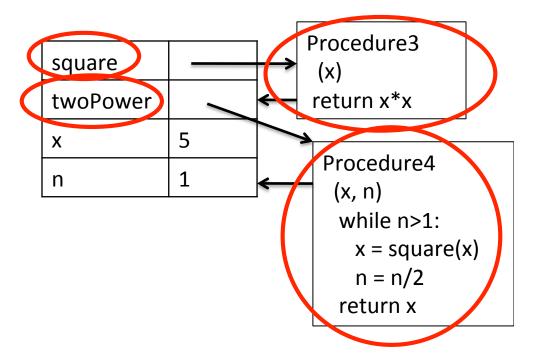
Another example

```
def square(x):
    return x*x
def twoPower(x, n):
    while n > 1:
        x = square(x)
        n = n/2
    return x
```

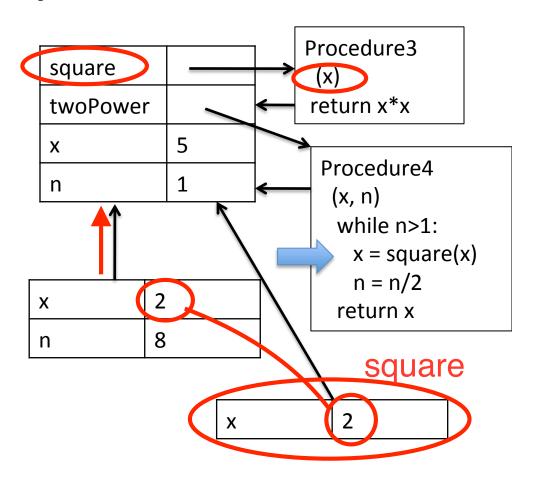
```
def square(x):
    return x*x
def twoPower(x, n):
    while n > 1:
        x = square(x)
        n = n/2
    return x
x = 5
n = 1
print(twoPower(2,8))
```



```
Procedure3
def square(x):
                                 square
                                                     (x)
     return x*x
                                twoPower
                                                     return x*x
                                           5
                                 Χ
                                                     Procedure4
def twoPower(x, n):
                                 n
                                                      (x, n)
     while n > 1:
                                                       while n>1:
                                                        x = square(x)
          x = square(x)
                                                        n = n/2
          n = n/2
                                         2
                                                       return x
     return x
                                         8
x = 5
n = 1
```

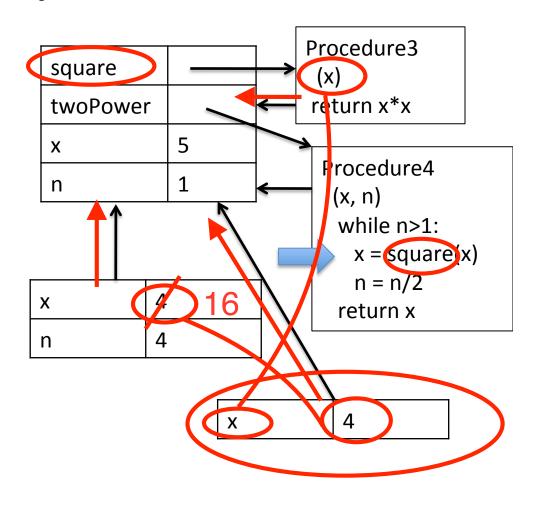
print(twoPower(208)

```
def square(x):
    return x*x
def twoPower(x, n):
    while n > 1:
        x = square(x)
        n = n/2
    return x
n = 1
print(twoPower(2,8))
```

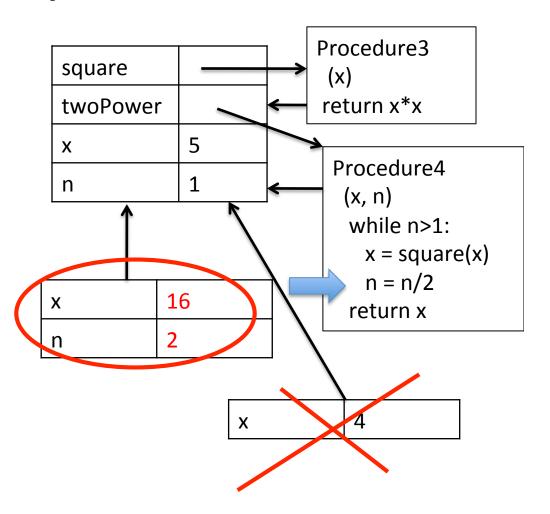


```
Procedure3
def square(x):
                                 square
                                                     (x)
     return x*x
                                                    return x*x
                                 twoPower
                                          5
                                 Χ
                                                     Procedure4
def twoPower(x, n):
                                 n
                                                      (x, n)
     while n > 1:
                                                      while n>1:
                                                       x = square(x)
          x = square(x)
                                                       n = n/2
          n = n/2
                                                      return x
     return x
                                         4
n = 1
print(twoPower(2,8))
```

