CEng 240 – Spring 2021
Week 7

Sinan Kalkan

Functions

Disclaimer: Figures without reference are from either from "Introduction to programming concepts with case studies in Python" or "Programming with Python for Engineers", which are both co-authored by me.

This Week

- Functions
  - Why define functions?
  - Defining functions
  - Parameter passing
  - Default parameters
  - Scope of variables
  - Examples
- Next week:
  - Recursion, higher-order functions
  - Examples

Administrative Notes

- Lab 4
- Midterm: 1 June, Tuesday, 17:40

Why define functions?

- Reusability
- Maintenance
- Structure

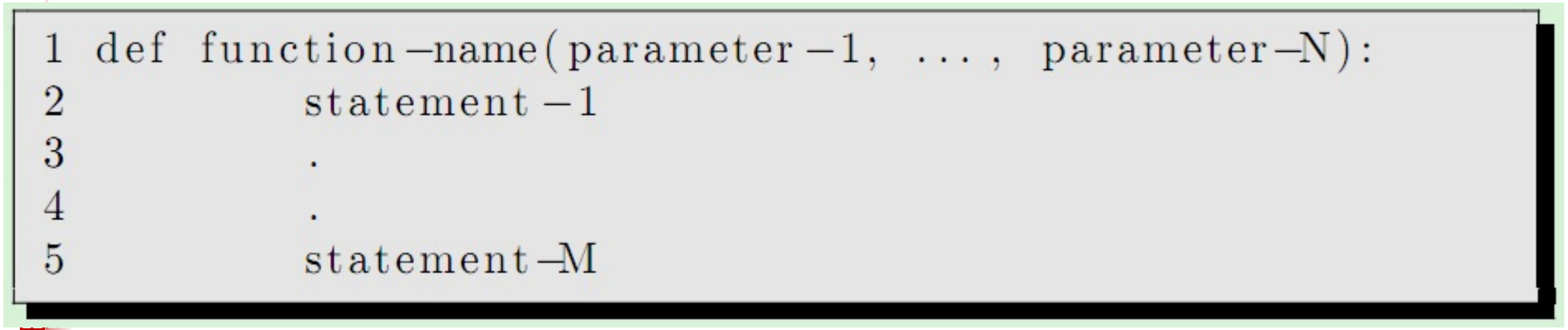
METU Computer Engineering

Functions in programming vs. Mathematics

- Functions in programming are similar to functions in Mathematics but there are differences.

- Difference to mathematical functions:
  - A function in programming may not return a value.
  - A function in mathematics only depends on its arguments unlike the functions in programming.
  - A mathematical function does not have the problem of side effects.

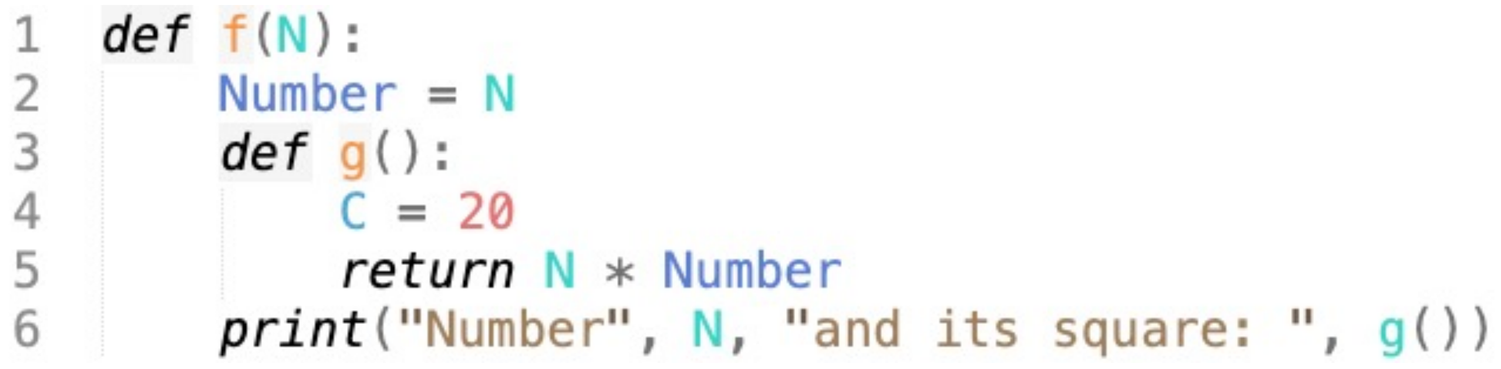
Functions in Python



```
def function-name(parameter-1, ..., parameter-N):
        statement-1
        .
        .
        statement-M
```

- Syntax is important!
- Indentation is extremely important!

