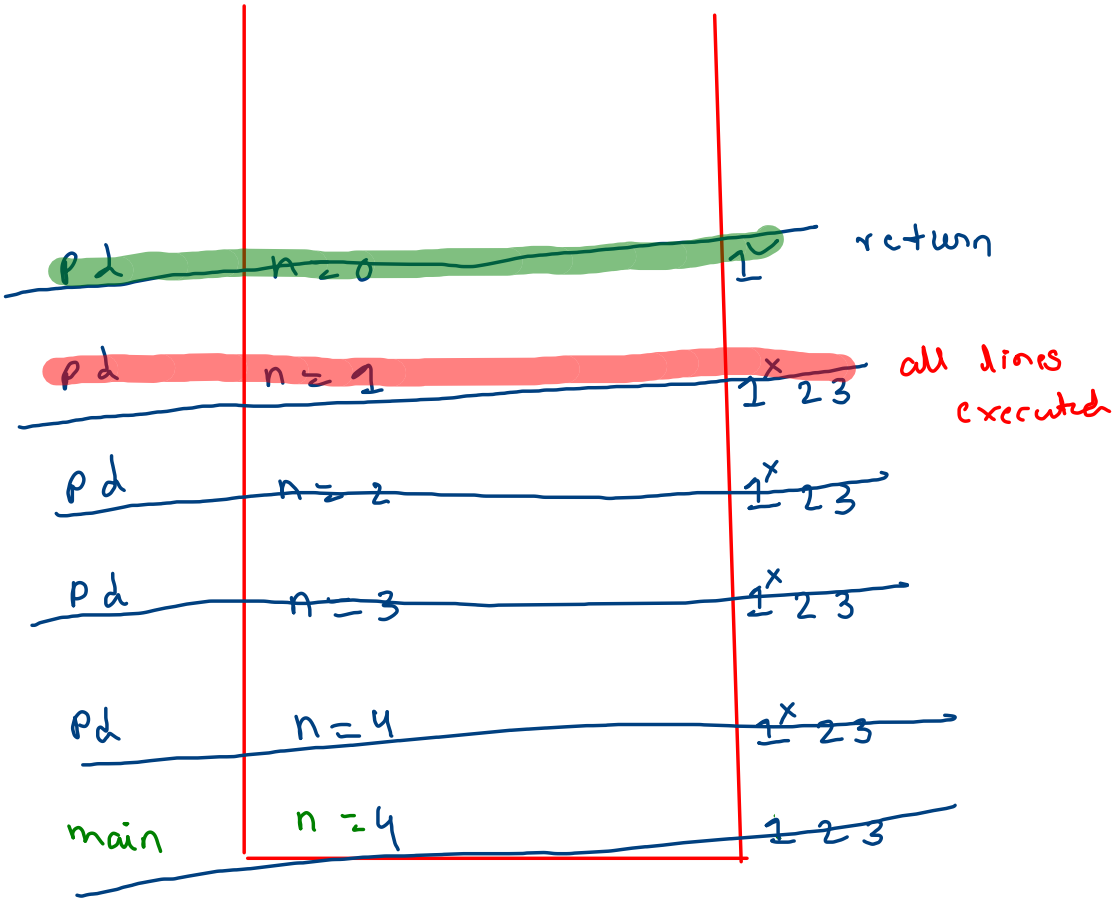
```java
public static void main(String[] args) throws Exception {
    // write your code here
1   Scanner scn = new Scanner(System.in);
2   int n = scn.nextInt();

3   printDecreasing(n); // expectation -> n n-1 n-2...1
}

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

2   System.out.println(n);

    //faith -> n-1 n-2 n-3...1
3   printDecreasing(n-1);
}
```

4

3

2

1



exrcetation  pi(n) ⟶ 1 2 3 .... n-1 n

Jaith  pi(n-1) ⟶ 1 2 3 .. .. n-1

pi(n) ⟶ pi(n-1) + syso(n)

