

Recursive version

- An alternative is to think of this computation as:

$$a * b = \underbrace{a + a + \dots + a}_{b \text{ copies}}$$


$$= a + \underbrace{a + \dots + a}_{b-1 \text{ copies}}$$

$$= a + a * (b - 1)$$

Recursion

- This is an instance of a **recursive** algorithm
 - Reduce a problem to a simpler (or smaller) version of the same problem, plus some simple computations
 - **Recursive step**
 - Keep reducing until reach a simple case that can be solved directly
 - **Base case**
- $a * b = a$; if $b = 1$ (**Base case**)
- $a * b = a + a * (b-1)$; otherwise (**Recursive case**)

```
def recurMul(a, b):  
    if b == 1:      Base case  
        return a  
    else:            Recursive step  
        return a + recurMul(a, b-1)
```

A red circle highlights the expression 'b-1' in the recursive call. Two red curved lines originate from the 'a' and '+' signs in the recursive call and point towards the 'a' parameter in the function definition above, illustrating the recursive step where the function calls itself with the same 'a' and a decremented 'b'.