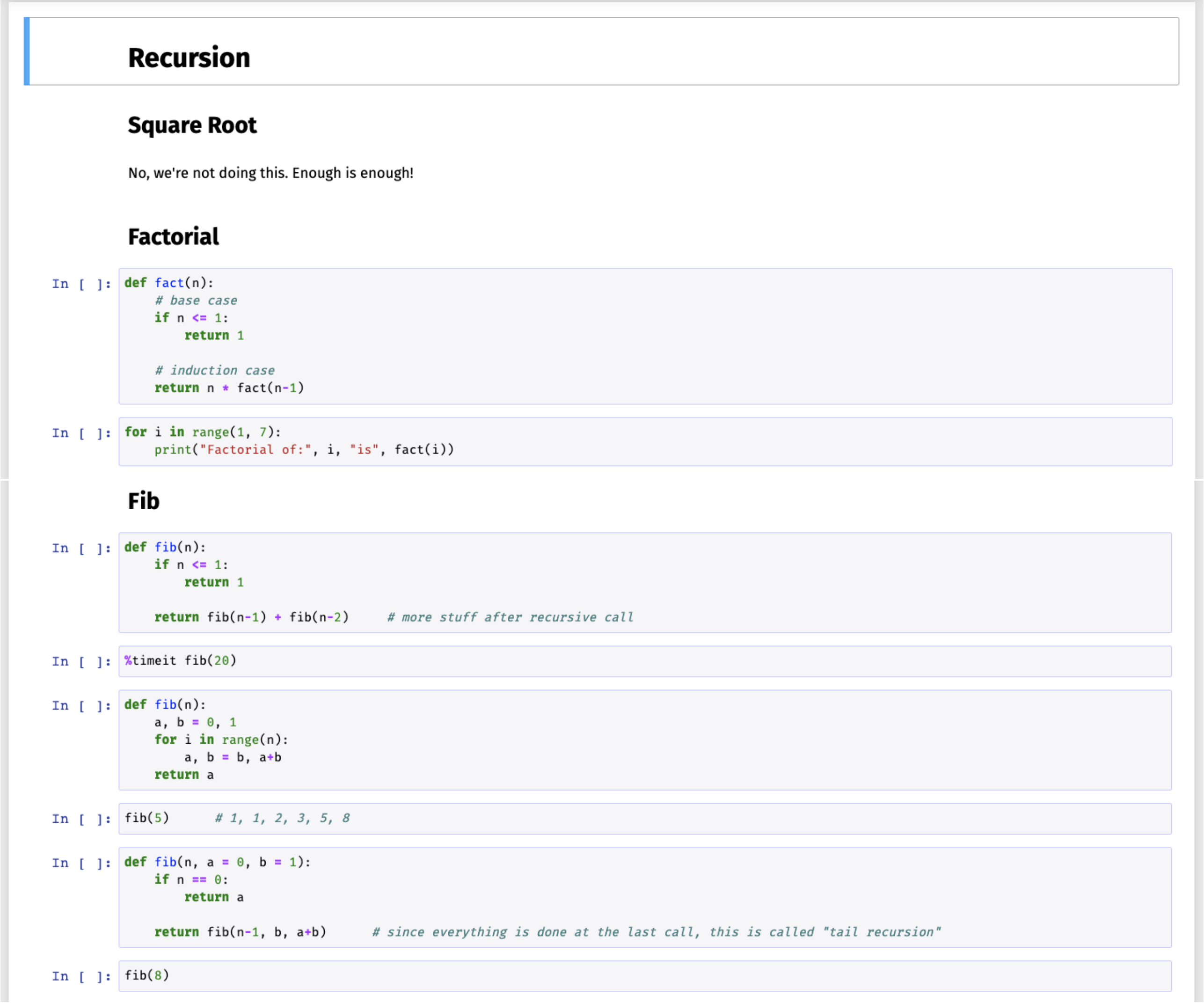
CS218 - Data Structures
FAST NUCES Peshawar Campus
Dr. Nauman (recluze.net)

September 16, 2019

1 Recursion

Raster images of the notebook 10-recursion



Recursion

Square Root

No, we're not doing this. Enough is enough!

Factorial

In [ ]:
```
def fact(n):
    # base case
    if n <= 1:
        return 1

    # induction case
    return n * fact(n-1)
```

In [ ]:
```
for i in range(1, 7):
    print("Factorial of:", i, "is", fact(i))
```

Fib

In [ ]:
```
def fib(n):
    if n <= 1:
        return 1

    return fib(n-1) + fib(n-2)      # more stuff after recursive call
```

