Lecture 8, 10 September 2024

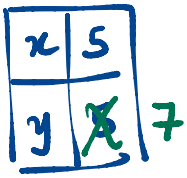
Madhavan Mukund

https://www.cmi.ac.in/~madhavan

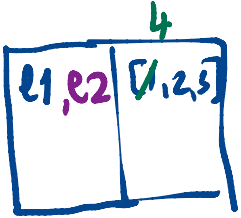
Programming and Data Structures with Python

Mutable & Immutable values

```
x = 5
y = x
y = 7 ✓
```




```
l1 = [1,2,3]
l2 = l1 ✓
l2[0] = 4 ✓
```



Mutable vs Immutable

Lists
Dictionaries

} Mutable

int
float
String
tuple
bool

} Immutable

Copying a list?

Slices create fresh copies

l[i:j]

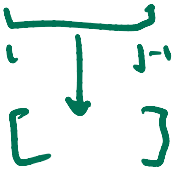
slice(l, i, j)

↓

new list

l — [ ]

l[i:j]



```
l2 = l1[0:len(l1)]
```

fresh copy of l1 in l2

```
l2 = l1[:]
```

full slice

Passing values to functions

```
def add(m,n):
    return(m+n)
```

```
x=5
y=7
z=add(x,y)
```

m+n

m = x
n = y

z = (m+n)

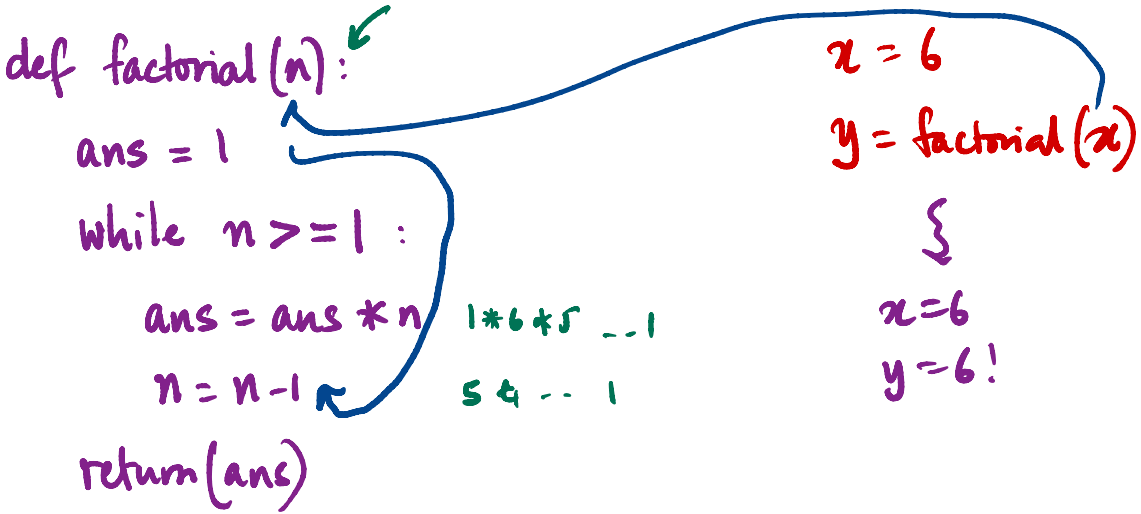
```
def swap(x,y):
    (x,y) = (y,x)
    return
```

m=5
n=7

swap(m,n)

x = m
y = n

no effect on m,n


def factorial(n):
ans = 1
while n>=1 :
ans = ans * n
n = n-1
return(ans)
1*6*5 ... 1
5 & ... 1
x = 6
y = factorial(x)
{
x=6
y=6!

Names in functions are local (for immutable values)

```
def printcount():
    print(count)
```

?

undefined in function

IGNORE

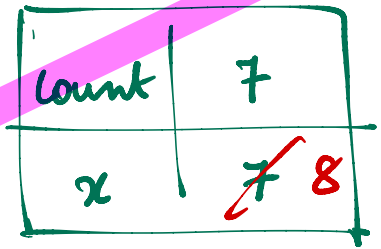
```
count = 7
printcount(count)
def printcount(x):
    print(x)
```

x = count

```
def increment(x):
    x = x+1
    return
```

```
count = 7
increment(count)
```

IGNORE



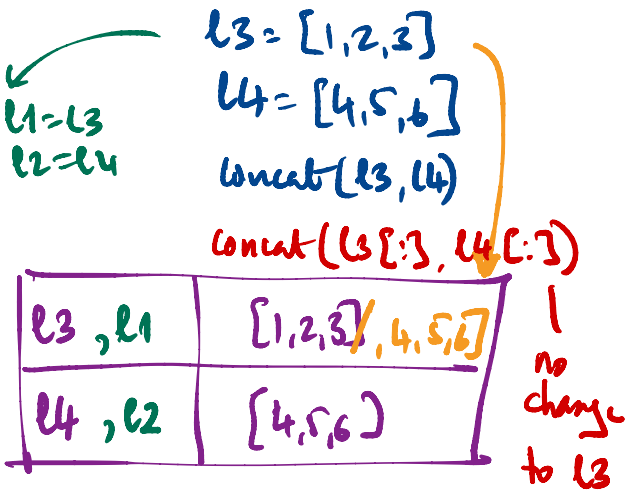
```
def increment(x):
    return(x+1)
```

```
count = increment(count)
```

IGNORE

Passing mutable values to functions

```
def concat(l1, l2):
    l1.extend(l2)
    return
```



```
def concat2(l1,l2):
    l1 = l1+l2
    return
```

l1+l2 creates a new list

```
l3 = [1,2,3]
l4 = [4,5,6]
concat2(l3,l4)
```

l3 is ? No change

l1 & l3 are no longer the same list

```
def myappend(l,x):
    l = l.append(x)
    return(l)
```

`l=l1`

```
l1 = [1,2]
l1 = myappend(l1,3)
```

return value of `l.append(x)` is assigned to `l`

None

```
def myappend(l,x):
    l = l+[x]
    return(l)
```

`x == y`

x & y have same value

```
l1 = [1,2,3]
l2 = l1
l3 = l1[:]
```

`l1 == l2 ?` ✓
`l1 == l3 ?` ✓

] No diff

`l1 is l2` ✓

`l1 is l3` ✗

```
def addprime(x):
    primelist.append(x)
    return
```

?

mutable values are
Implicitly global

```
primelist = []
.
.
.
if prime(x):
    addprime(x)
```