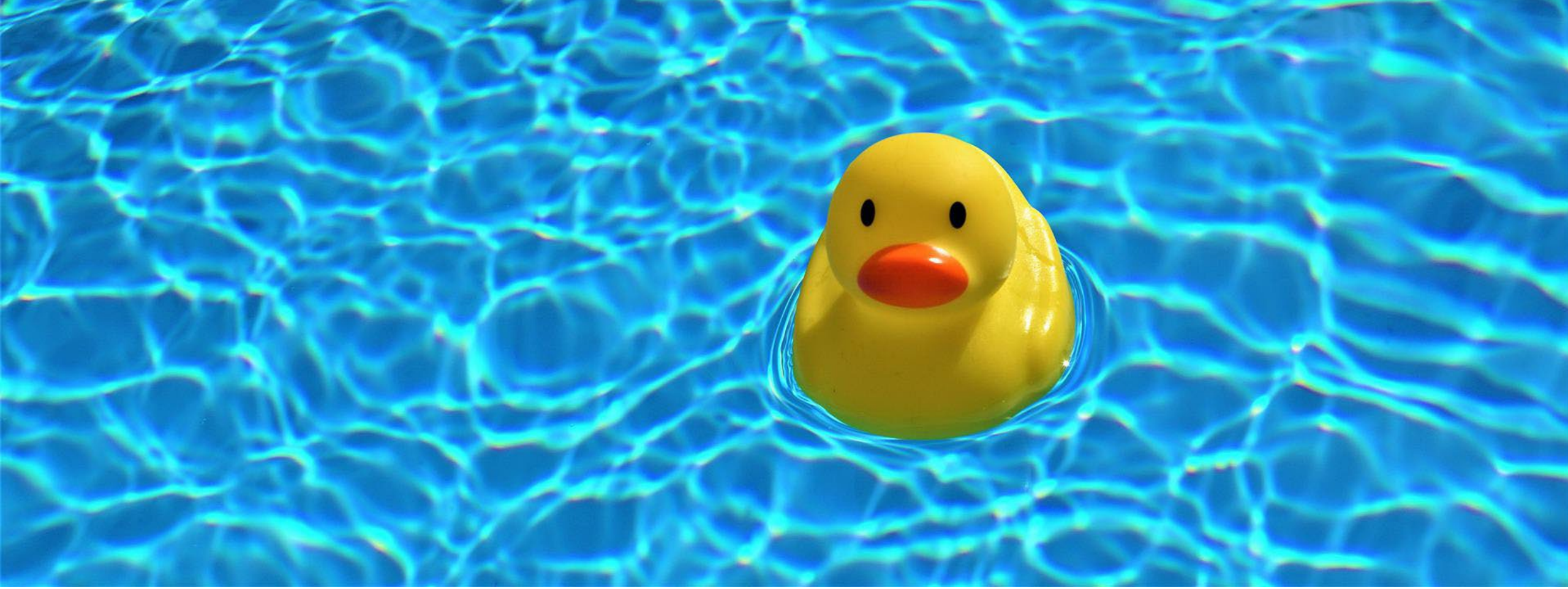


CODE FROM: 15 RECURSIVE FUNCTIONS

CASE STUDY: FACTORIALS

SOLVING WITH FOR-LOOP

```python
import math

number = 5
factorial_number = 1
for i in range(1, number+1):
    factorial_number *= i
print(f"{number}! = {factorial_number}")
```

RECURSIVE FUNCTION

definition: a function that calls on itself

```
1  def factorial(n):
2      if n==1:
3          return n
4      else:
5          return n*(factorial(n-1))
```

consists of two parts:

base case – a certain parameter that will always result in a specific, predefined result

recursive case – where the function calls on itself

FACTORIAL(5) STEP BY STEP

```
1  def factorial(n):
2      if n==1:
3          return n
4      else:
5          return n*(factorial(n-1))
```

FACTORIAL(5): FIRST STEP

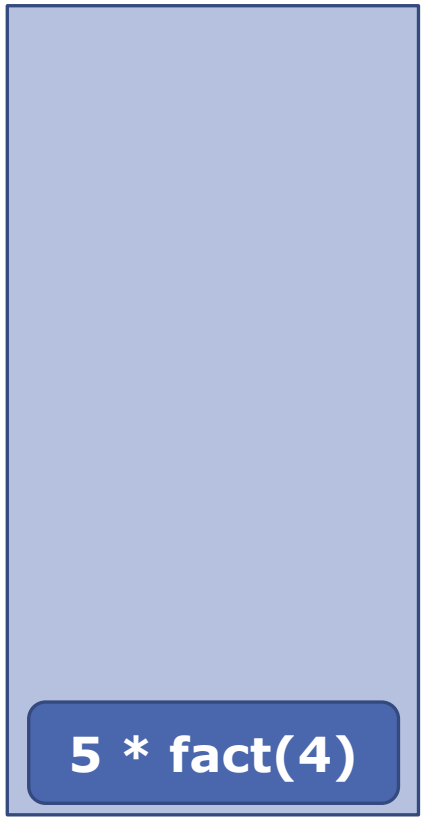
```
1  def factorial(n):
2      if n==1:
3          return n
4      else:
5          return n*(factorial(n-1))
```

n=5

when n!=1, the function returns n*(factorial(n-1))

return: 5 * factorial(4)

