Python: The Easy Way

Lecture 2

Data Structures

lists

Intro

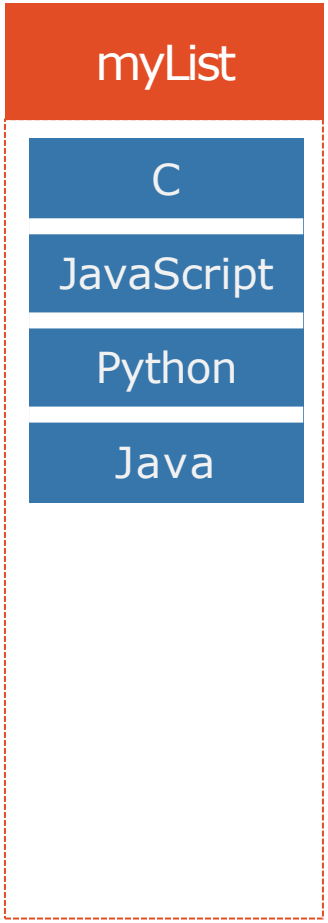
A collection of various data types

```
newList = []
```

```
newList = [1, "hi", True]
newList[0]   #1
newList[1]   #"Hi"
newList[2]   #True
newList[3]   #Index Error
```

Methods

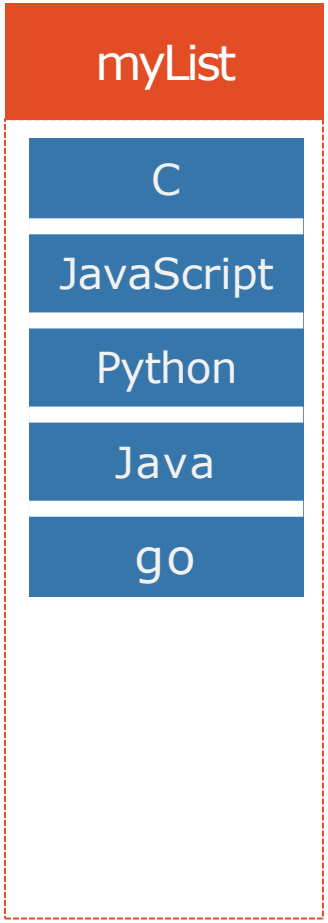
```
myList = ["C", "JavaScript", "Python", "Java", "php"];
```




```
myList.pop(4)
```

Methods

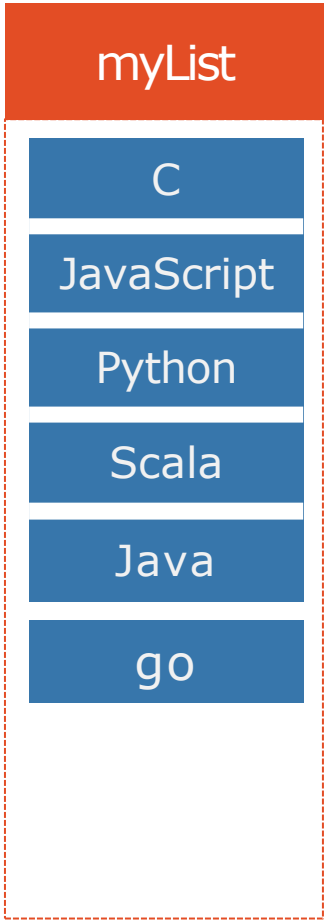
```
myList = ["C", "JavaScript", "Python", "Java", "php"];
```




```
myList.pop(4)

myList.append("go")
```

Methods

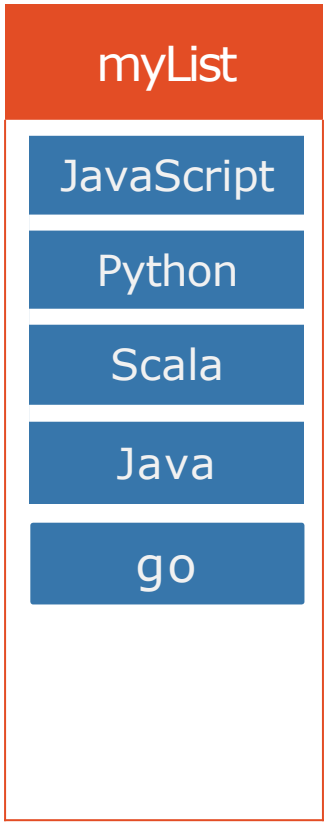
```
myList = ["C", "JavaScript", "Python", "Java", "php"];
```




```
myList.pop(4)

myList.append("go")

myList.insert(3, 'Scala')
```

Methods

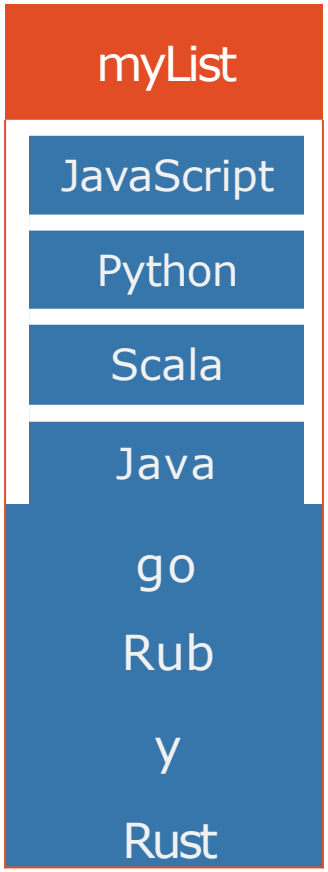
myList = [“C”, “JavaScript”, “Python”, “Java”, “php”];




```
myList.pop(4)

myList.append(“go”)

myList.insert(3, ‘Scala’)

myList.remove(“C”)
```

Methods

```
myList = ["C", "JavaScript", "Python", "Java", "php"];
```




```
myList.pop(4)

myList.append("go")

myList.insert(3, 'Scala')

myList.remove("C")

yourList = ["Ruby", "Rust"];

myList.extend(yourList)
```

Tuples

Immutable Lists

Same as Lists but Tuples are immutable

```
newTuple = ()
```

```
t = (1, “hi”, True)
t[1]
# hi
t[1] = 4
TypeError: 'tuple' object does not support item assignment
```

Dictionaries

Key/value Pairs

A key: value **comma** seperated elements Data Structure

```
newDict = {}
```

```
d = {"name": "Ahmed", "track": "OS"}
d["name"]
# Ahmed
d["name"] = "Ali"
# {name: "Ali", track: "OS"}
```

Methods

```
infoDict = {'track': 'OS', 'name': 'Ahmed', 'age': 17}

infoDict.keys() # dict_keys(['track', 'name', 'age'])

'name' in infoDict # True

infoDict.items()

# dict_items([('track', 'OS'), ('name', 'Ahmed'), ('age', 17)])

addInfoDict = {'track': 'SD', 'branch': "Smart"}

infoDict.update(addInfoDict)

#{'track': 'SD', 'name': 'Ahmed', 'age': 17 , 'branch': "Smart"}
```

Scope

To know your limits

Lexical Scope

Output:

Lexical Scope

```
name = "Ahmed"
```

Output:

Lexical Scope

```
name = "Ahmed"
def outerFn():
        name = "Ali"
        def innerFn():
                print(name)
        innerFn()
```

Output:

Lexical Scope

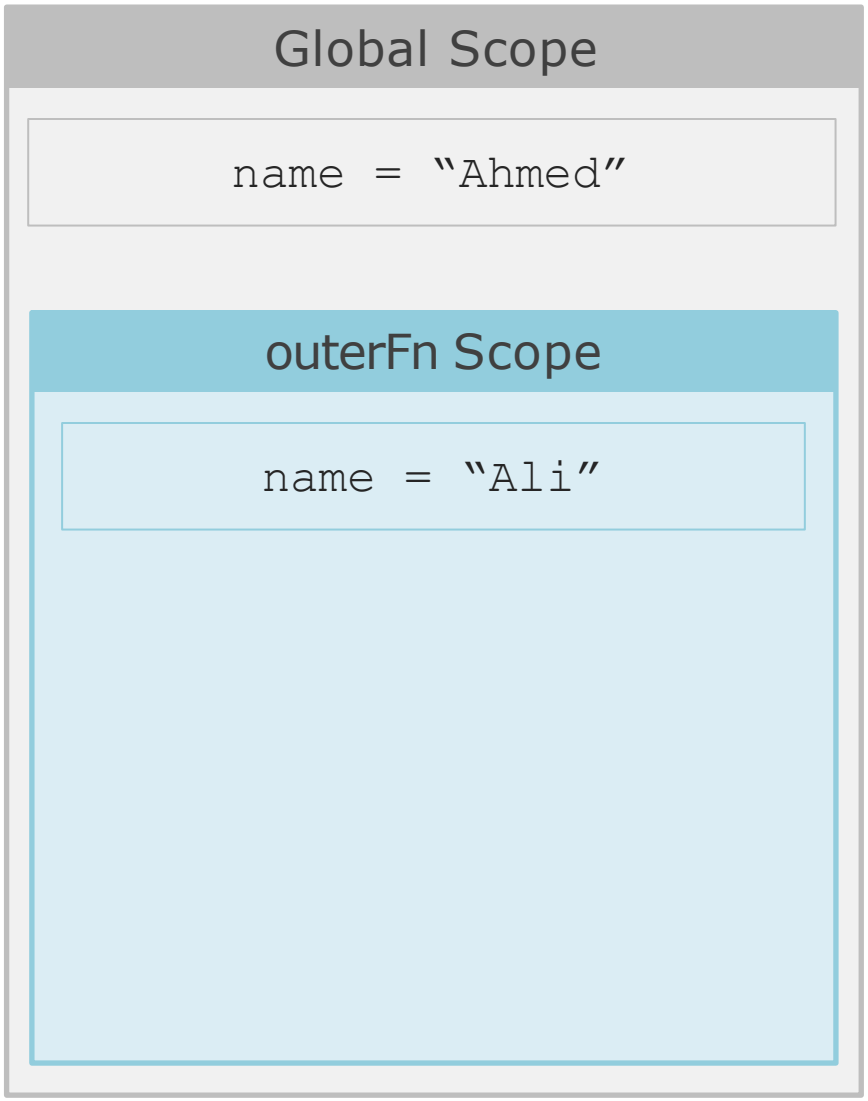
```
name = "Ahmed"
def outerFn():
        name = "Ali"
        def innerFn():
                print(name)
        innerFn()
outerFn()
```

Output:

Lexical Scope

```
name = "Ahmed"
def outerFn():
    → name = "Ali"
        def innerFn():
            print(name)
        innerFn()
outerFn()
```

