Recursive Functions

Recursive Functions

Recursive functions are functions that make calls to themselves.

They can be used in place of loops. Though in Python they don't necessarily provide a more efficient solution, there are many problems for which a recursive function is the most elegant and convenient solution.

Worked Example: Factorial

One of the most famous implementations of a recursive function is to implement the factorial:

$$\begin{aligned} &0! = 1 \\ &n! = n \times (n-1) \times (n-2) \times (n-3) \times \cdots \times 2 \times 1 \end{aligned}$$

This is achieved by using the recurrence relation:

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

The recursive function which solves this is:

[10]:
```
def factorial(n):
    if not type(n) is int:
        print('n must be an integer')
        return
    if n <0:
        print('n must be greater than or equal to 0')
        return

    if n == 0:
        return 1

    return n*factorial(n-1)
```

Note, an important aspect of this function is the return value of 1 for `n == 0`. This is called the base class, without it the function would never finish it's recursion.

Putting this function into action:

[11]:
```
factorial(-1)
```

```
n must be greater than or equal to 0
```

[12]:
```
factorial(0.5)
```

```
n must be an integer
```

[4]:
```
factorial(0)
```

[4]: 1

[5]:
```
factorial(1)
```

[5]: 1

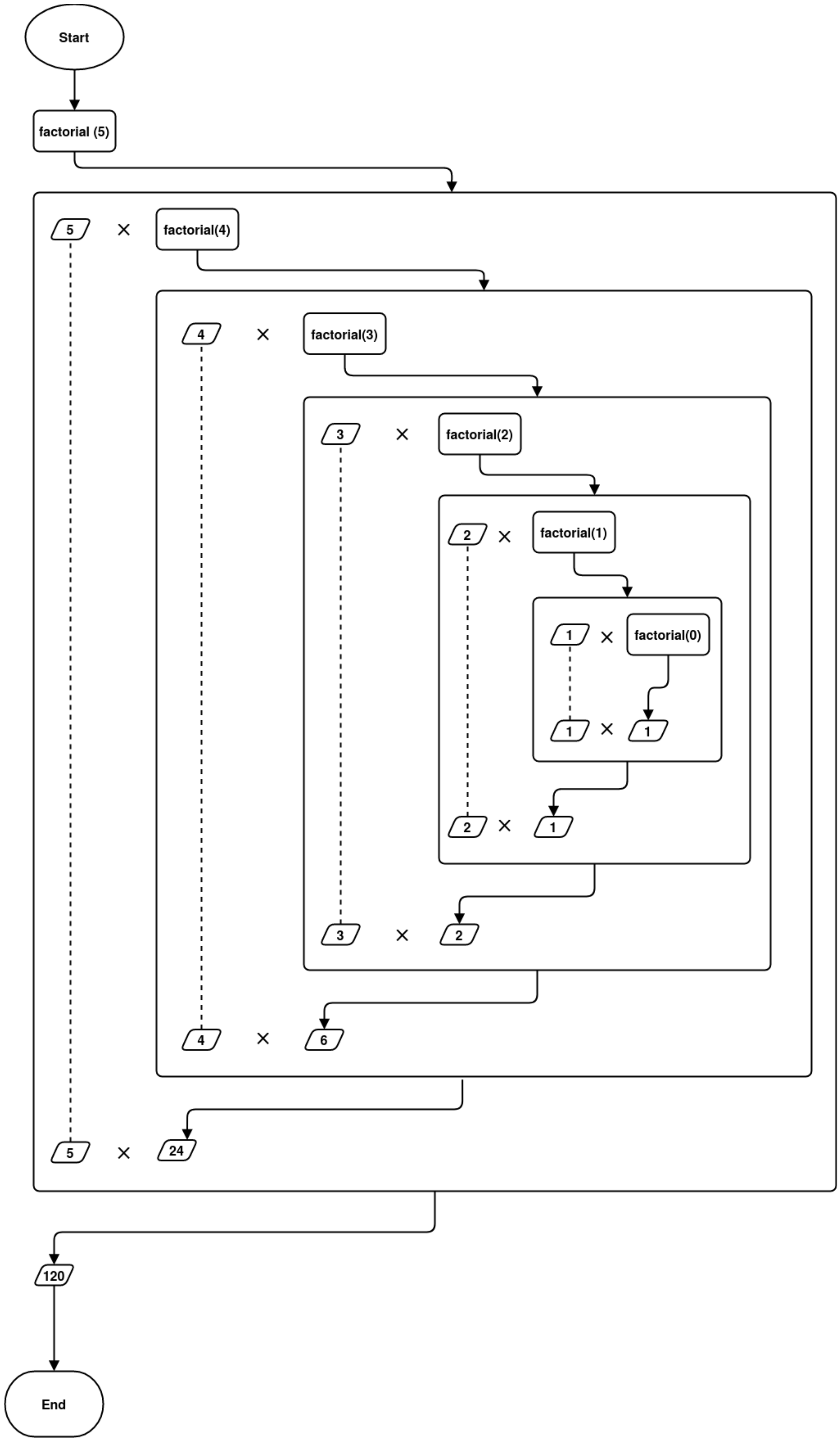
[8]:
```
factorial(5)
```

[8]: 120

[9]:
```
factorial(10)
```

[9]: 3628800

The inner workings of this `factorial()` function are fairly subtle. The (informal) flow diagram below illustrates the function call for `factorial(5)`:



Start
factorial (5)
5
×
factorial(4)
4
×
factorial(3)
3
×
factorial(2)
2
×
factorial(1)
1
×
factorial(0)
1
×
1
2
×
1
3
×
2
4
×
6
5
×
24
120
End

The Base Class

As mentioned earlier, a recursive function must have at least one base class. The base class is a return state that **doesn't** make another recursive function call.

It's also important to make sure that the recursion eventually reaches the base class when designing your function.