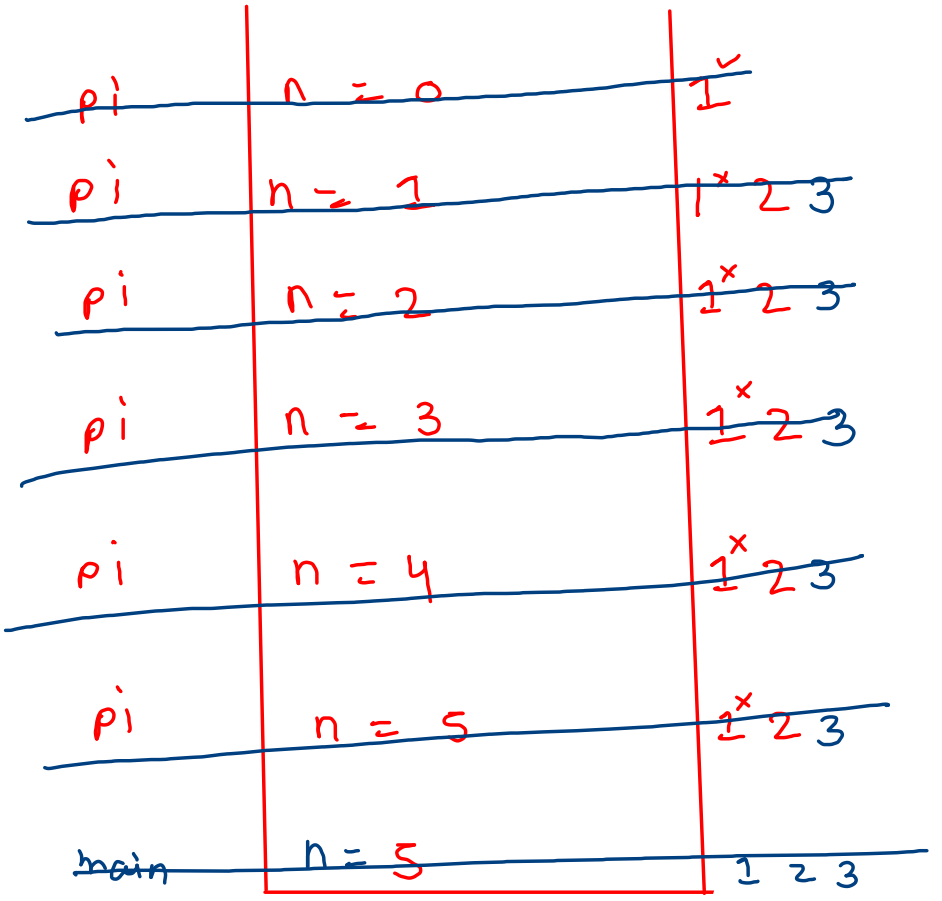
1 2 ... n-1 n ⟶ 1 2 .... n-1 n

pi(5) ⟶ pi(4) + syso(n);

1 2 3 4 5

```java
public static void main(String[] args) throws Exception {
    // write your code here
    Scanner scn = new Scanner(System.in);
    int n = scn.nextInt();

    printIncreasing(n); // expectation -> 1 2 3 4...n-1 n
}

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

    //faith -> 1 2 3 4...n-1
    printIncreasing(n-1);

    System.out.println(n);
}
```