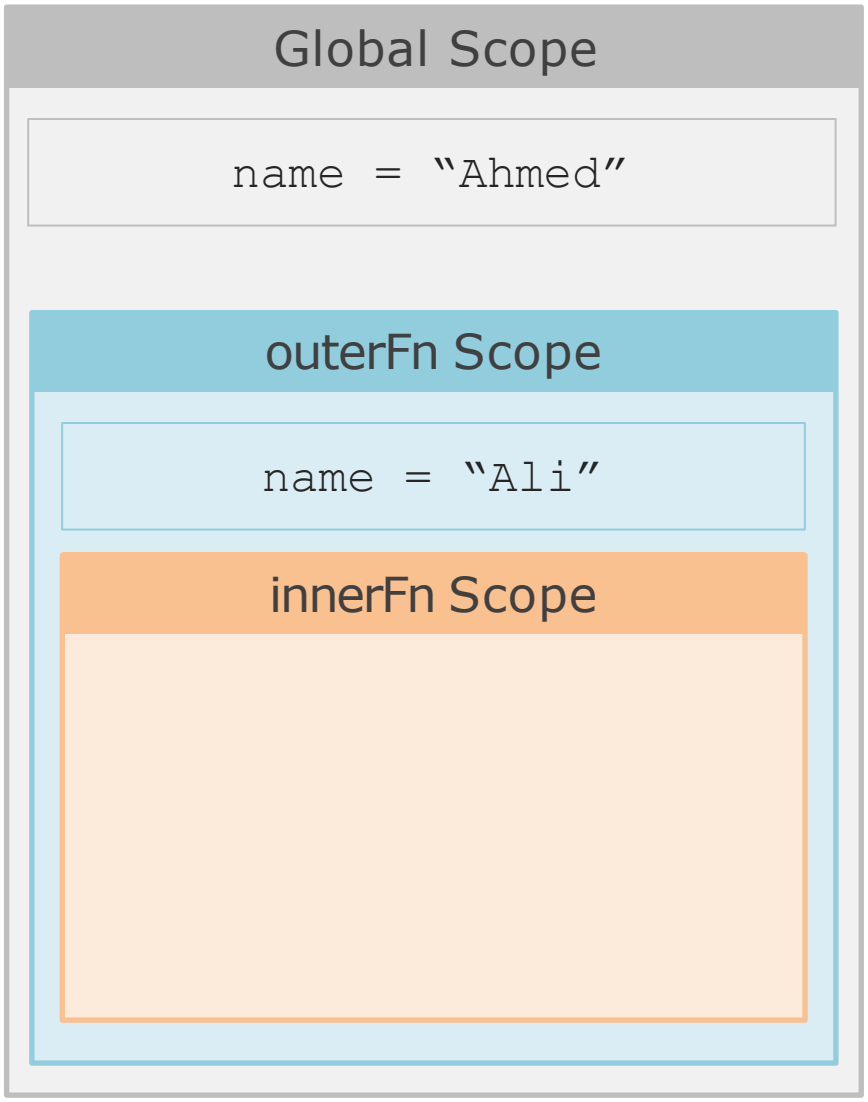
Output:



Lexical Scope

```
name = "Ahmed"
def outerFn():
        name = "Ali"
        def innerFn():
                print(name)
    →   innerFn()
outerFn()
```

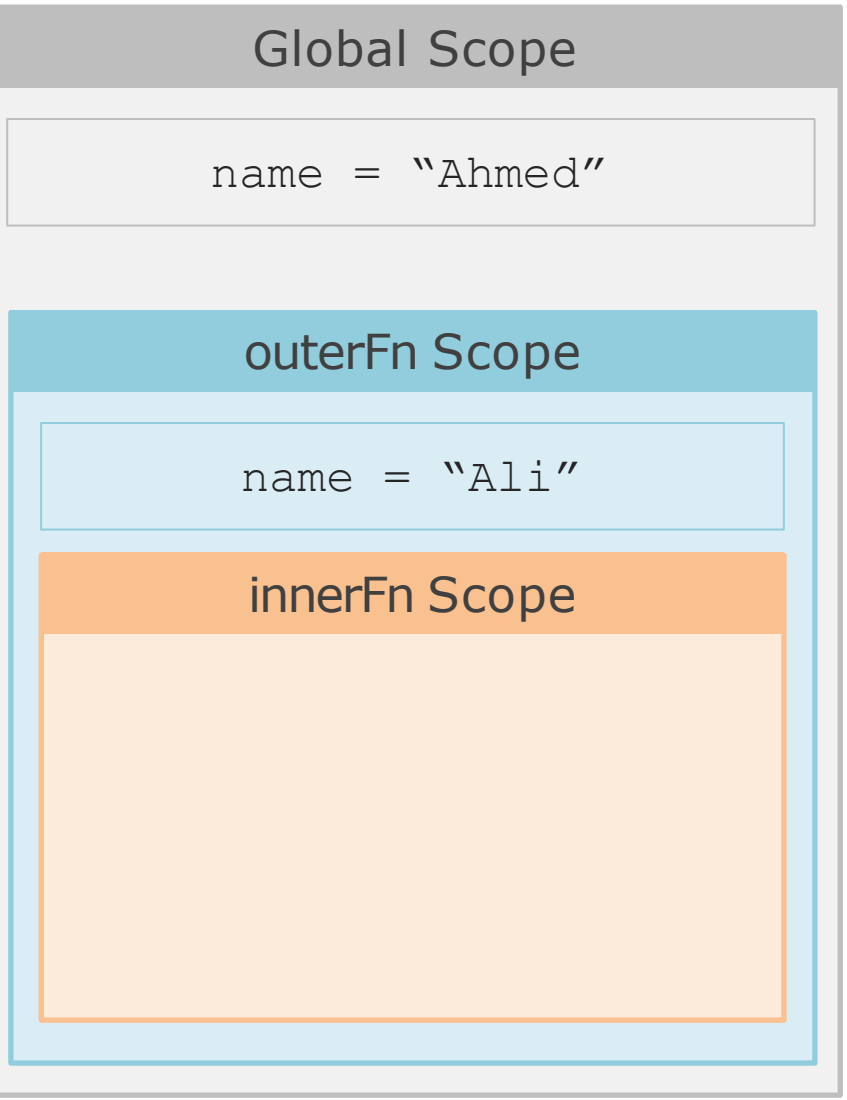
Output:



Lexical Scope

```
name = "Ahmed"
def outerFn():
        name = "Ali"
        def innerFn():
            print(name)
        innerFn()
outerFn()
```

Output:

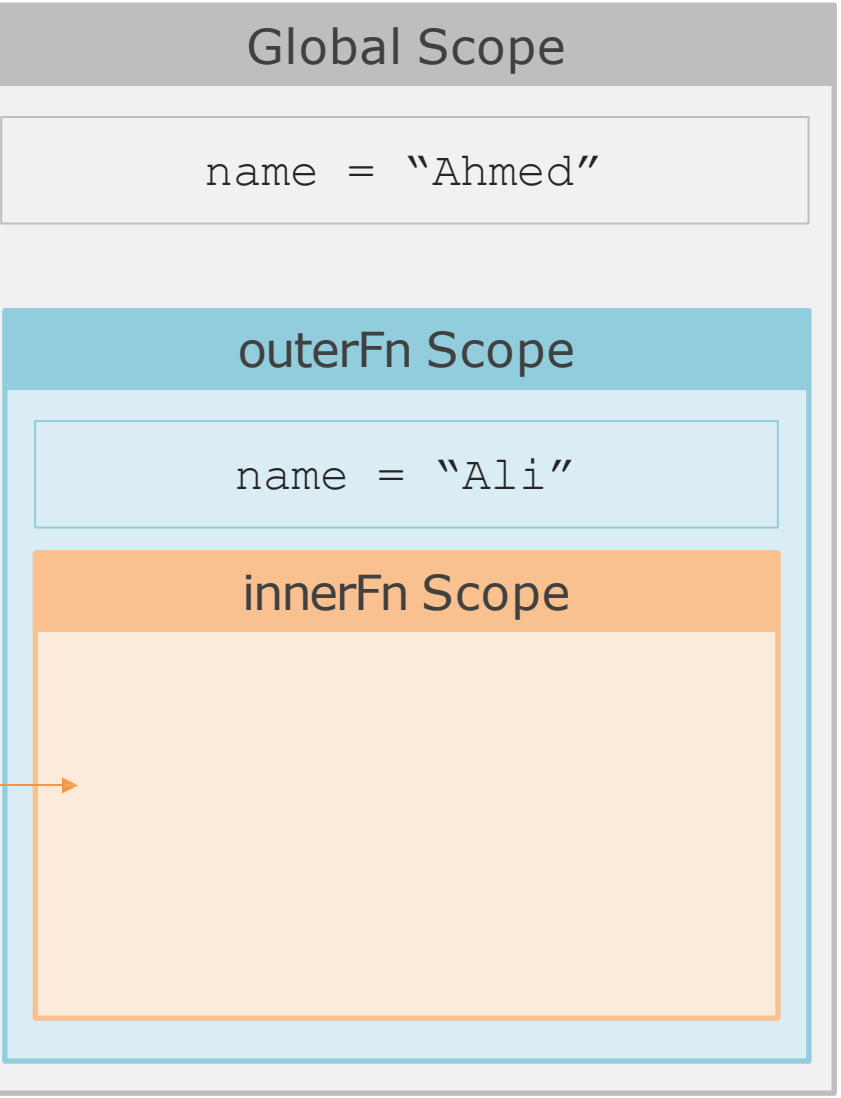


Lexical Scope

```
name = "Ahmed"
def outerFn():
        name = "Ali"
        def innerFn():
            print(name)
        innerFn()
outerFn()
```

Output:

name
???