In [ ]:
```
%timeit fib(20)
```

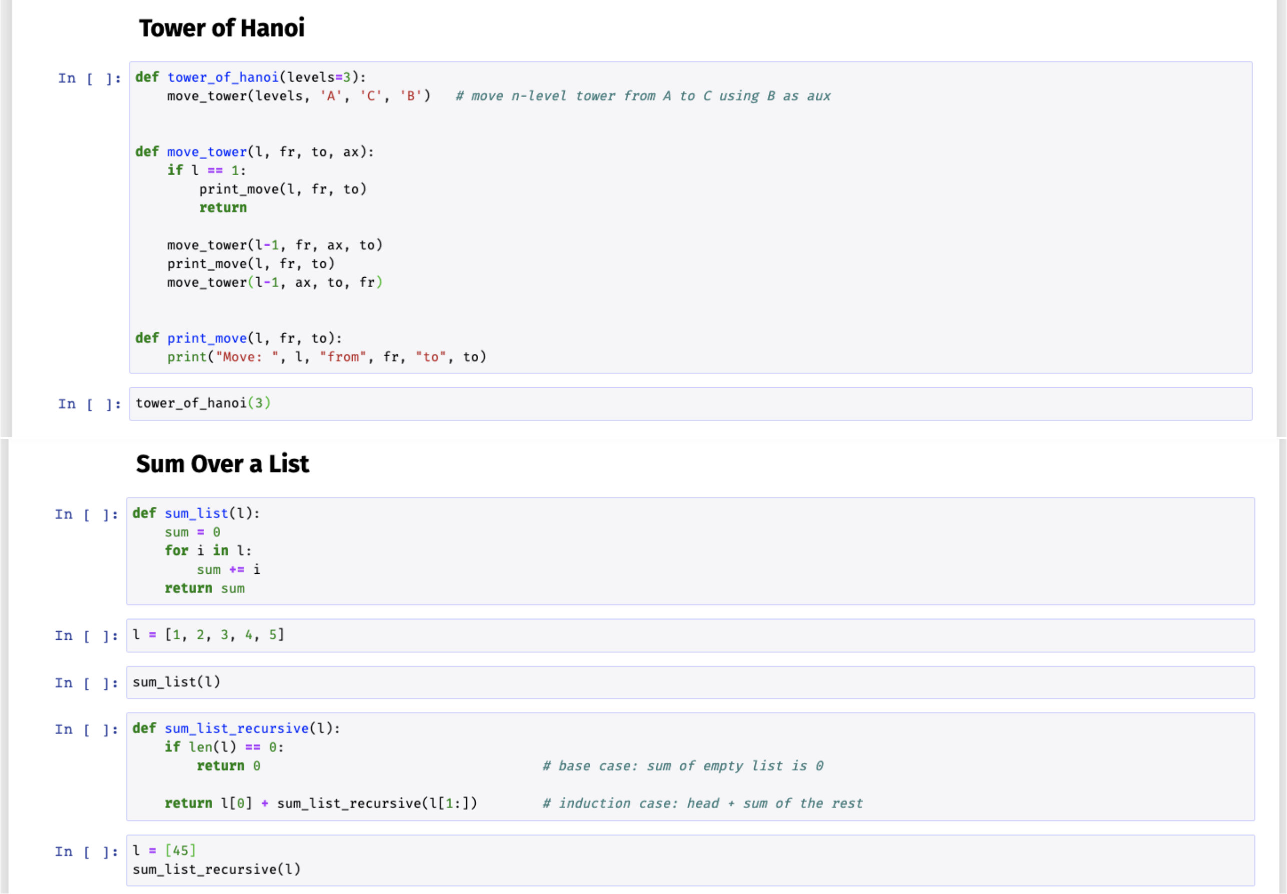
In [ ]:
```
def fib(n):
    a, b = 0, 1
    for i in range(n):
        a, b = b, a+b
    return a
```

In [ ]:
```
fib(5)      # 1, 1, 2, 3, 5, 8
```

In [ ]:
```
def fib(n, a = 0, b = 1):
    if n == 0:
        return a

    return fib(n-1, b, a+b)      # since everything is done at the last call, this is called "tail recursion"
```

In [ ]:
```
fib(8)
```



Tower of Hanoi

```
In [ ]: def tower_of_hanoi(levels=3):
            move_tower(levels, 'A', 'C', 'B')    # move n-level tower from A to C using B as aux


        def move_tower(l, fr, to, ax):
            if l == 1:
                print_move(l, fr, to)
                return

            move_tower(l-1, fr, ax, to)
            print_move(l, fr, to)
            move_tower(l-1, ax, to, fr)


        def print_move(l, fr, to):
            print("Move: ", l, "from", fr, "to", to)
```

```
In [ ]: tower_of_hanoi(3)
```

Sum Over a List

```
In [ ]: def sum_list(l):
            sum = 0
            for i in l:
                sum += i
            return sum
```

```
In [ ]: l = [1, 2, 3, 4, 5]
```

```
In [ ]: sum_list(l)
```

```
In [ ]: def sum_list_recursive(l):
            if len(l) == 0:
                return 0                              # base case: sum of empty list is 0

            return l[0] + sum_list_recursive(l[1:])        # induction case: head + sum of the rest
```

```
In [ ]: l = [45]
        sum_list_recursive(l)
```