1
2
3
4
5



print dec_inc

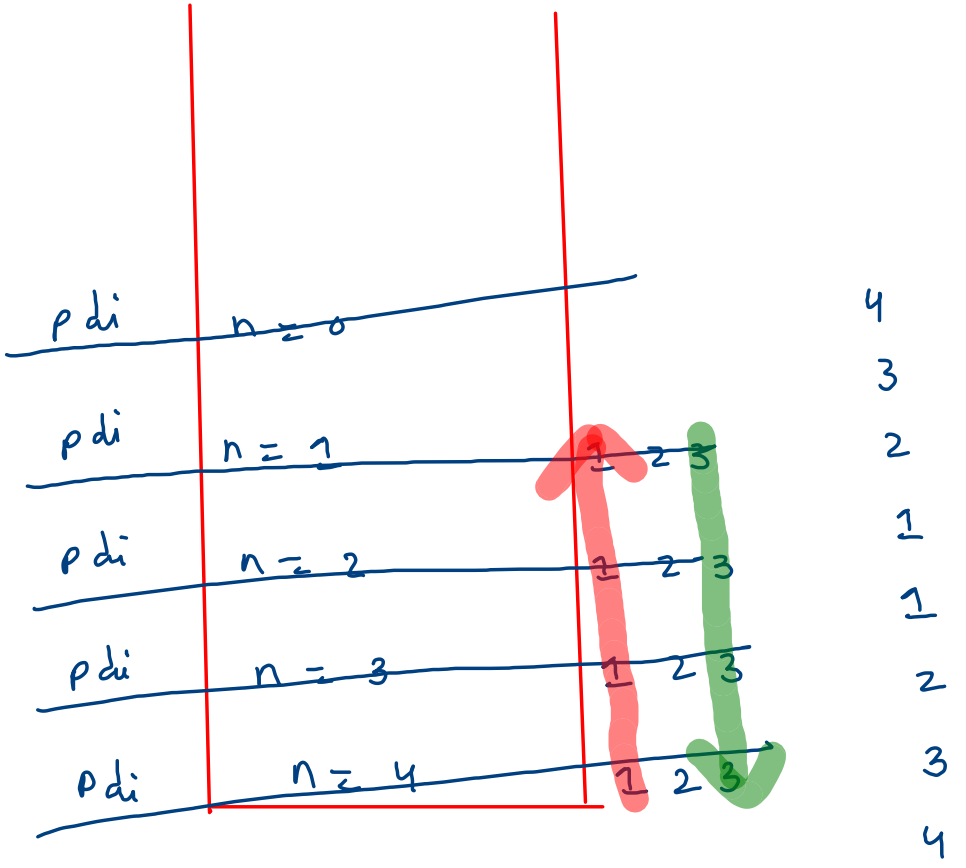
pdi(4) → 4 3 2 1 1 2 3 4

pdi(3) → 3 2 1 1 2 3

pdi(4) → sysout(n) + pdi(3) + sysout(n)

4 3 2 1 1 2 3 4

```java
public static void pdi(int n){
    if(n == 0) {
        return;
    }

    System.out.println(n);

    pdi(n-1);

    System.out.println(n);
}
```



