Recursion

> A function that calls itself. The recursion must be stopped when it reaches the base case.

Base Case

This is the key behind recursion, and the first thing that you need to keep in mind.

The Steps

- Pre

  > Something done before the recursion

- Recurse

  > The recursive function call itself

- Post

  > Something done after the recursion

For example, where `n` if a value used in the base case of the function, depending on the place where we print it, it is going to start either at the end of the recursion or at the beginning:

```
def foo(n):

        print(n) # PRE
        if n == 1:
                return 1
        v = n + foo(n - 1)
        print(n) # POS

        return v

n = 5
foo(n)
```

In Pre the output is going to be:

```
5
4
3
2
1
```

And in Pos:

```
2
3
4
5
```

Futhermore, this code:

```
def foo(n):
        print(f'Pre {n}') # pre
        if n == 1:
                return 1

        v = n + foo(n - 1)
        print(f'Pos {n}') # post
        return v
```

```
n = 5
foo(n)
```

Results in:

```
Pre 5
Pre 4
Pre 3
Pre 2
Pre 1
Pos 2
Pos 3
Pos 4
Pos 5
```

> Note
>
> In the post printing there is no `1` because of the `if` statement that is our base case.

> Important
>
> When doing Tree Traversal depending on where we locate the function that does "node visiting" we get a different time or ordering (Breath or Depth First Traversal).