# Functions

$f(x) = x+5;$

input = 2 output = 7

$f(a, b) = a + b;$

input = 2, 3 output = 5

---

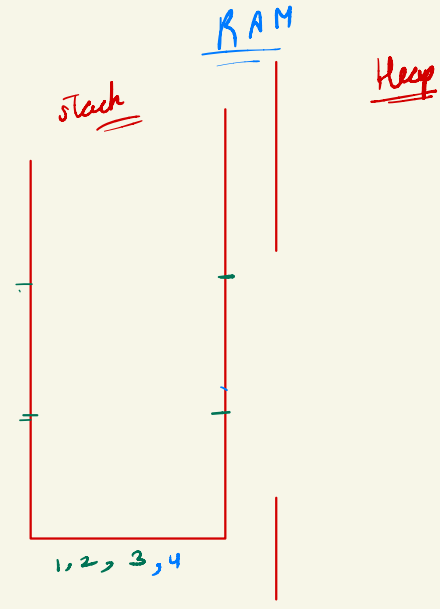
⇒

```
public static int sum(int a, int b){
    int sum = a+b;
    return sum;
}
```

```
public static return_type fun_name (parameters) {
    // logic
}
```

(return_type: void)

```
public class Main {
    public static int sum(int a, int b){
        int sum = a + b;
        return sum;
    }

    public static void main(String[] args) {
        int a=2;
        int b=3;

        int ans=sum(a, b);
        System.out.println(ans);
    }
}
```



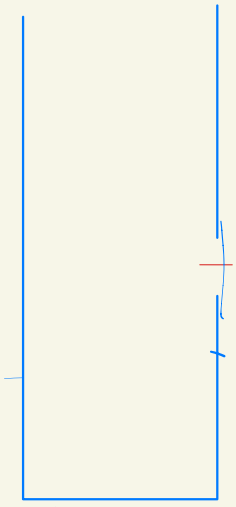

5

```java
public class Main {
    public static void printSomething(){
        System.out.println(x: "Hi!!");
        int a=5;
        a++;
        a--;

        a=11;
    }

    public static int sum(int a, int b){
        printSomething();

        int sum = a + b;
        return sum;
    }

    Run | Debug
    public static void main(String[] args) {
        int a=2;
        int b=3;

        int ans=sum(a, b);
        System.out.println(ans);
    }
}
```

printSomething();

Hi!!

5



# PMI

Principal of Mathematical Induction

sum of first n natural number

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

$n = 0,\ n = 1$ → base case

suppose $n = k$ → smaller problem

↳ $n = k+1$ → solving

# Recursion

a function calling itself

1) solve for the smallest problem
2) keep a faith that it will work for a smaller problem. (faith)
3) solve for rest of the problem. (to reach your expectation)

# print decreasing

1) n=1 is the smallest problem
2) n=5, (n-1)
   4

```
if (n==1)
    Sys out (1);
    return;
```

---

Faith (smaller value)

n-1 ⇒ 4

4 3 2 1

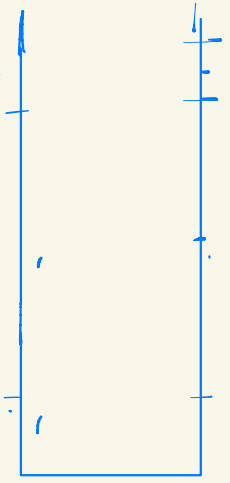
Expectation

If I provide n=5, it will 5 4 3 2 1

very important

10 lines → 2 pages of notebook

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

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



Ques print increasing recursively.

n=5

1) you won't be able to learn recursion if you don't dry run.

Faith

It will print 1 numbers for (n-1);

4 ⇒ 1 2 3 4

```
public static void PI (int n){
    if(n==0) return;
    1) PI (n-1);
    2) System.out.println(n);
}
```

Expectation

1) my function should print increasing numbers.
2) for n=5, it should print 1 2 3 4 5

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

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

break till ⇒ 10:15

# print decreasing increasing

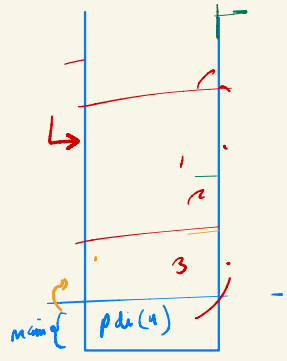
Faith

n-1 n=3

3 2 1 1 2 3

4 3 2 1 1 2 3

```
public static void PDI(int n){
    if(n==0) return;

    1) System.out.println(n);
    2) PDI(n-1);
    3) System.out.println(n);
}
```




Expectation

It will print numbers in decreasing, then increasing.

n=4

4 3 2 1 1 2 3 4

# Find factorial of a number

**Faith**

it will give me fac(n-1);

4 X 3 X 2 X 1

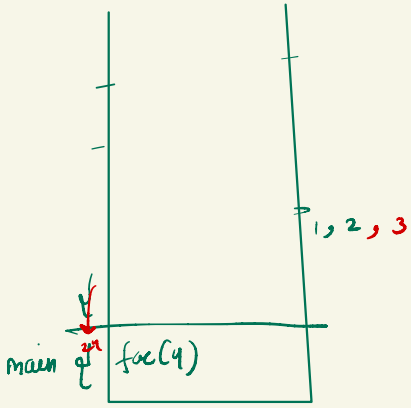
```
public static int fac(int n){
1  if(n==1) return 1
2  int sAns = fac(n-1);
3  int ans= n * sAns;
4  return ans;
}
```

