* A cute story on recursion :-

A child couldn't sleep, so her mother told a story about a little frog who wouldn't sleep

- So the little frog's mother told a story about a little bear who couldn't sleep
  - So the little bear's mother told a story about a little weasel who fell asleep.
  - .... & the little bear fell asleep.
- ... & the little frog fell asleep.

... & the child fell asleep.

* Principle of mathematical induction :-

→ It is a proving technique in mathematics.

→ It is a three step framework :-

i) Base case :- The smallest problem for which we already know the answer.

ii) Self work :- Something that we will do ourselves.

iii) Assumption :- Task delegation that you will do on someone else & expect it to complete.

Q :- We know that $f(n) = \frac{n(n+1)}{2}$ is the formula to calculate sum of first n natural numbers. Prove that $f(n-1) = \frac{n(n+1)}{2}$ using PMI.

→ Base case :- If $n = 1$

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

→ Assuming the formula works for some value k.

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

If $k = n-1$, then

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

So this is sum of $(n-1)$ natural numbers.

* Recursion :-

→ It is a mechanism in which a function (representing a bigger problem) calls itself again & again with different arguments (representing sub problems) to solve the entire problem.

Q. Write a recursive function to calculate n!

→ $f(n) = n!$

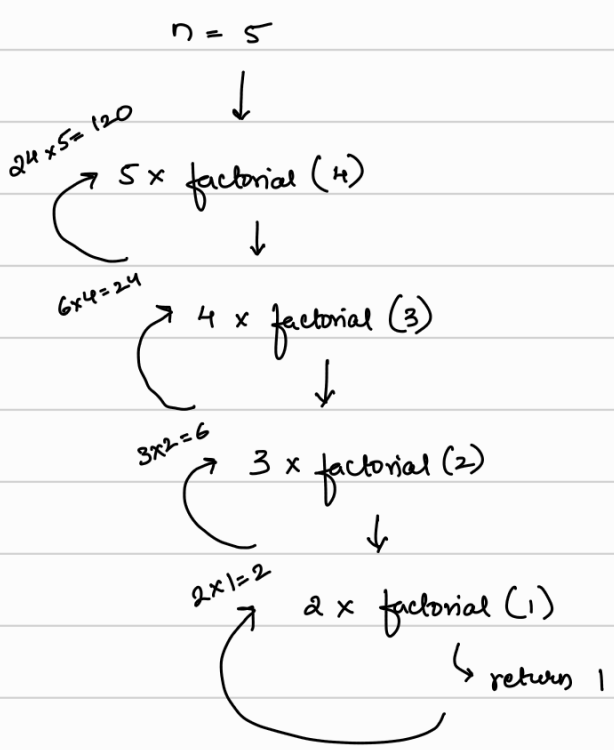
$f(1) = 1$ & $f(0) = 1$ (Base case)

$\therefore f(n) = n \times (n-1) \times (n-2) \times (n-3) \times \dots \times 3 \times 2 \times 1$

$f(n-1) = (n-1)!$ ... $f(1)$

$f(n) = n \times (n-1)!$

```
int factorial (int n) {
  if (n==0 || n==1) {
      return 1;
  }
  return n x factorial (n-1);
}
```



Recurrence relation :- $f(n) = n \times f(n-1)$

Q. Given two integers a & b, write a recursive function to implement a^b.

→ Base case, anything to the power of 0 is 1.

Anything to the power of 1 is the number itself.

a = 4	b = 0	Ans = 1
a = 4	b = 1	Ans = 4
a = 4	b = 2	Ans = 4 × 4 = 16

$f(a^b) = a \times f(a^{b-1})$ → Recurrence relation

```
int power (int a, int b) {
    if (b==0) return 1;
    return a x power (a, b-1);
}
```

Q. Given a number n, print all natural numbers till n in decreasing order.

n = 5

Base case :- 1

First print n.

Then call function for n-1

Base case :- 0 (simple return)

print n

call for (n-1)

```
void print (int n) {
    if (n==0) return;
    sout (n);
    print (n-1);
}
```