expect: factorial(5) ⟶ 5 * 4 * 3 * 2 * 1

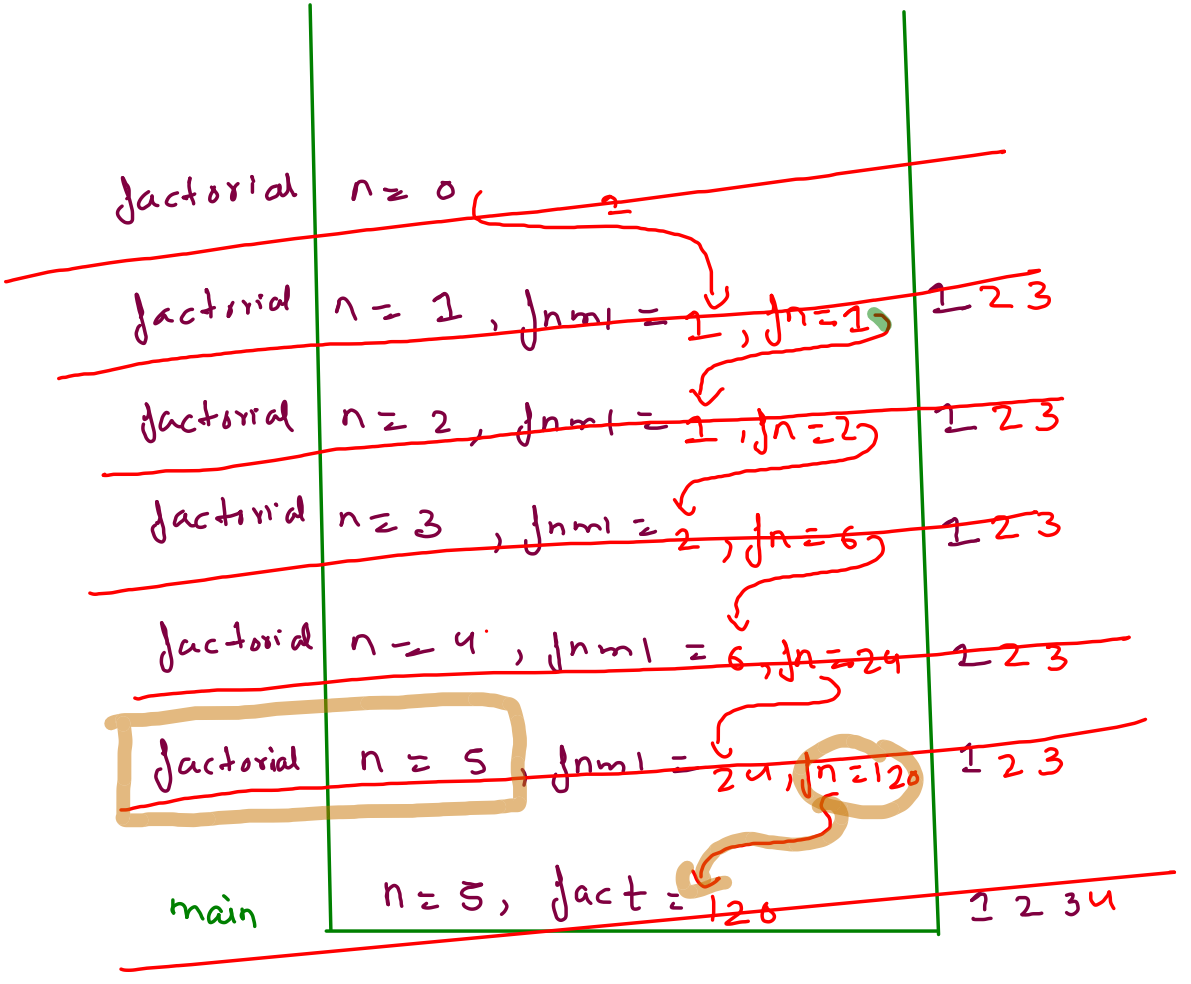
faith: factorial(4) ⟶ 4 * 3 * 2 * 1

factorial(n) = n * factorial(n-1);

n * n-1 * n-2 ... 1 ⟶ n * n-1 * n-2 * ... 1

```java
public static void main(String[] args) throws Exception {
    // write your code here
1   Scanner scn = new Scanner(System.in);
2   int n = scn.nextInt();

3   int fact = factorial(n);
4   System.out.println(fact);
}

public static int factorial(int n){
    if(n == 0) {
        //0! = 1
        return 1;
    }

1   int fnm1 = factorial(n-1); //factorial of n-1
2   int fn = n * fnm1; //factorial of n

3   return fn;
}
```