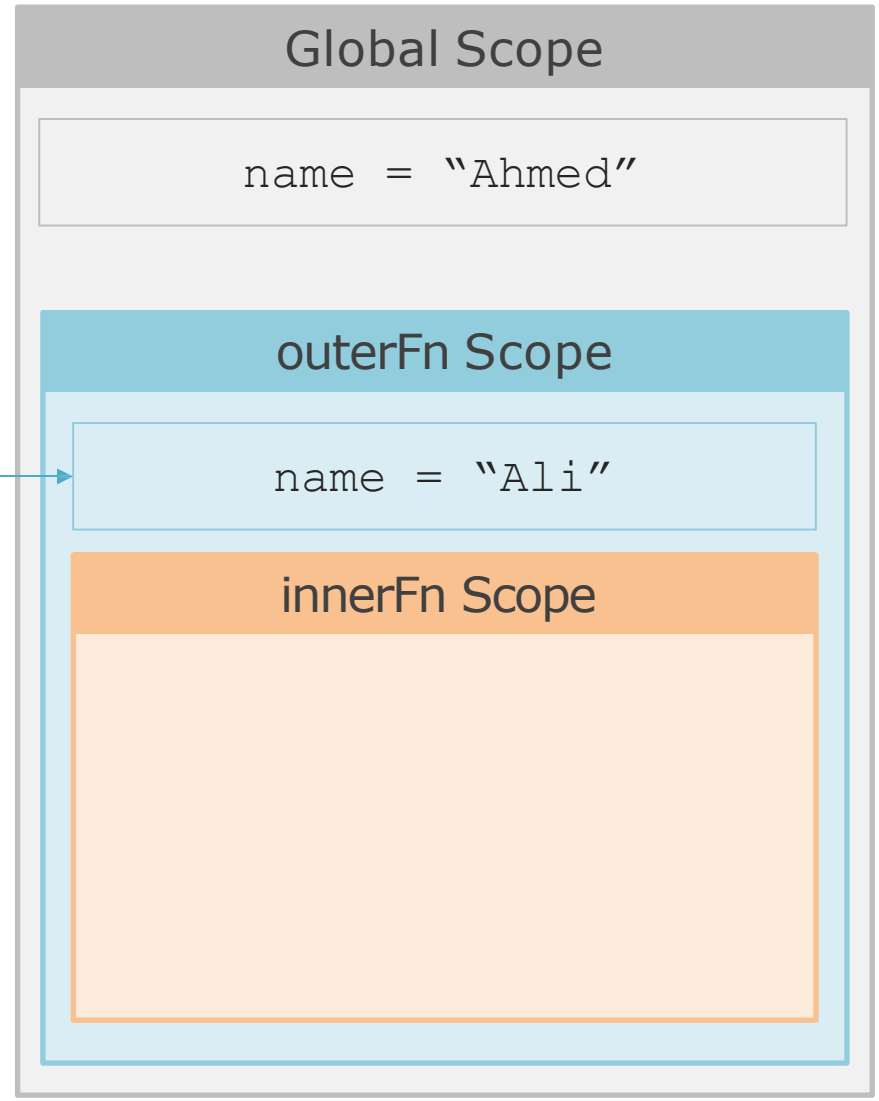


Lexical Scope

```
name = "Ahmed"
def outerFn():
    name = "Ali"
    def innerFn():
        print(name)
    innerFn()
outerFn()
```

Output:

name
???



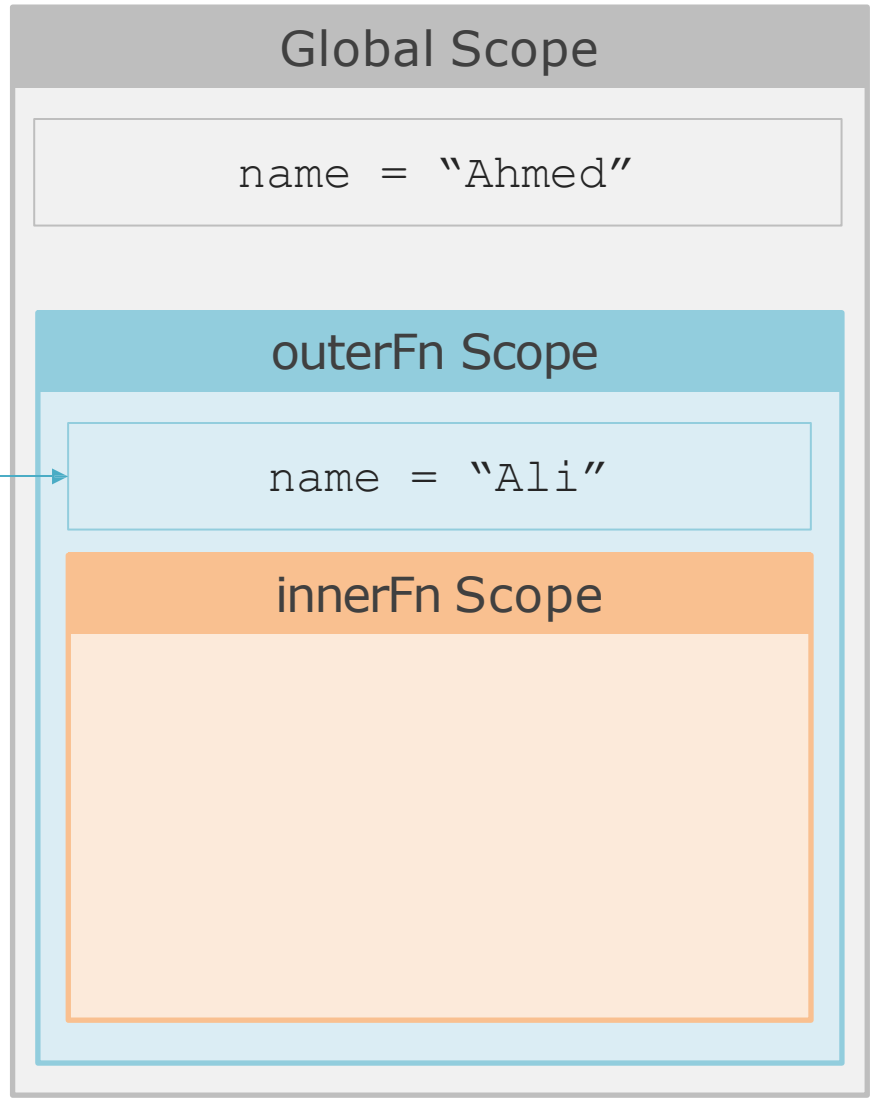
Lexical Scope

```
name = "Ahmed"
def outerFn():
        name = "Ali"
        def innerFn():
            print(name)
        innerFn()
outerFn()
```

Output:

```
Ali
```

name
???



Lexical Scope

```
name = "Ahmed"
def outerFn():
        name = "Ali"
        def innerFn():
                print(name)
        innerFn()
outerFn()
print(name)
```

Output:

```
Ali
```

Lexical Scope

```
name = "Ahmed"
def outerFn():
        name = "Ali"
        def innerFn():
                print(name)
        innerFn()
outerFn()
print(name)
```

Output:

```
Ali
```

name
???

Lexical Scope

```
name = "Ahmed"
def outerFn():
        name = "Ali"
        def innerFn():
                print(name)
        innerFn()
outerFn()
print(name)
```

Output:

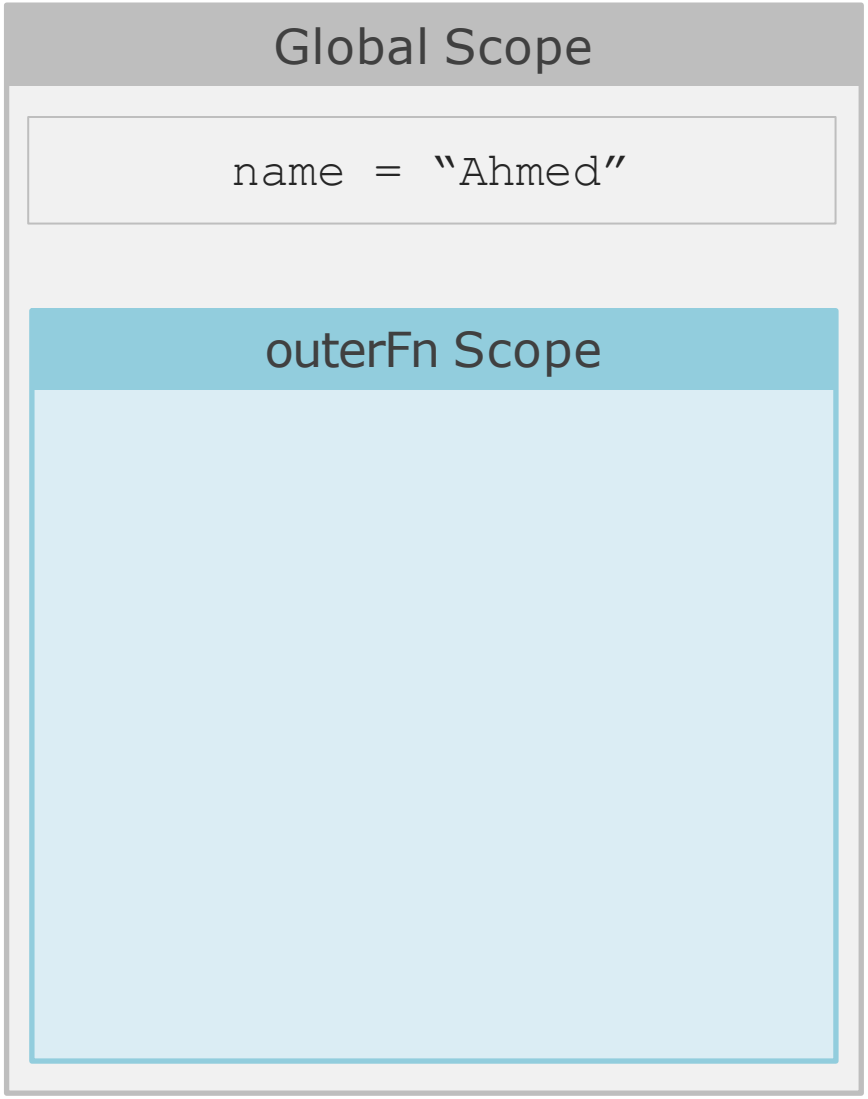
```
Ali
Ahmed
```

name
???

global Keyword

```
name = "Ahmed"
def outerFn():
        global name
        name = "Ali"
        def innerFn():
                print(name)
        innerFn()
outerFn()
```

