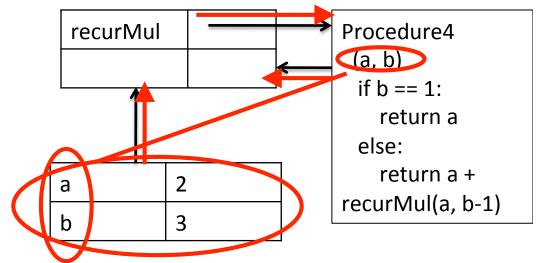
```
def recurMul(a, b):
    if b == 1:
        return a
    else:
        return a +
    recurMul(a, b-1)
```

```
def recurMul(a, b):
    if b == 1:
        return a
    else:
        return a +
    recurMul(a, b-1)
```



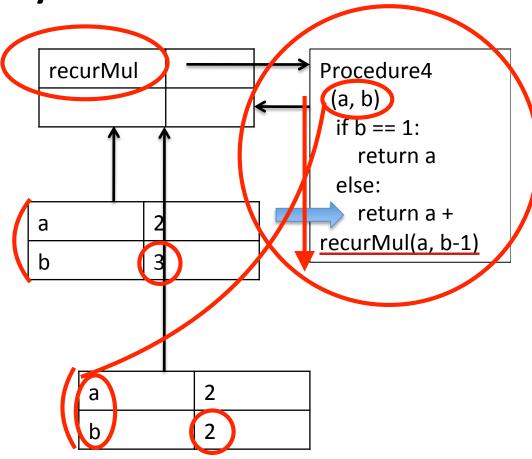


```
recurMul
                                                    Procedure4
def recurMul(a, b):
                                                     (a, b)
     if b == 1:
                                                     if b == 1:
                                                       return a
         return a
                                                     else:
     else:
                                                      return a +
                                        2
                               a
                                                    recurMui a, b-1)
         return a +
                               b
                                        3
  recurMul(a, b-1)
```

recurMul(2, 3)

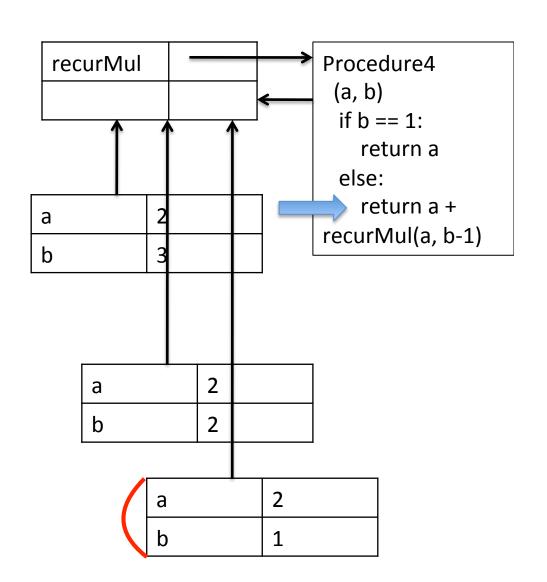
```
def recurMul(a, b):
    if b == 1:
        return a
    else:
        return a +
    recurMul(a, b-1)

recurMul(2, 3)
```



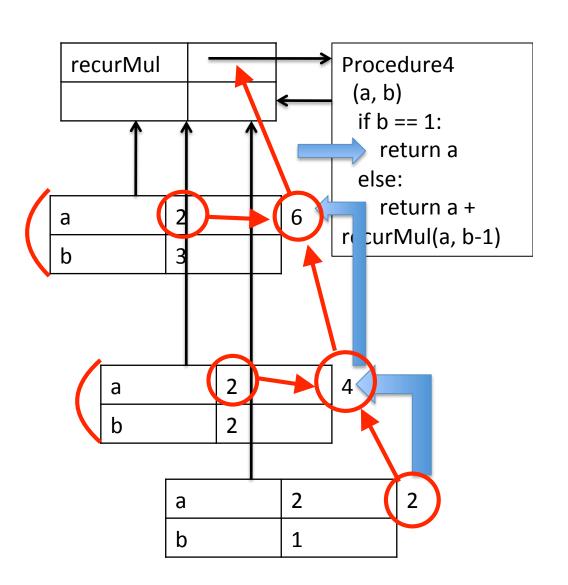
```
def recurMul(a, b):
    if b == 1:
        return a
    else:
        return a +
    recurMul(a, b-1)

recurMul(2, 3)
```



```
def recurMul(a, b):
    if b == 1:
        return a
    else:
        return a +
    recurMul(a, b-1)

recurMul(2, 3)
```



Some observations

- Each recursive call to a function creates its own environment, with local scoping of variables
- Bindings for variable in each frame distinct, and not changed by recursive call
- Flow of control will pass back to earlier frame once function call returns value