120



expectation  $power(x, n) = x^n$

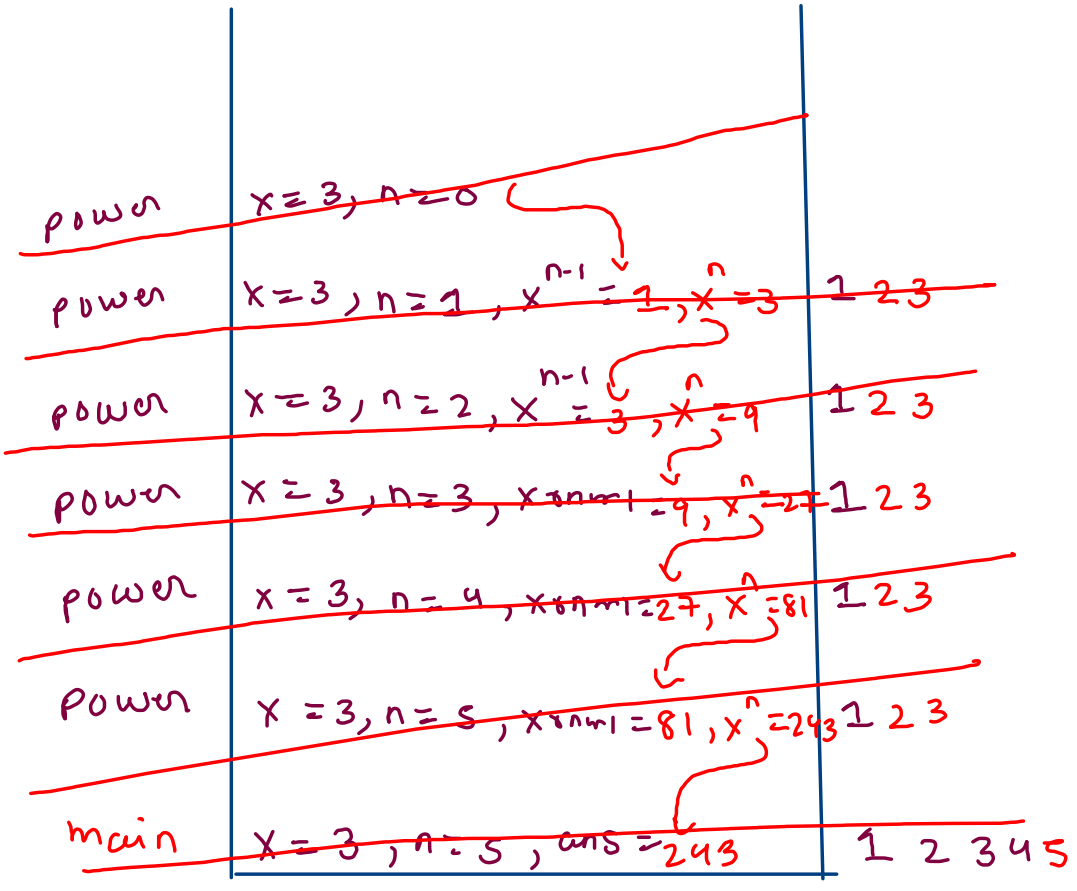
Jaikh  $pow(x, n-1) = x^{n-1}$

$pow(x, n) = x * pow(x, n-1)$

$x^n = x * x^{n-1}$

```java
public static void main(String[] args) throws Exception {
    // write your code here
1   Scanner scn = new Scanner(System.in);
2   int x = scn.nextInt();
3   int n = scn.nextInt();

4   int ans = power(x,n);
5   System.out.println(ans);
}

public static int power(int x, int n){
    if(n == 0) {
        return 1;
    }

1   int xnm1 = power(x,n-1);
2   int xn = x * xnm1;

3   return xn;
}
```

$$3^5 = 243$$



$2^{16}$

$\downarrow$

$X^n = X * X^{n-1}$

$2^{15}$

$\downarrow$

$2^{14}$

$\downarrow$

$2^{13}$

$\downarrow$

$2^{12}$

$\downarrow$

$2^{11} \to 2^{10} \to 2^9 \to 2^8 \to 2^7 \to 2^6 \to 2^5 \to 2^4 \to 2^3 \to 2^2 \to 2^1 \to 2^0$

$$\begin{cases} x^{\frac{n}{2}} * x^{\frac{n}{2}} & n \text{ is even} \\ x^{\frac{n}{2}} * x^{\frac{n}{2}} * x & n \text{ is odd} \end{cases}$$

```
int xrn = pow(x, n/2) * pow(x, n/2)
if(n%2==1){
    xrn = xrn * x;
}
```

=

```
int xrnb2 = pow(x, n/2);
int xrn = xrnb2 * xrnb2;
if(n%2==1){
    xrn = xrn * x;
}
```

Algn

```java
public static int power(int x, int n){
    if(n == 0) {
        return 1;
    }

1   int xrnb2 = power(x,n/2);
2   int xrn = xrnb2 * xrnb2;

    if(n % 2 == 1) {
3       //n is odd
        xrn = xrn * x;
    }

4   return xrn;
}
```

