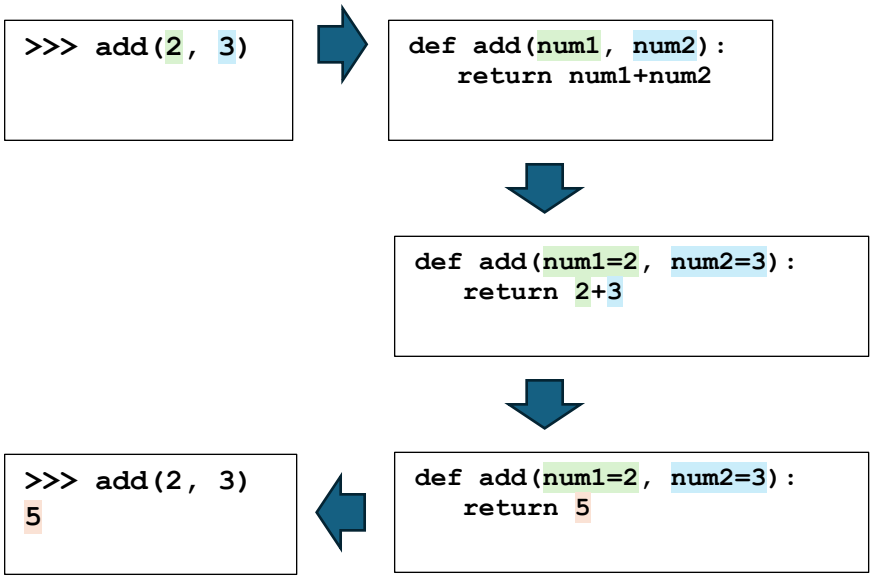
"Functions" allow code reuse. The three functions below simply add, subtract, or multiply two numbers and return the result:

```
>>> def add(num1, num2):
...     return num1+num2
...
>>> add(2, 3)
5
>>> add(8, 7)
15
```

```
>>> def sub(num1, num2):
...     return num1-num2
...
>>> sub(2, 3)
-1
>>> sub(8, 7)
1
```

```
>>> def mul(num1, num2):
...     return num1*num2
...
>>> mul(2, 3)
6
>>> mul(8, 7)
56
```

When you "call" a function, parameter values in the call map to function local variables in the body (such as variables named "num1" and "num2" in the examples above), which are then evaluated, and a return value, if specified, is returned to the caller:



Functions can also define new local variables to hold temporary results, like (this function behaves identically to the "add" function above, but temporarily stores the answer in a new local variable named "answer"):

```
>>> def add(num1, num2):
...     answer = num1+num2
...     return answer
...
```

Challenge:

In mathematics, the **factorial of a non-egative integer n**, denoted as **n!**, is defined as the product of all positive integers from 1 to n. For example, **5! = 5 × 4 × 3 × 2 × 1 = 120**. The factorial of zero is defined as **0! = 1**.

Create a function to compute and return the factorial of an integer parameter:

```
def fact(num):
    what goes here?
    return what?
```

Then call it five times to compute 2!, 3!, 5!, 60!, and finally 0!:

```
>>> fact(2)
2
>>> fact(3)
6
>>> fact(5)
120
>>> fact(60)
8320987112741390144276341183223364380754172606361245952449277696409600000000000000
>>> fact(0)
1
```