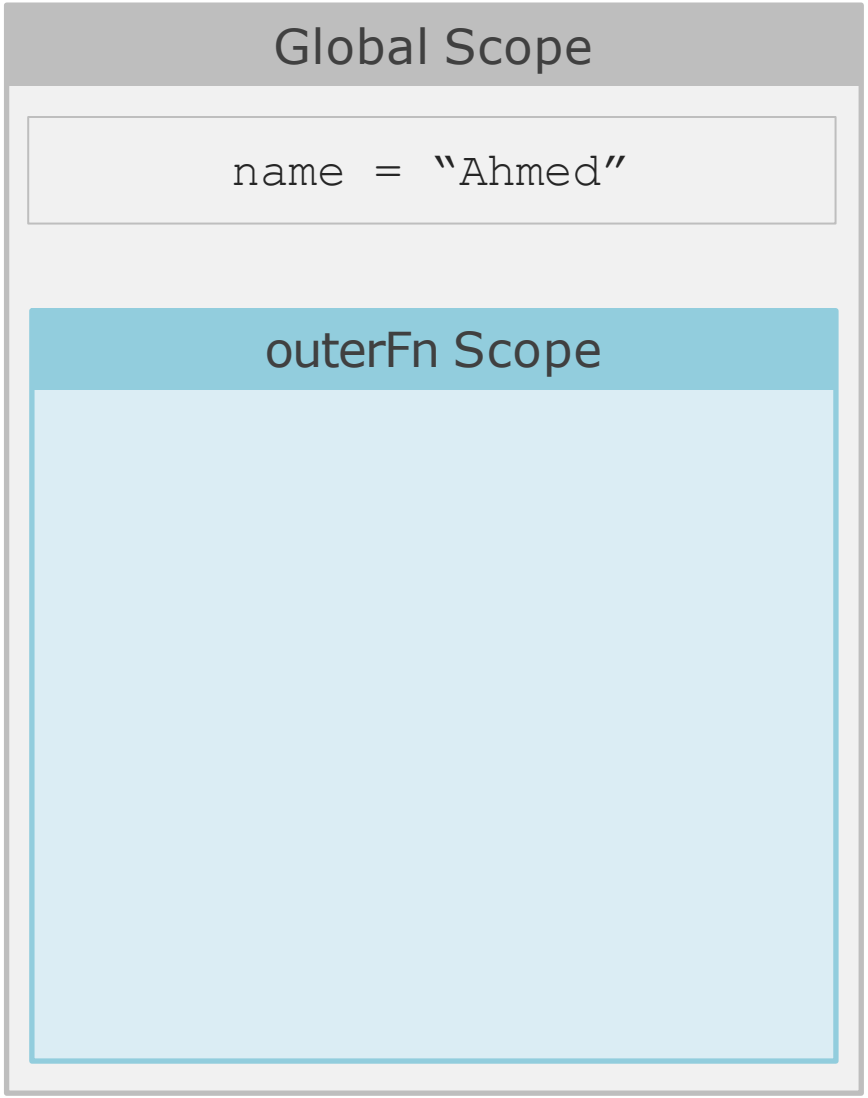
Output:



global Keyword

```
name = "Ahmed"
def outerFn():
    global name
    name = "Ali"
    def innerFn():
        print(name)
    innerFn()
outerFn()
```

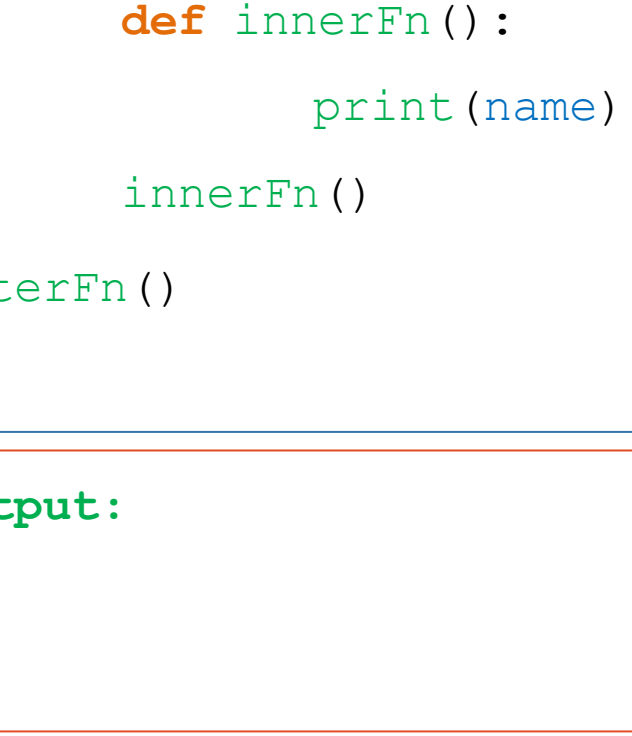
Output:



global Keyword

```
name = "Ahmed"
def outerFn():
        global name
        name = "Ali"
        def innerFn():
                print(name)
        innerFn()
outerFn()
```

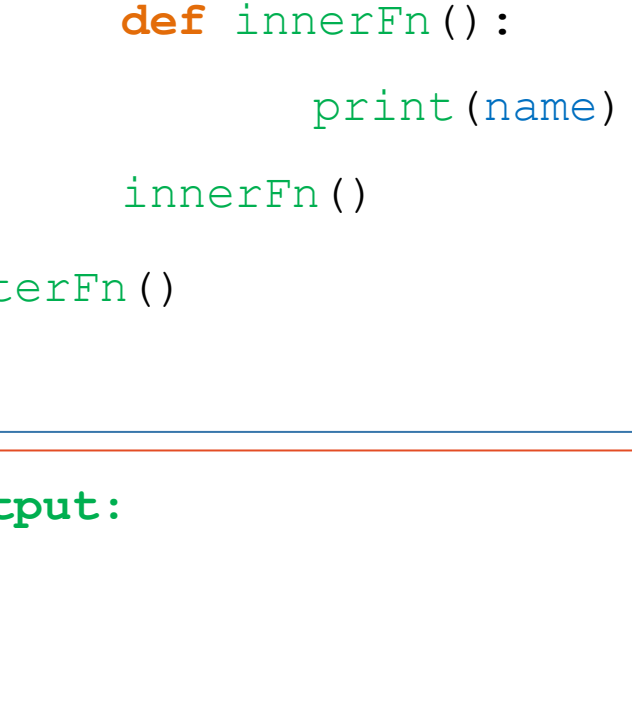
Output:



global Keyword

```
name = "Ahmed"
def outerFn():
        global name
        name = "Ali"
        def innerFn():
                print(name)
        innerFn()
outerFn()
```

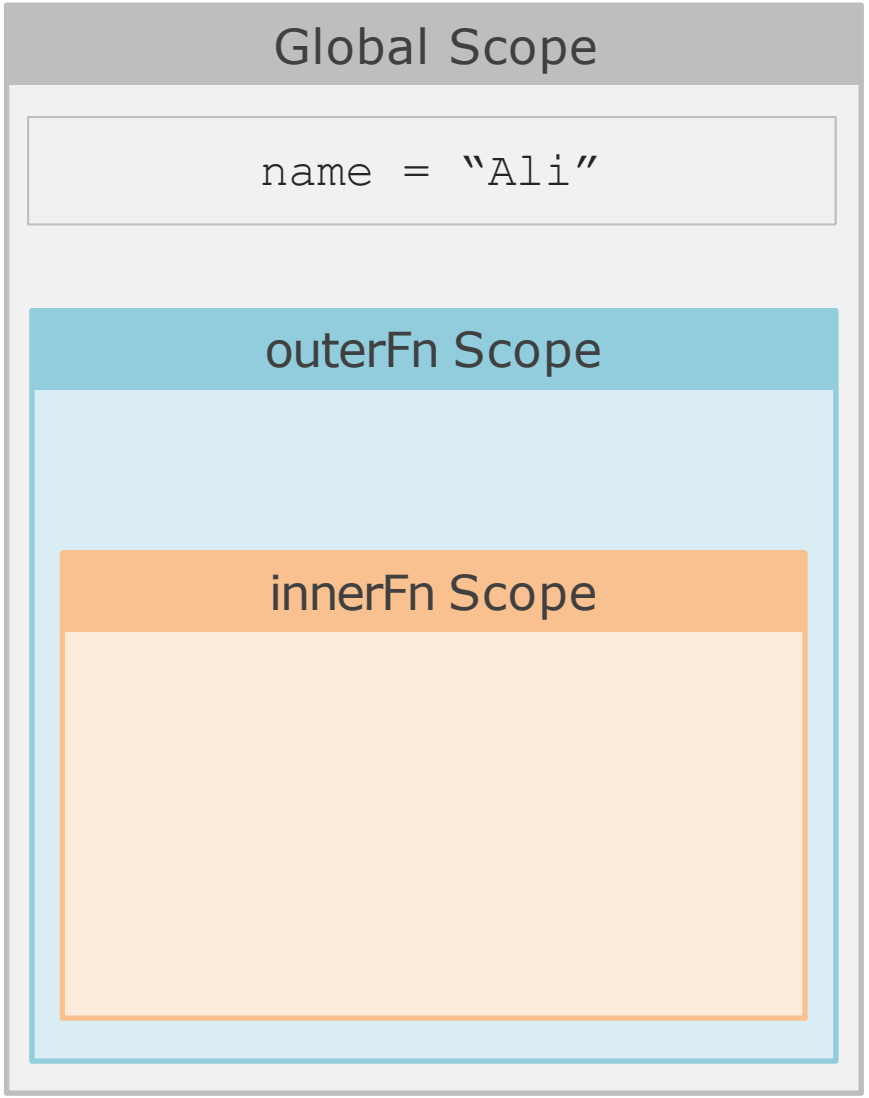
Output:



global Keyword

```
name = "Ahmed"
def outerFn():
        global name
        name = "Ali"
        def innerFn():
                print(name)
    →   innerFn()
outerFn()
```

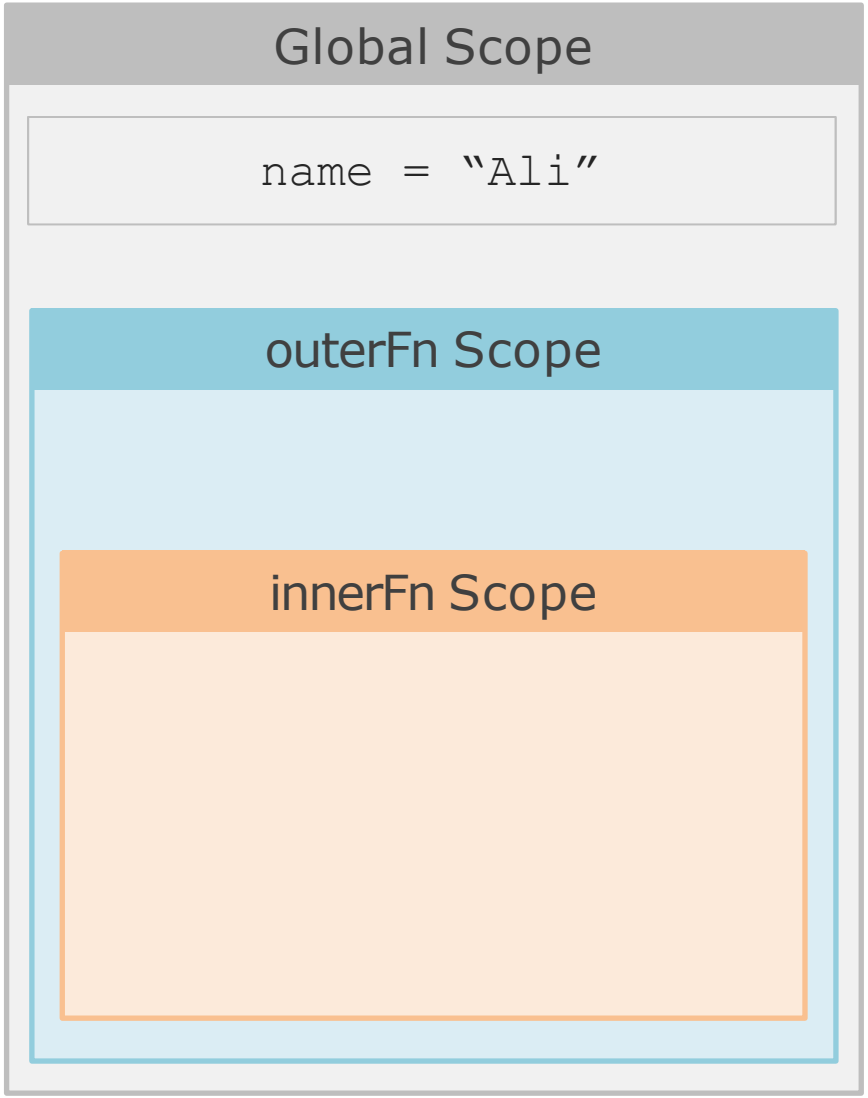
Output:



global Keyword

```
name = "Ahmed"
def outerFn():
        global name
        name = "Ali"
        def innerFn():
            print(name)
        innerFn()
outerFn()
```