Nested Functions in Python

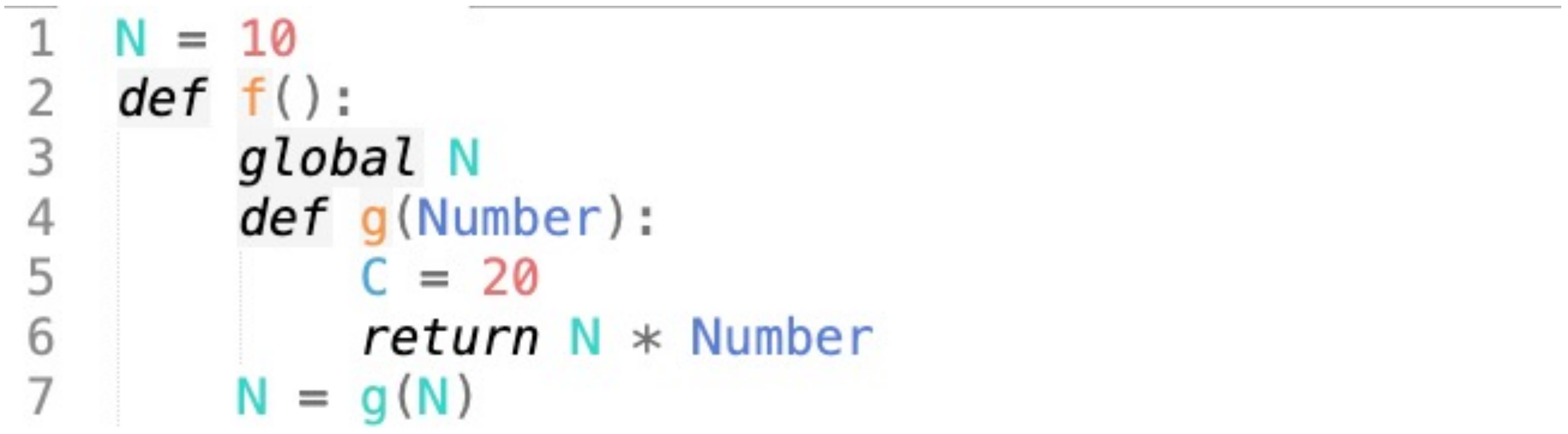


```python
def f(N):
    Number = N
    def g():
        C = 20
        return N * Number
    print("Number", N, "and its square: ", g())
```

- Function `g()` can access all the local variables as well as the parameters of function `f()`.
- Function `f()` cannot access the local variables of function `g()`!
- Function `g()` cannot be used before it is defined! For example, the second line could not have been `Number = 10 * g(10)`.
- The indentation is extremely important to understand which statement belongs to which function! For example, the last line is part of function `f()` since they are at the same indentation!

Global Variables in Python

- To access variables in the global workspace, you should use “global <varname>”