Every time we call on a recursive function and we peel away part of the problem, that little part we just peeled away is put on "the stack" (do it later list)



FACTORIAL(5): NEXT STEP

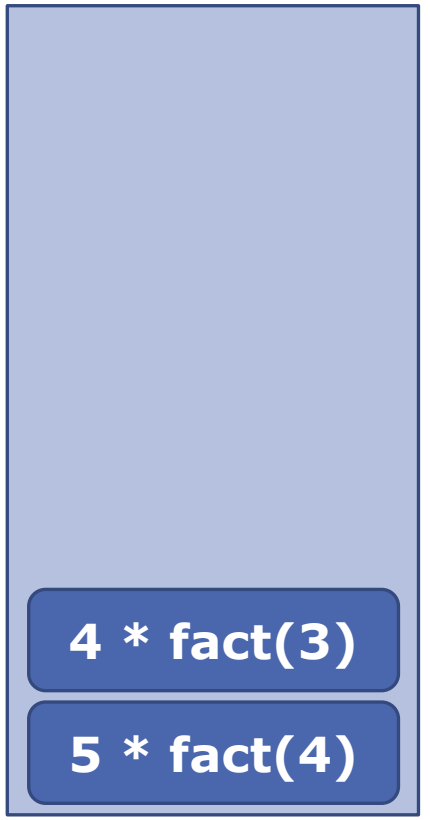
```
1  def factorial(n):
2      if n==1:
3          return n
4      else:
5          return n*(factorial(n-1))
```

n=4

when n!=1, the function returns n*(factorial(n-1))

return: 4 * factorial(3)

STACK:



FACTORIAL(5): NEXT STEP

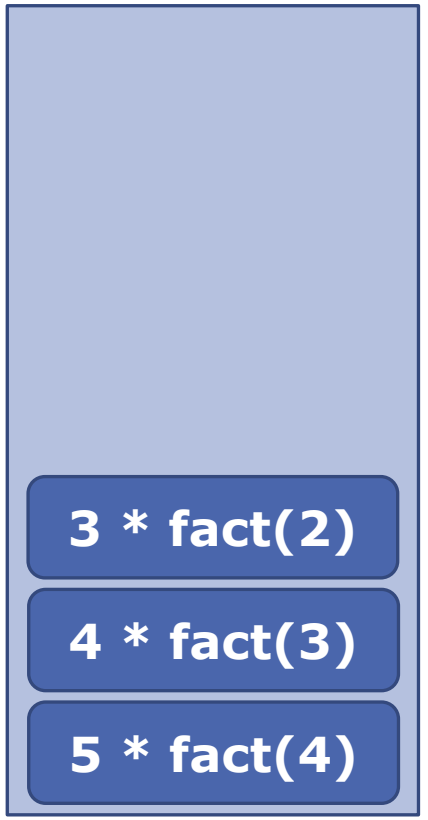
```
1  def factorial(n):
2      if n==1:
3          return n
4      else:
5          return n*(factorial(n-1))
```

n=3

when n!=1, the function returns n*(factorial(n-1))

return: 3 * factorial(2)

STACK:



FACTORIAL(5): NEXT STEP

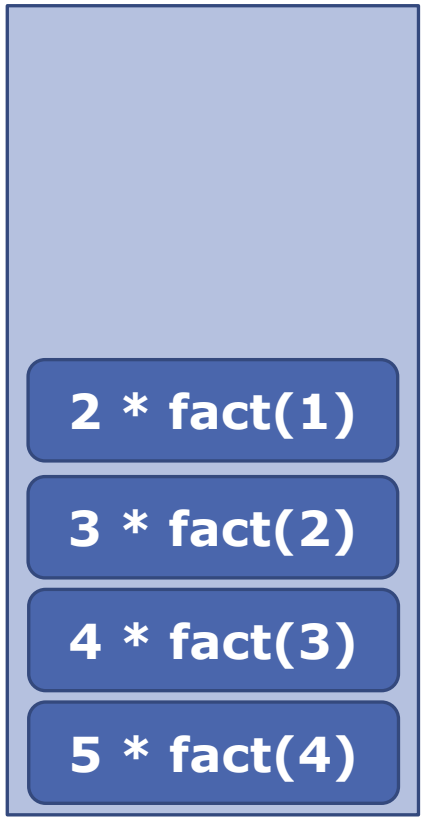
```
1  def factorial(n):
2      if n==1:
3          return n
4      else:
5          return n*(factorial(n-1))
```

n=2

when n!=1, the function returns n*(factorial(n-1))

return: 2 * factorial(1)

STACK:



FACTORIAL(5): NEXT STEP

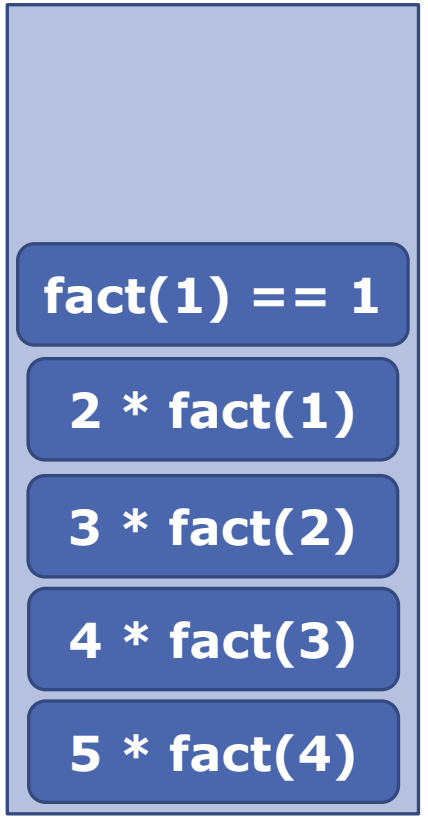
```
1  def factorial(n):
2      if n==1:
3          return n
4      else:
5          return n*(factorial(n-1))
```

n=1

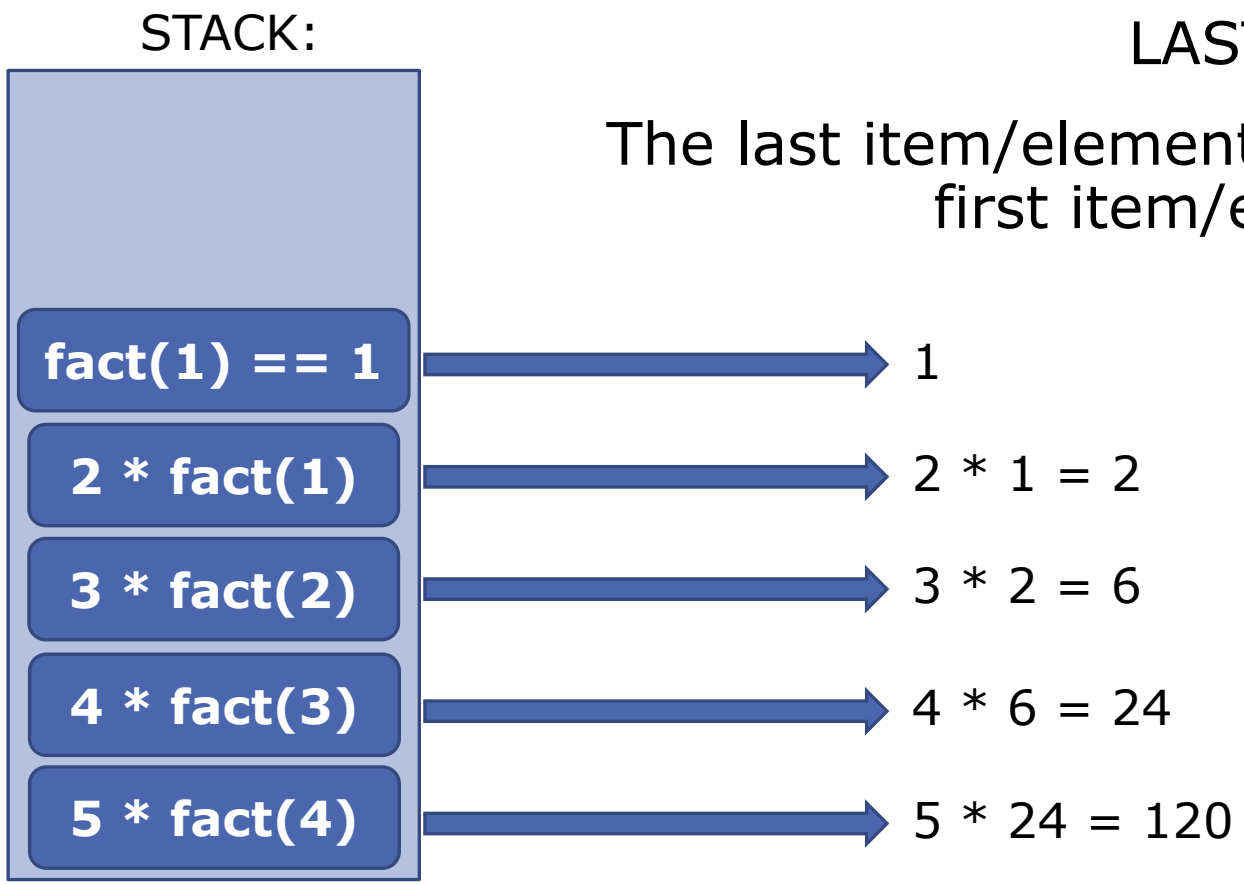
when n==1, the function returns n

return: 1

STACK:



THE STACK



LAST IN, FIRST OUT

The last item/element to be put on the stack is also the first item/element to be retrieved