Output:

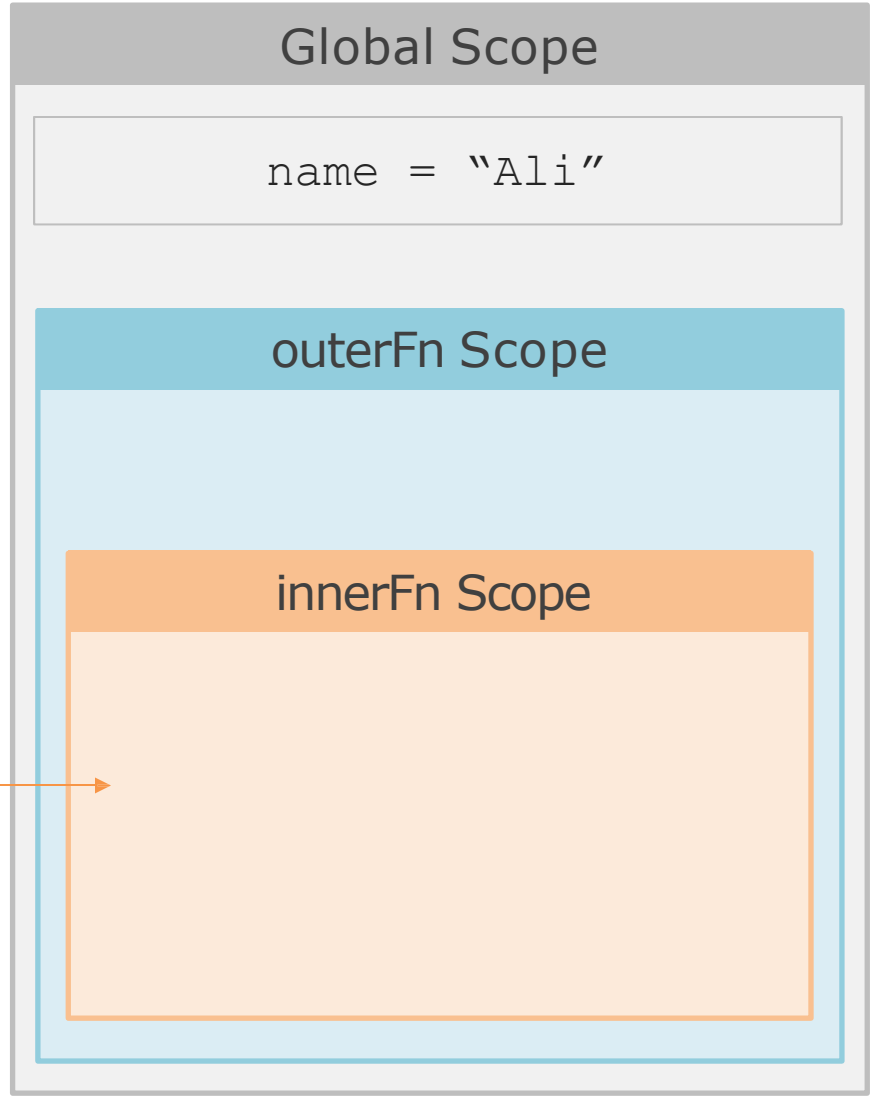


global Keyword

```
name = "Ahmed"
def outerFn():
        global name
        name = "Ali"
        def innerFn():
            print(name)
        innerFn()
outerFn()
```

Output:

name
???

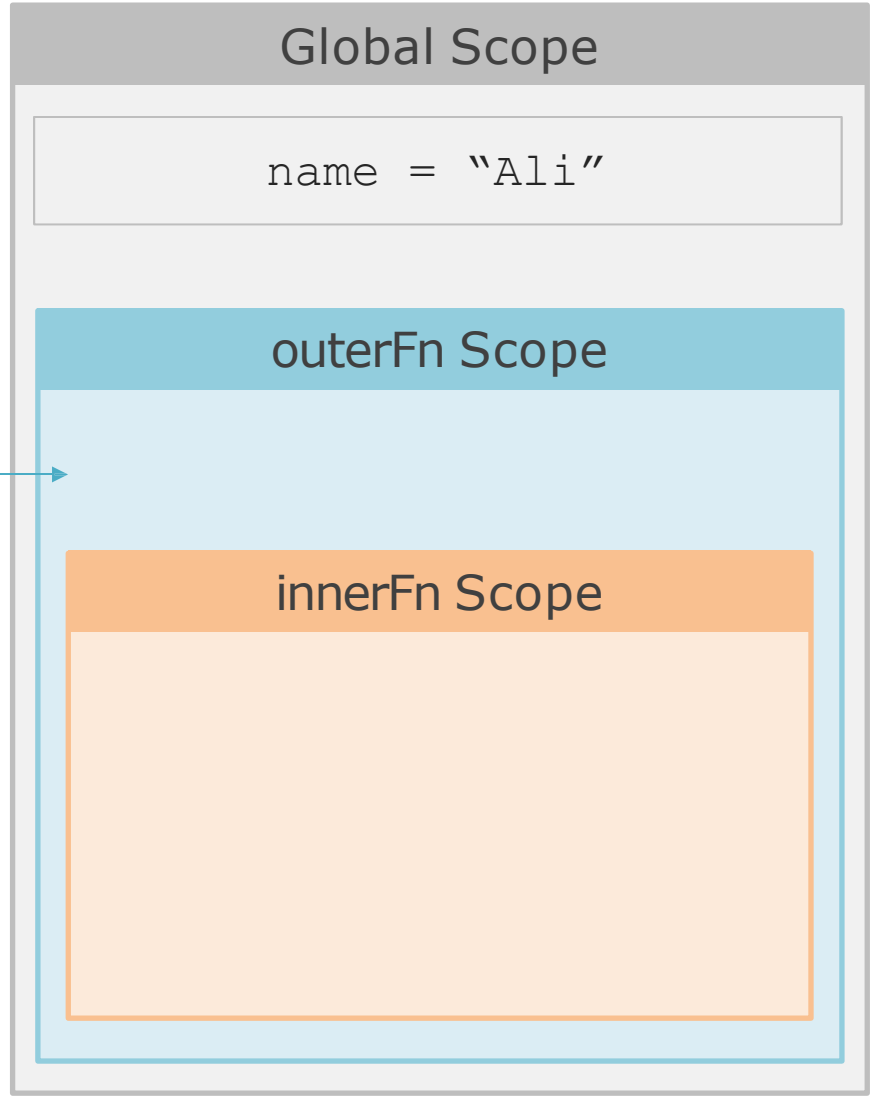


global Keyword

```
name = "Ahmed"
def outerFn():
        global name
        name = "Ali"
        def innerFn():
            print(name)
        innerFn()
outerFn()
```

Output:

name
???

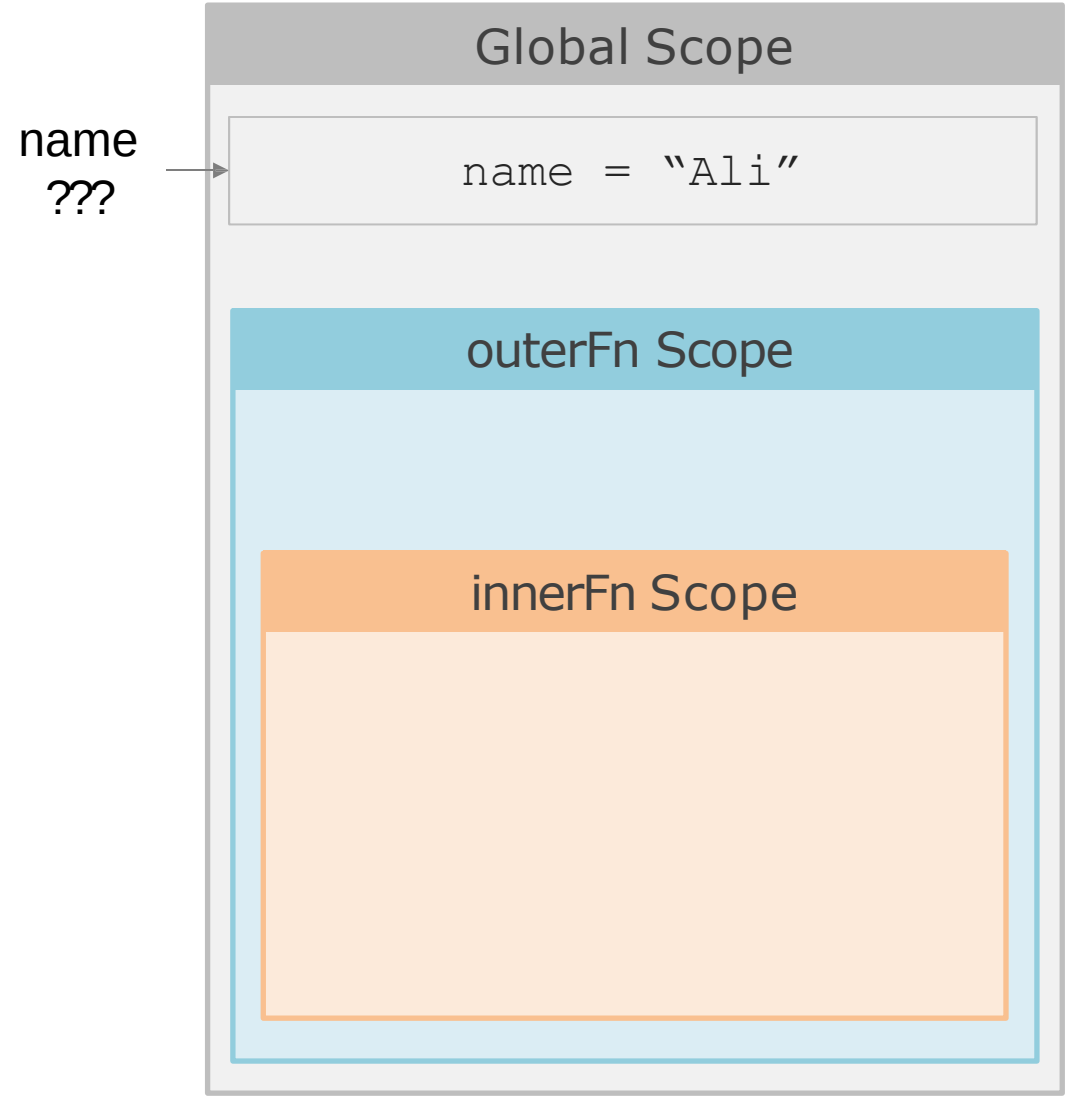


global Keyword

```
name = "Ahmed"
def outerFn():
        global name
        name = "Ali"
        def innerFn():
            print(name)
        innerFn()
outerFn()
```

Output:

name
???