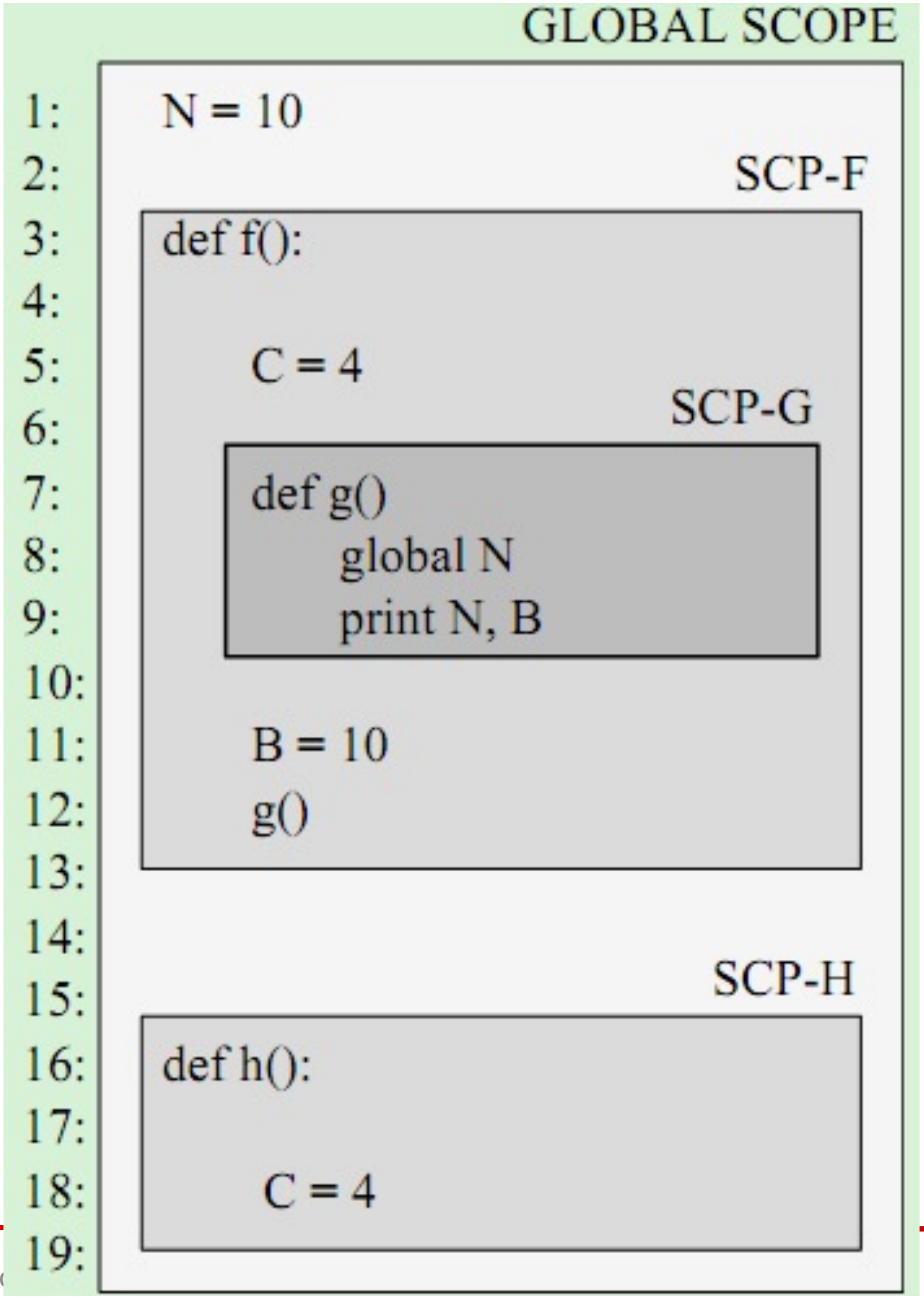
```
N = 10
def f():
    global N
    def g(Number):
        C = 20
        return N * Number
    N = g(N)
```