$x = 2,\ n = 20$

$2^{20}$

$\downarrow$

$2^{10}$

$\downarrow$

$2^5 \rightarrow 2^2 \rightarrow 2^1 \rightarrow 2^0$

power $x = 2,\ n = 20,\ x^{\frac{n}{2}} = 1024,\ x^n = 2^{20}$ 1234

power $x = 2,\ n = 10,\ x^{\frac{n}{2}} = 32,\ x^n = 1024$ 1234

power $x = 2,\ n = 5,\ x^{\frac{n}{2}} = 4,\ x^n = 32$ 1234

power $x = 2,\ n = 2,\ x^{\frac{n}{2}} = 2,\ x^n = 4$ 1234

power $x = 2,\ n = 1,\ x^{\frac{n}{2}} = 1,\ x^n = 2$ 1234

power $x = 2,\ n = 0$ 1234

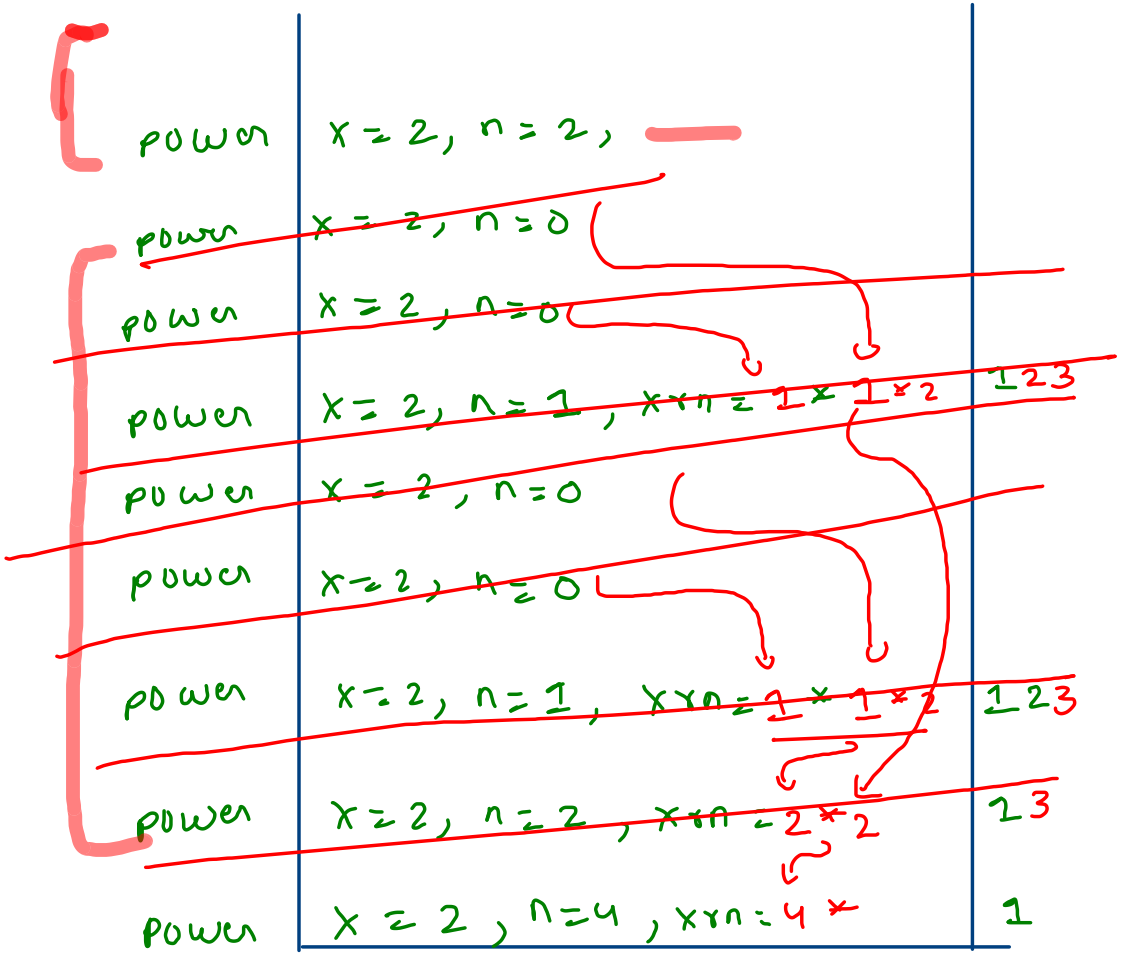
$1024 \times 1024$

```java
public static int power(int x, int n){
    if(n == 0) {
        return 1;
    }

    int xrn = power(x,n/2) * power(x,n/2);

    if(n % 2 == 1) {
        //n is odd
        xrn = xrn * x;
    }

    return xrn;
}
```