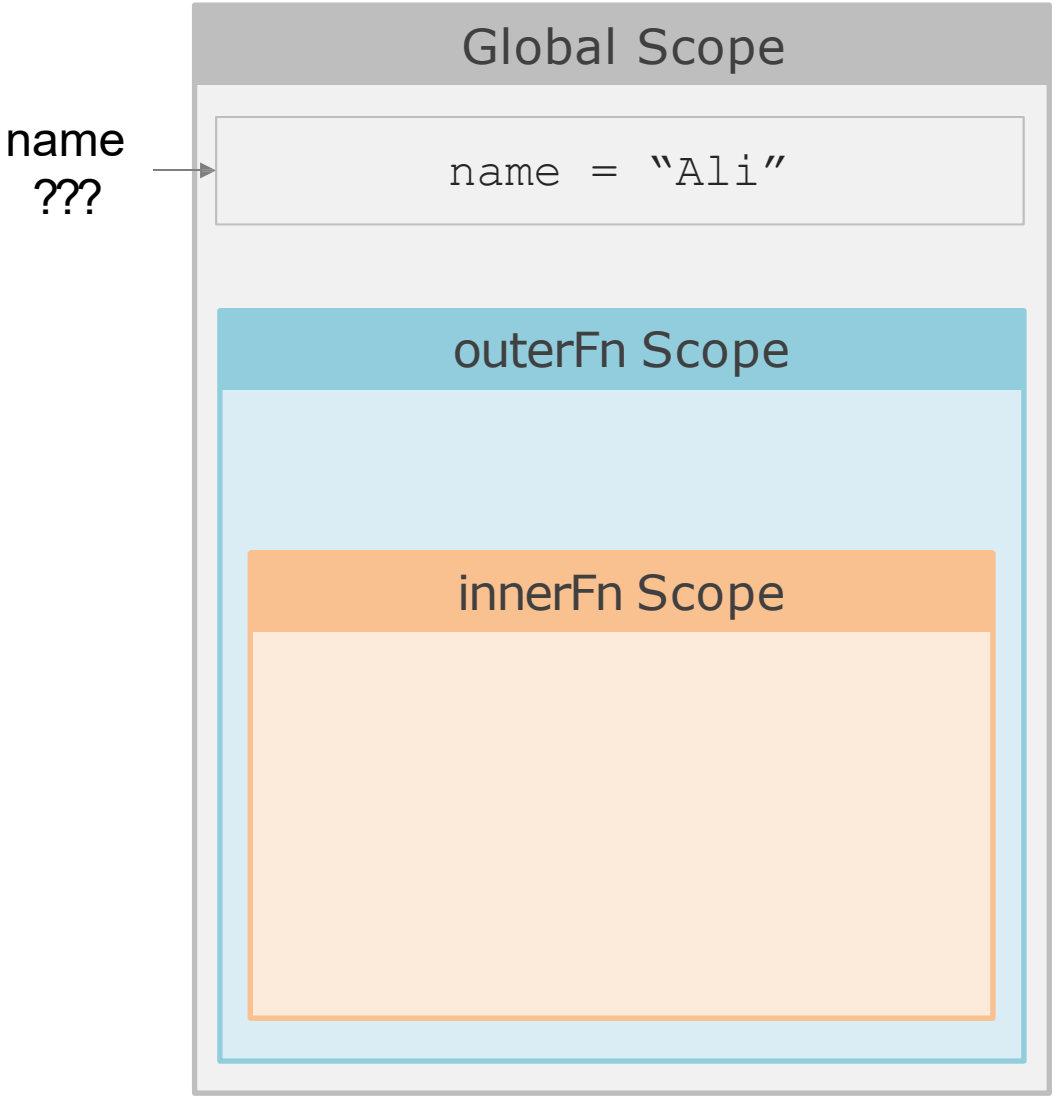


global Keyword

```
name = "Ahmed"
def outerFn():
        global name
        name = "Ali"
        def innerFn():
            print(name)
        innerFn()
outerFn()
```

Output:

```
Ali
```



global Keyword

```
name = "Ahmed"
def outerFn():
        global name
        name = "Ali"
        def innerFn():
                print(name)
        innerFn()
outerFn()
```

Output:

```
Ali
```

name
???

global Keyword

```
name = "Ahmed"
def outerFn():
        global name
        name = "Ali"
        def innerFn():
                print(name)
        innerFn()
outerFn()
print(name)
```

Output:

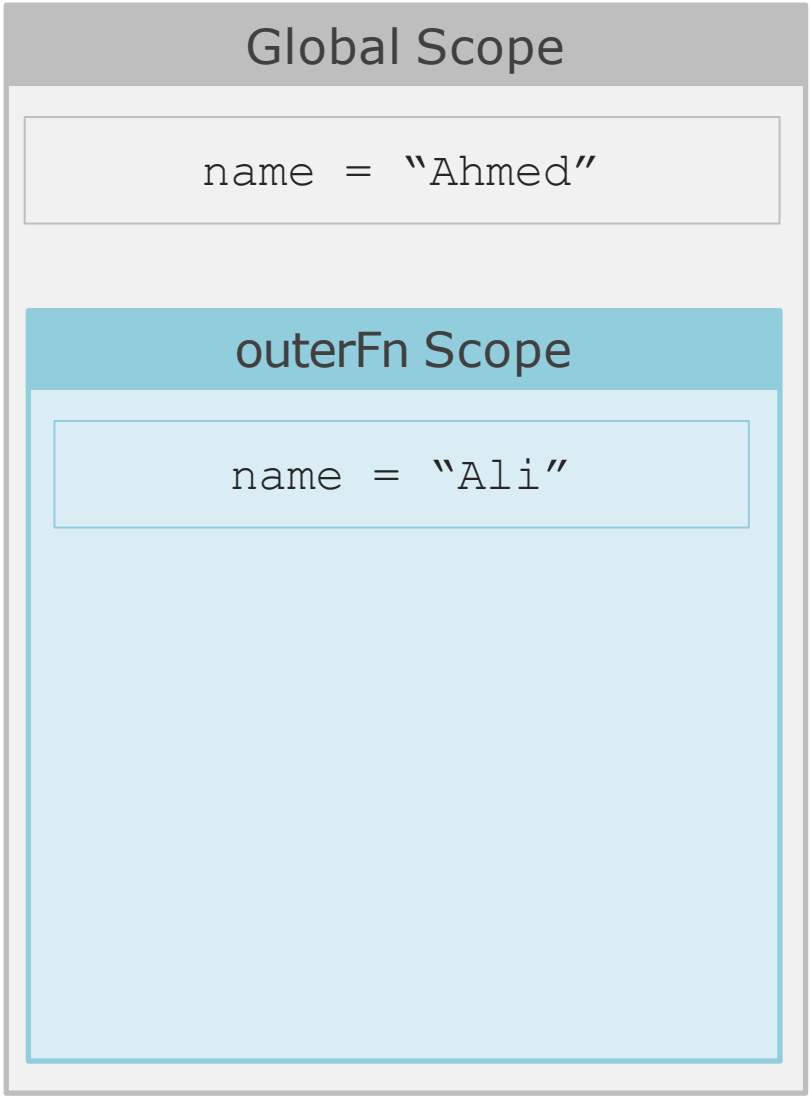
```
Ali
Ali
```

name
???

nonlocal Keyword

```
name = "Ahmed"
def outerFn():
    name = "Ali"
    def innerFn():
        nonlocal name
        print(name)
        name = "Sara"
    innerFn()
    print(name)
outerFn()
```

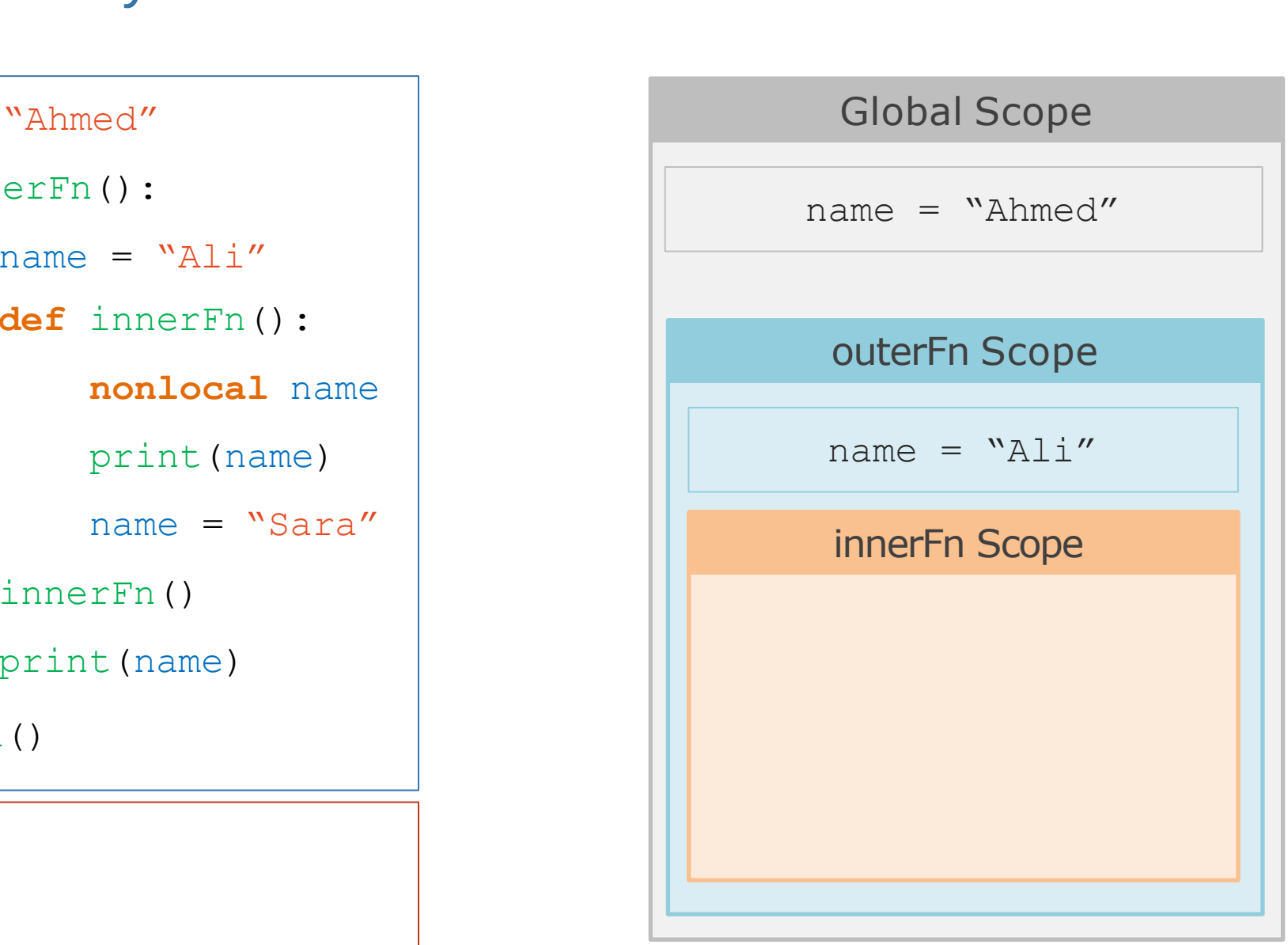
Output:



nonlocal Keyword

```
name = "Ahmed"
def outerFn():
        name = "Ali"
        def innerFn():
            nonlocal name
            print(name)
            name = "Sara"
    →   innerFn()
        print(name)
outerFn()
```

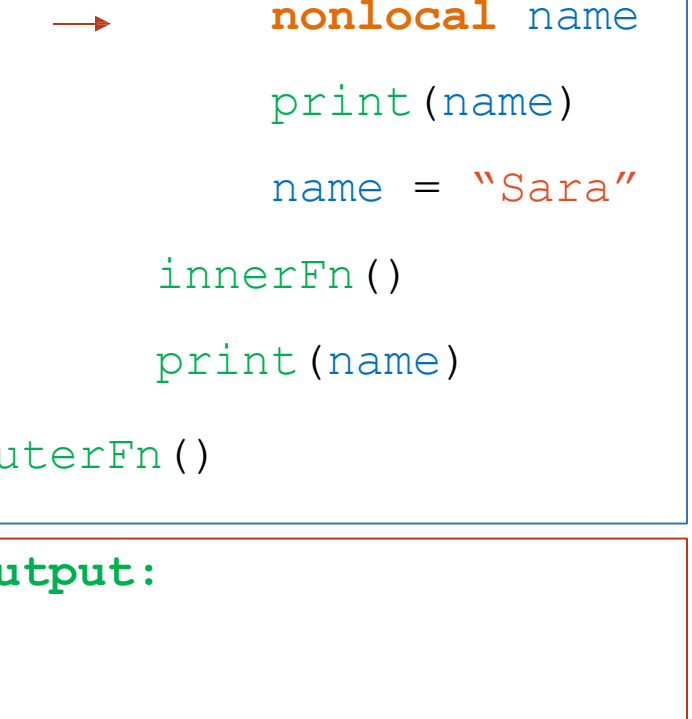
Output:



nonlocal Keyword

```
name = "Ahmed"
def outerFn():
        name = "Ali"
        def innerFn():
            nonlocal name
            print(name)
            name = "Sara"
        innerFn()
        print(name)
outerFn()
```

Output:

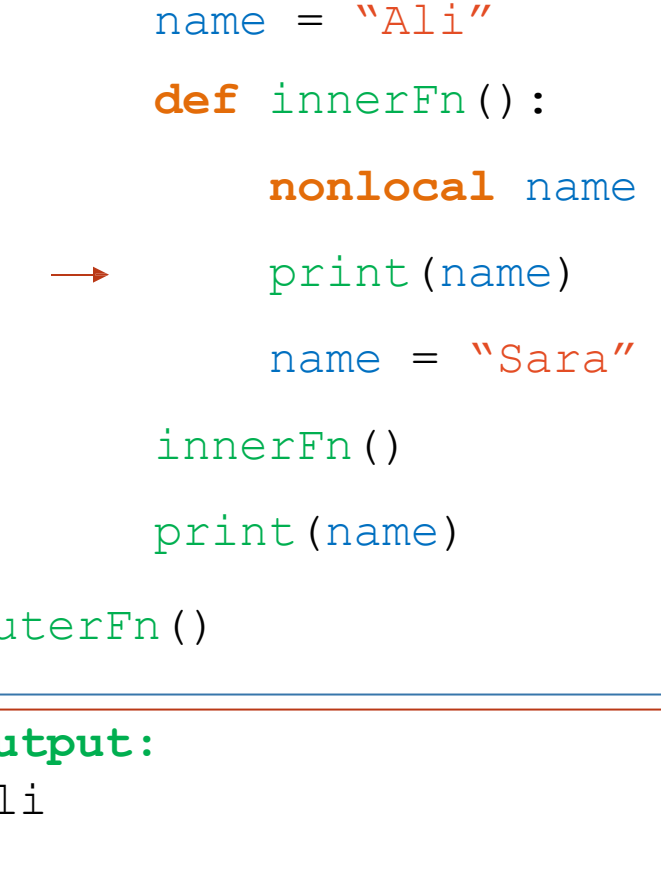


nonlocal Keyword

```
name = "Ahmed"
def outerFn():
        name = "Ali"
        def innerFn():
            nonlocal name
            print(name)
            name = "Sara"
        innerFn()
        print(name)
outerFn()
```

Output:

```
Ali
```

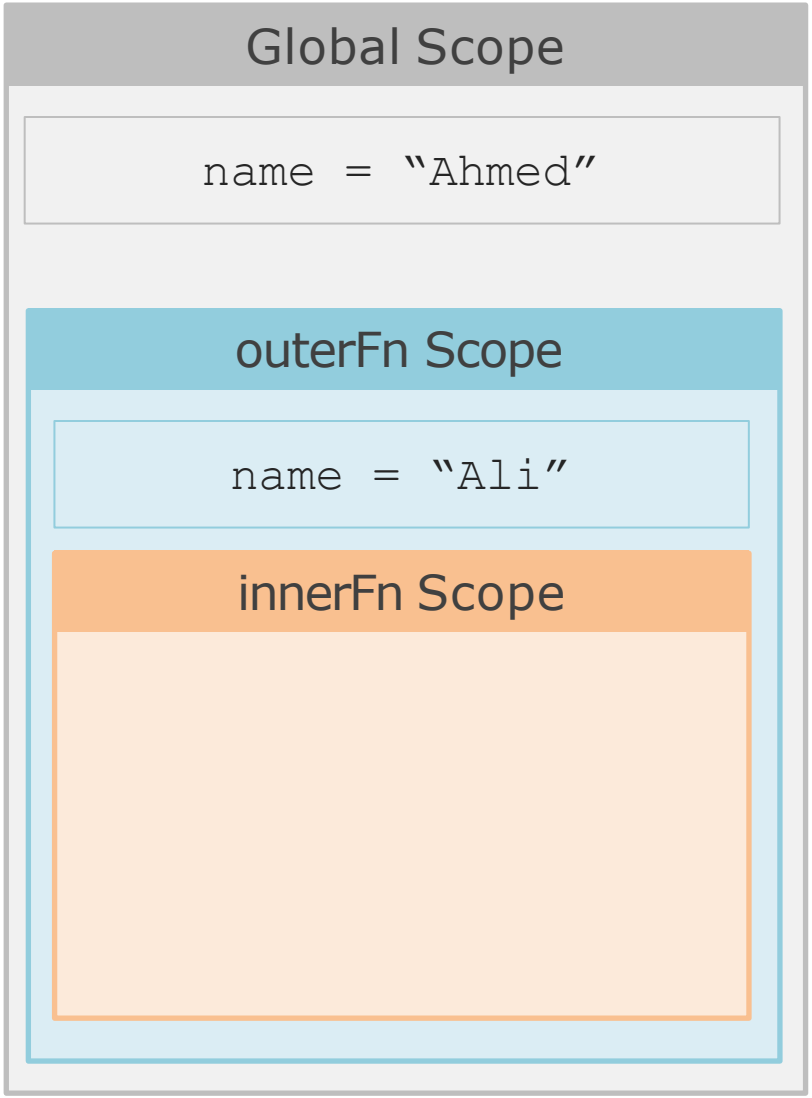


nonlocal Keyword

```
name = "Ahmed"
def outerFn():
        name = "Ali"
        def innerFn():
            nonlocal name
            print(name)
            name = "Sara"
        innerFn()
        print(name)
outerFn()
```

Output:

```
Ali
```

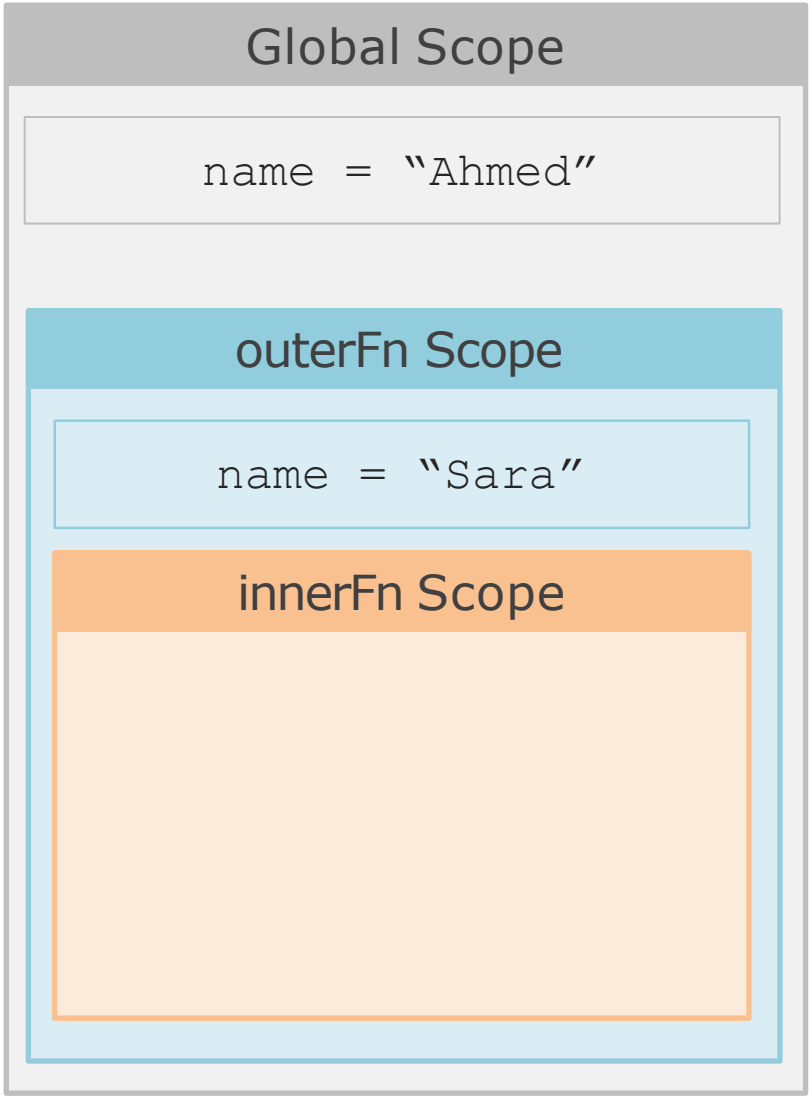


nonlocal Keyword

```
name = "Ahmed"
def outerFn():
        name = "Ali"
        def innerFn():
            nonlocal name
            print(name)
            name = "Sara"
        innerFn()
        print(name)
outerFn()
```