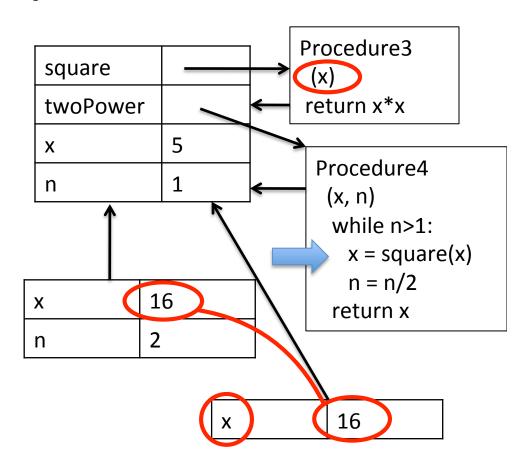
```
def square(x):
    return x*x
def twoPower(x, n):
    while n > 1:
        x = square(x)
        n = n/2
    return x
x = 5
n = 1
print(twoPower(2,8))
```



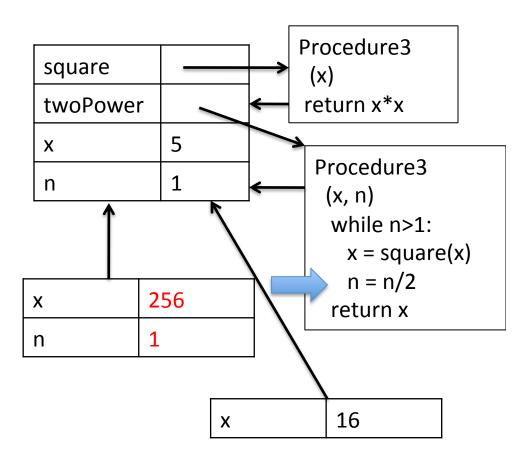
```
def square(x):
    return x*x
def twoPower(x, n):
    while n > 1:
        x = square(x)
        n = n/2
    return x
x = 5
n = 1
print(twoPower(2,8))
```



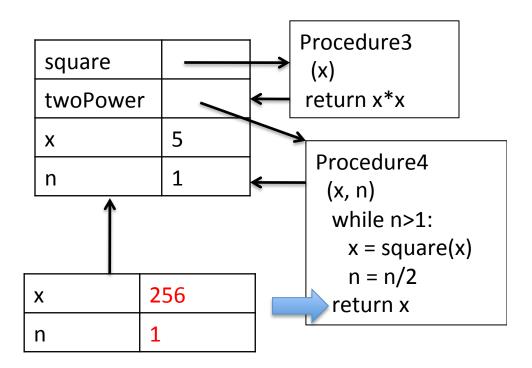
```
def square(x):
    return x*x
def twoPower(x, n):
    while n > 1:
        x = square(x)
        n = n/2
    return x
x = 5
n = 1
print(twoPower(2,8))
```



```
def square(x):
    return x*x
def twoPower(x, n):
    while n > 1:
        x = square(x)
        n = n/2
    return x
x = 5
n = 1
print(twoPower(2,8))
```



```
def square(x):
    return x*x
def twoPower(x, n):
    while n > 1:
        x = square(x)
        n = n/2
    return x
x = 5
n = 1
print(twoPower(2,8))
256
```



Some observations

Same variable in different procedures

- Notice how each call to square created a new frame, with a local binding for x
- The value of x in the global environment was never confused with values within frames from function calls
- The value of x used by the call to square is different from the binding for x in the call to twoPower
- The rules we described can be followed mechanically to determine scoping of variables