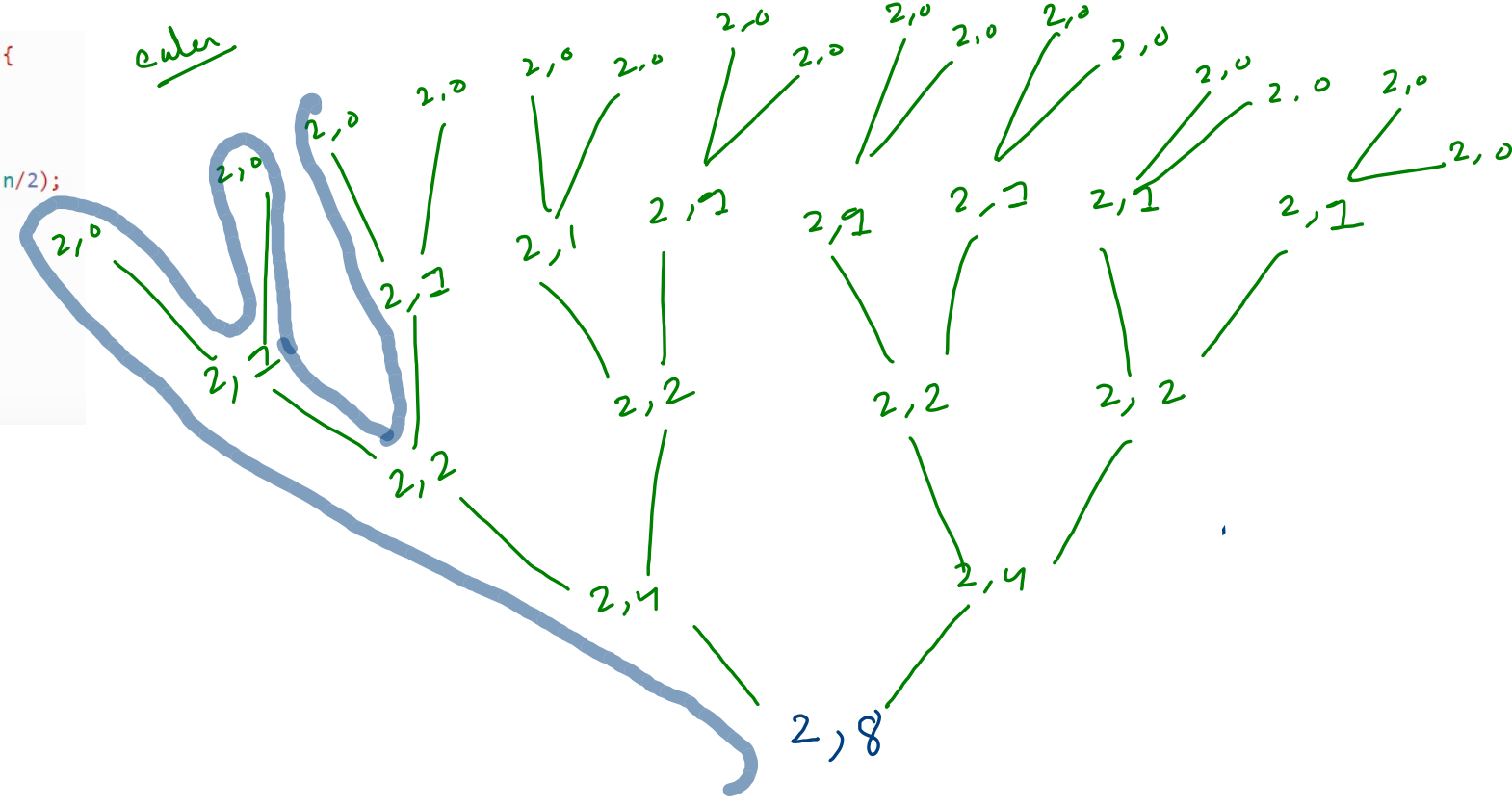
1
2
3

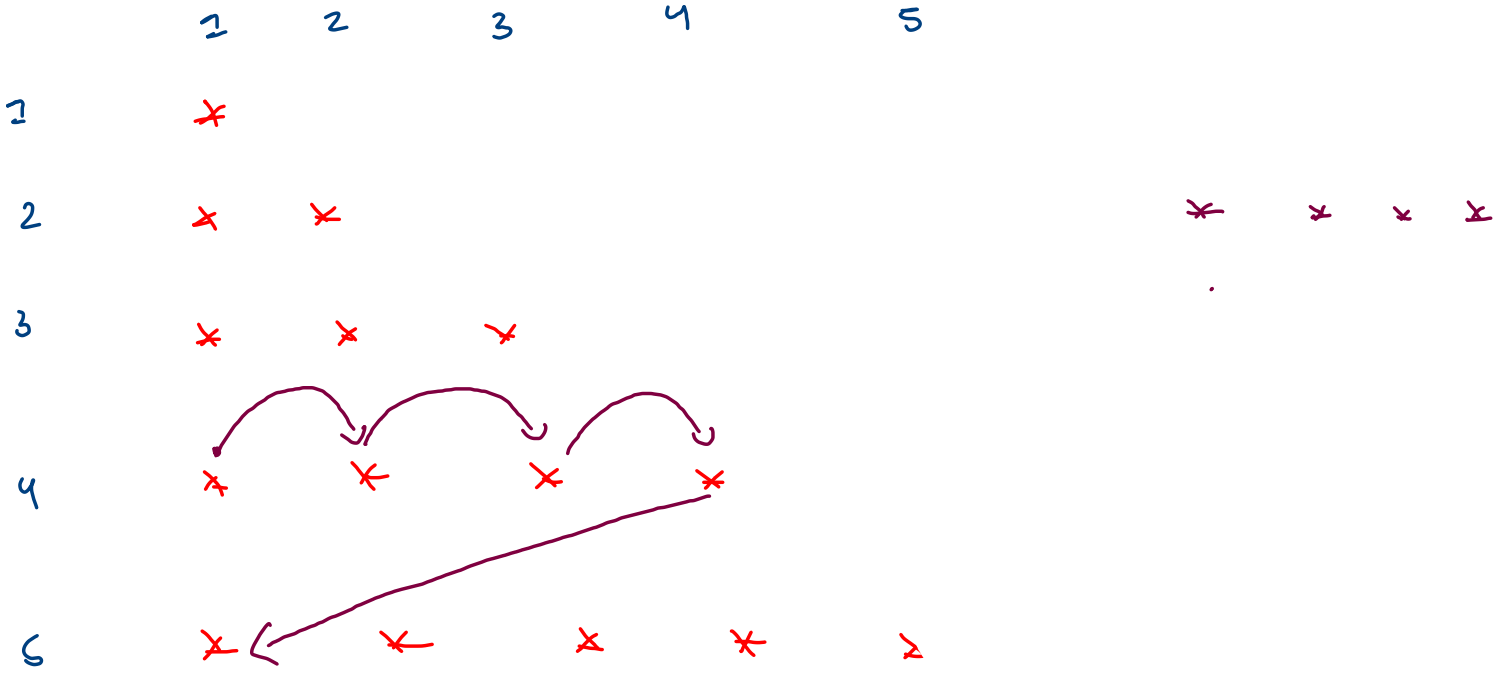
$2^4$

| power-linear | power-log | power-log fake |
|---|---|---|
| 4 | 2 | very bad |



```java
public static int power(int x, int n){
    if(n == 0) {
        return 1;
    }

    int xrn = power(x,n/2) * power(x,n/2);

    if(n % 2 == 1) {
        //n is odd
        xrn = xrn * x;
    }

    return xrn;
}
```




1 2 3 4 5
1
2
3
4
5