Output:

```
Ali
```

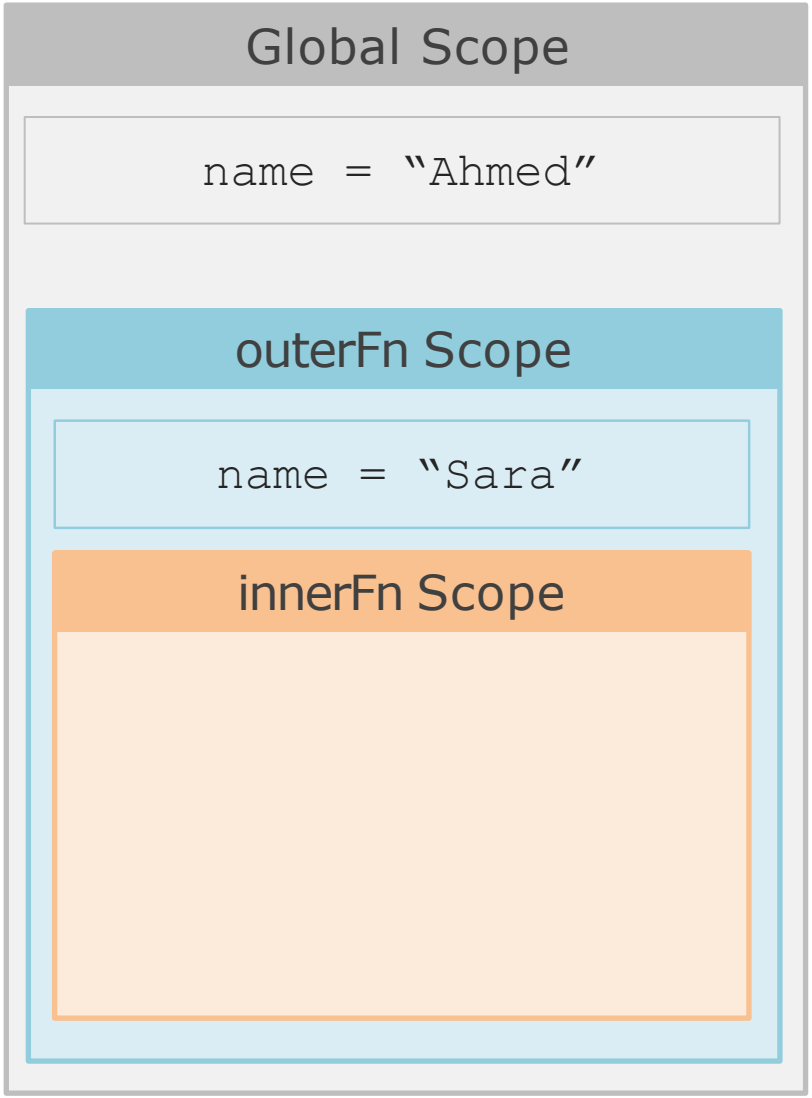


nonlocal Keyword

```
name = "Ahmed"
def outerFn():
        name = "Ali"
        def innerFn():
            nonlocal name
            print(name)
            name = "Sara"
        innerFn()
        print(name)
outerFn()
```

Output:

```
Ali
```

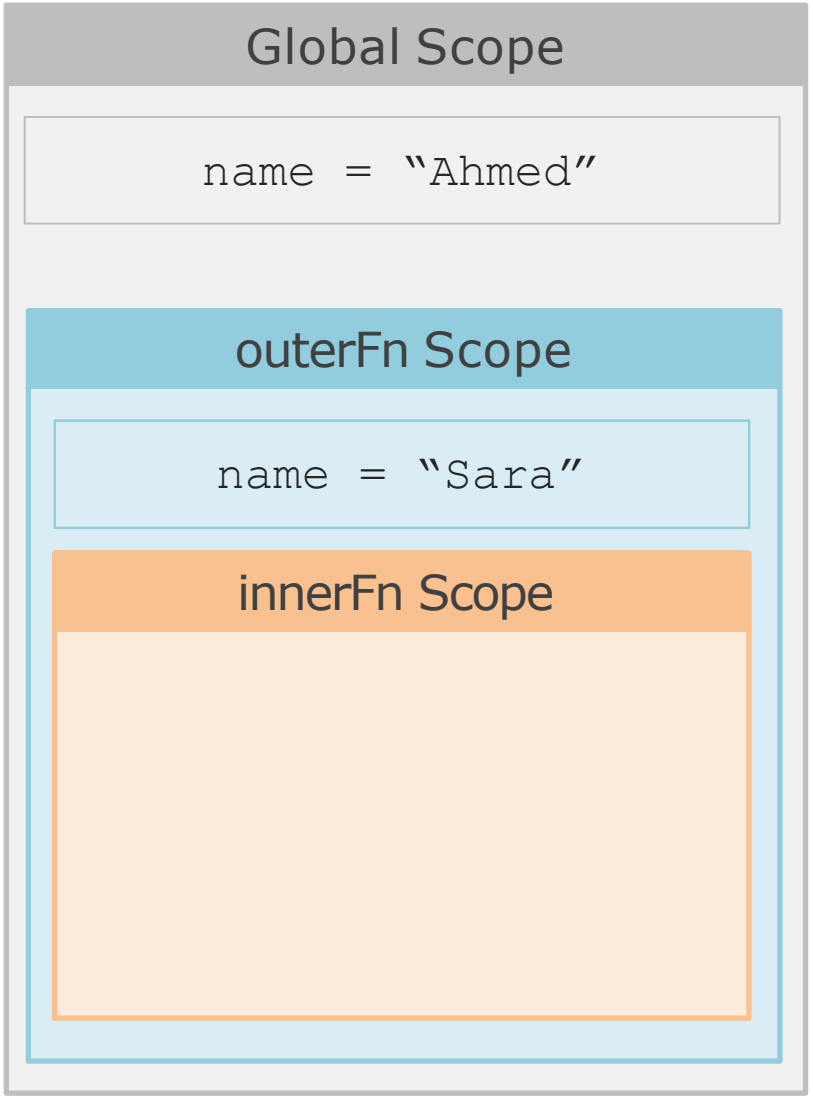


nonlocal Keyword

```
name = "Ahmed"
def outerFn():
        name = "Ali"
        def innerFn():
            nonlocal name
            print(name)
            name = "Sara"
        innerFn()
        print(name)
outerFn()
```

Output:

```
Ali
Sara
```



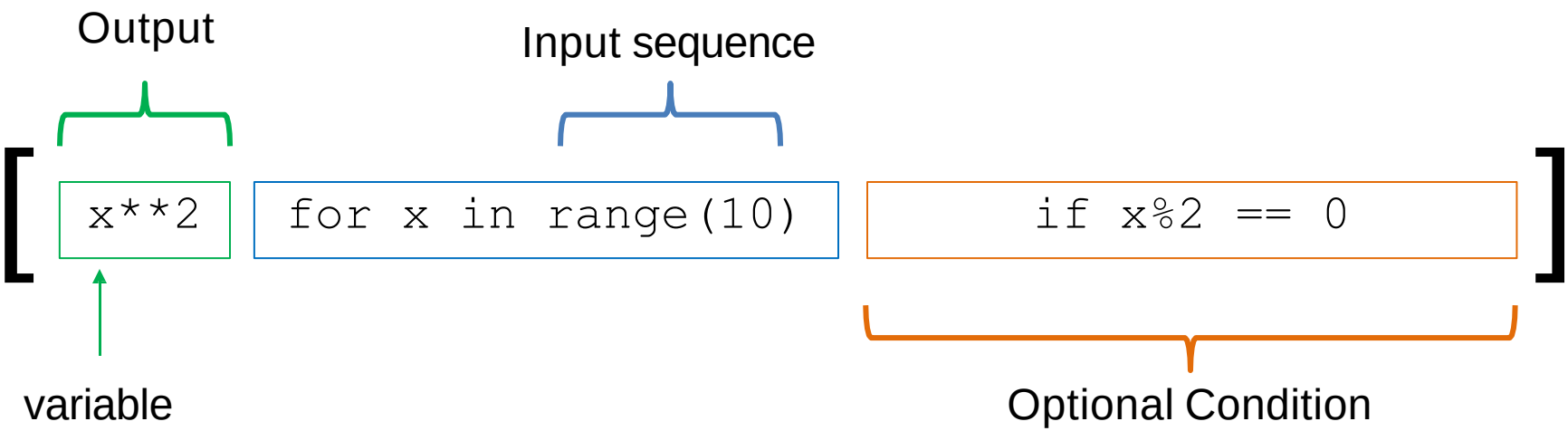
Tips and Tricks

Sequence Unpacking

```
l = [1,13,3,7]

a,b,c,d = l

# a=1,b=13,c=3,d=7

a,*b,c = l

# a=1,b=[13,3],c=7
```

List Comprehension

It is an easy method to construct a list




```
L = [ x**2 for x in range(10) if x%2 == 0 ]
#output: [0, 4, 16, 36, 64]
```

enumerate Function

```
languages = [“JavaScript”, “Python”, “Java”]
for i , l in enumerate(languages):
      print(“Element Value: ” , l, end=“, ”)
      print(“Element Index: ” , i)
```

```
Output:
Element Value: JavaScript, Element index: 0
Element Value: Python, Element index: 1
Element Value: Java, Element index: 2
```

all & any

all check if all items in an iterable are truthy value.
any check if one item at least in an iterable is truthy value.

```
L = [0,5,9,7,8]
all(L)                    #False
any(L)                    #True
```

Thank You