**Expectation**

My function will return factorial of n

n = 5

5 X 4 X 3 X 2 X 1




ans = 2 X 1 ⇒ 2

ans = 3 X 2 = 6

ans = 4 X 6 ⇒ 24

---

**Ques** Calculate x raised to power n. **Recursively**

**Faith**

My function will calculate $x^{n-1}$ perfectly

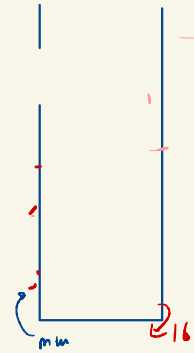
$2^4 \Rightarrow 2 \times 2 \times 2 \times 2$

```
public static int power(int x, int n){
    if(n==0) return 1;
1)  int sAns = power(x,n-1);
2)  int ans = x * sAns;
3)  return ans;
}
```

**Expectation**

My function will calculate x raised to power $x^n$

2 X 2 X 2 X 2 X 2




ans = x * 1 = 2 * 1

ans = 2 * 2 = 4

ans = 2 * 4 = 8

ans = 2 * 8 = 16

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

2) int sAns = power(x,n-1);
3) int ans = x * sAns;

4) return ans;
}
```

psvi power (int x, int n) {

int sAns = power(x, n/2);
if (n%2 == 0)
ans = sAns * sAns
else
ans = x * sAns * sAns
}

Faith — My function will calculate $x^{\frac{n}{2}}$ perfectly.

$3^{\frac{6}{2}} \Rightarrow 3^3$
sAns = 3×3×3
ans = sAns * sAns

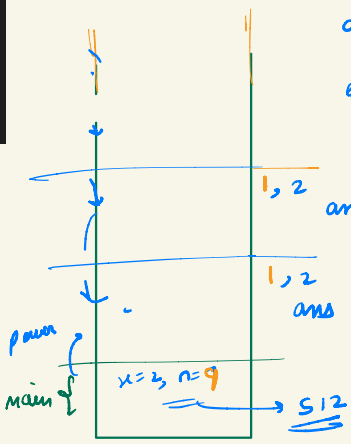
$x^{\frac{n}{2}} \Rightarrow 3^{\frac{1}{2}} \Rightarrow 3^3$
sAns = 3×3×3
ans = sAns × sAns × x

Expectation — My function will calculate $x^n$.

$3^6$ ans ⇒ (3×3×3) × (3×3×3)  (sAns, sAns)

$3^1$ ans ⇒ (3×3×3) × (3×3×3) × 3  (sAns, sAns, x)

```java
public static int power_log(int x, int n){
    if(n==0) return 1;
1) int sAns = power_log(x, n/2);
                          3, 5
2) int ans=0;
   if(n%2==0){ // if n is even
       ans = sAns * sAns;
   } else { // if n is odd
       ans = sAns * sAns * x;
   }

3) return ans;
}
```




ans = 1 * 1 × 2 ⇒ 2
ans = 2 * 2 ⇒ 4
$a^0 \Rightarrow 1$
ans = 4 * 4
ans = 16 * 16 * 2 = 512

on schedule
1 class

10:10 break

recursion in string

$\frac{n}{2^k} = 1$ → solve this for k.

Ques Sum of first n natural numbers recursively.

Faith

(n-1)

1+2+3+4

```java
// sum of first n natural numbers
public static int sum_natural(int n){
    if(n==0){
        return 0;
    }

    int sAns = sum_natural(n-1);
    int ans = n + sAns;

    return ans;
}
```

Expectation

n = 5

1+2+3+4+5

---

# Recursion in Arrays

Ques Print the array recursively.

Faith

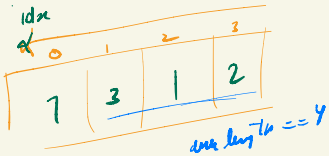
My fun. will print array from idx+1 to last.

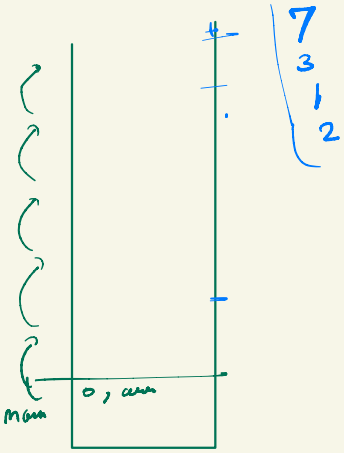
psv print(int idx, int[] arr){
Sysout(arr[idx]);
print(idx+1, arr);

Expectation

My function will print array from idx = 0 to end

3




3
1
2

Recursion on the way up

```java
public static void printArray(int idx, int[] arr){
    if(idx == arr.length){
        return;
    }
    System.out.println(arr[idx]);
    printArray(idx+1, arr);
}
```

→ why?

quality >>> quantity

Java to point

4 hours

15 min

b a b a b

if ( s > e)
return 1;

if ( str[s] != ch[e]
return 0;