Scope in Python

- Since you can nest functions in Python, understanding scope is important

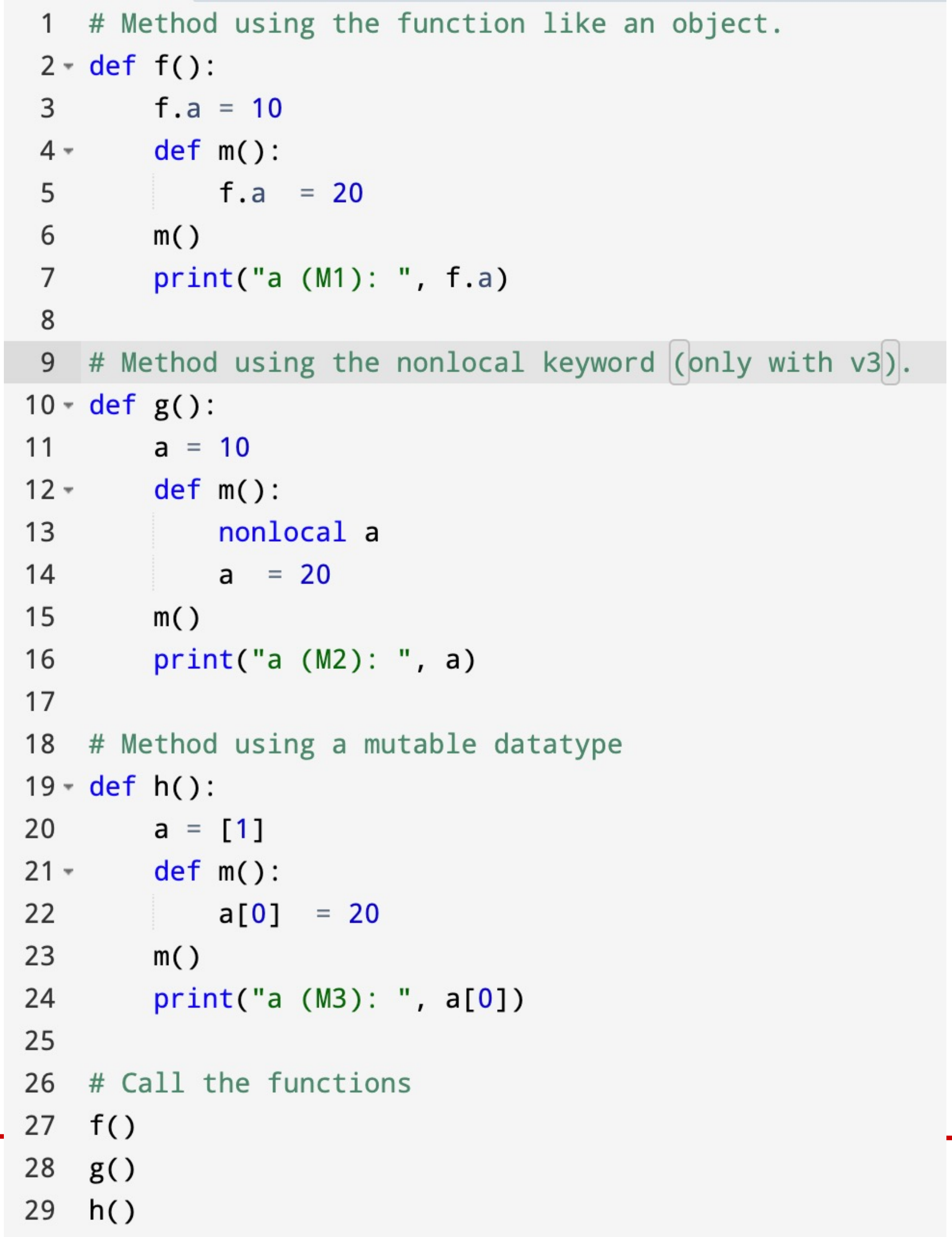
- LEGB rule:

Local < Enclosing < Global < Built-in



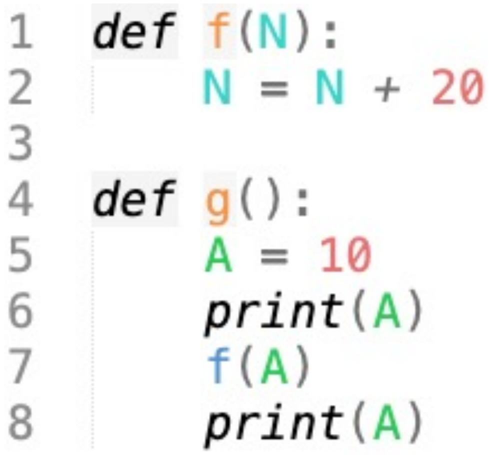
```
N = 10

def f():

    C = 4

    def g()
        global N
        print N, B

    B = 10
    g()



def h():

    C = 4

```

- Updating variables of an outer function

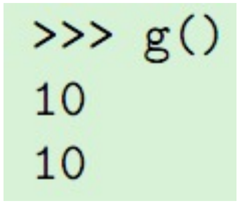


```python
# Method using the function like an object.
def f():
    f.a = 10
    def m():
        f.a  = 20
    m()
    print("a (M1): ", f.a)

# Method using the nonlocal keyword (only with v3).
def g():
    a = 10
    def m():
        nonlocal a
        a  = 20
    m()
    print("a (M2): ", a)

# Method using a mutable datatype
def h():
    a = [1]
    def m():
        a[0]  = 20
    m()
    print("a (M3): ", a[0])

# Call the functions
f()
g()
h()
```

Parameter passing in functions in Python



```python
def f(N):
    N = N + 20

def g():
    A = 10
    print(A)
    f(A)
    print(A)
```



```
>>> g()
10
10
```

Parameter passing in functions in Python

```python
def f(List):
    List[0] = 'A'

def g():
    L = [1, 2, 3]
    print(L)
    f(L)
    print(L)
```

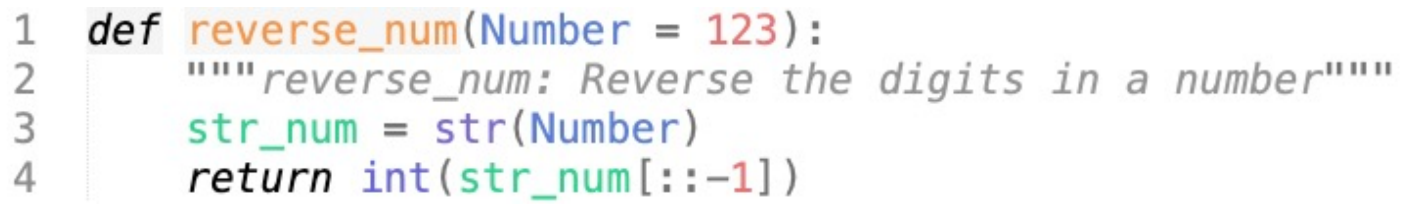
```
>>> g()
[1, 2, 3]
['A', 2, 3]
```

Parameter passing in functions in Python

```python
def f(List):
    List = List[::-1]

def g():
    L = [1, 2, 3]
    print(L)
    f(L)
    print(L)
```

```
>>> g()
[1, 2, 3]
[1, 2, 3]
```

Default Parameters in Python



```python
def reverse_num(Number = 123):
    """reverse_num: Reverse the digits in a number"""
    str_num = str(Number)
    return int(str_num[::-1])
```

- We can now call this function with reverse_num() in which case Number is assumed to be 123.
- If we supply a value for Number, that value is used instead.

Function Examples

- Finding max of numbers
- Sequential search
- Write a Python function named only_numbers() that removes items in a list that are not numbers.
  - E.g. only_numbers([10, "ali", [20], True, 4]) should return [10, 4].
- Insertion sort

Final Words: Important Concepts

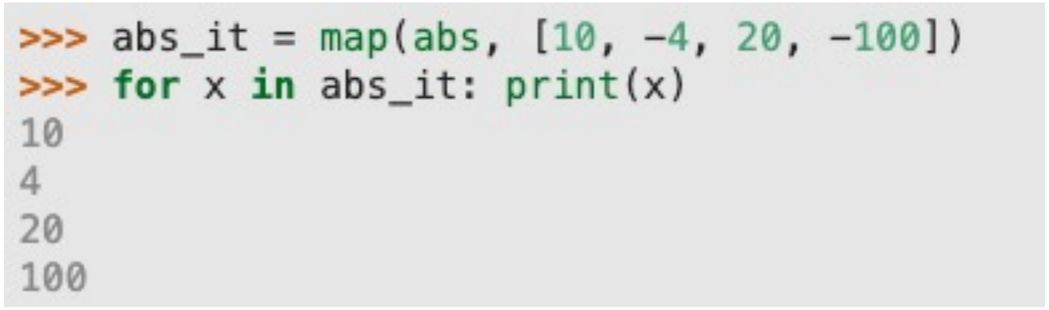
- Benefits of defining functions
- How to define functions
- Default parameters
- Scopes of variables

THAT'S ALL FOLKS!
STAY HEALTHY

N

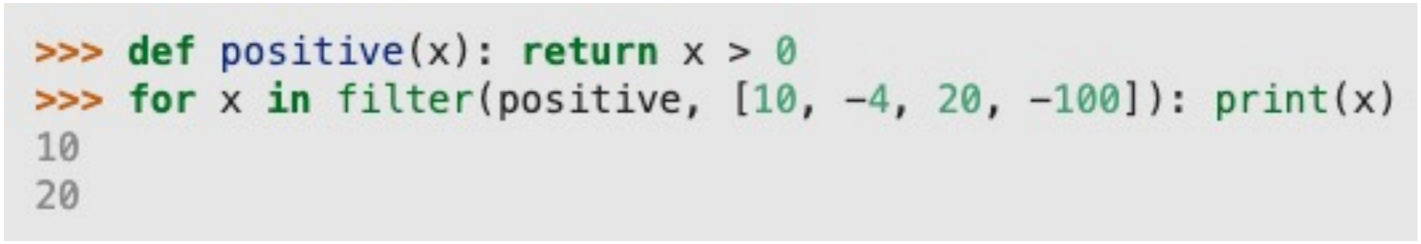
Higher-order functions

- map(function, Iterator)



```
>>> abs_it = map(abs, [10, -4, 20, -100])
>>> for x in abs_it: print(x)
10
4
20
100
```

- filter(predicate, Iterator)



```
>>> def positive(x): return x > 0
>>> for x in filter(positive, [10, -4, 20, -100]): print(x)
10
20
```

Recursion