Environments to understand bindings

- Environments are formalism for tracking bindings of variables and values
- Assignments pair name and value in environment
- Asking for value of name just looks up in current environment
- Python shell is default (or global) environment
- Definitions pair function name with details of function

```
x = 5
p = 3

result = 1

for turn in range(p):
    print('iteration: ' + str(turn) + 'current result: ' +
   str(result))
    result = result * x
```

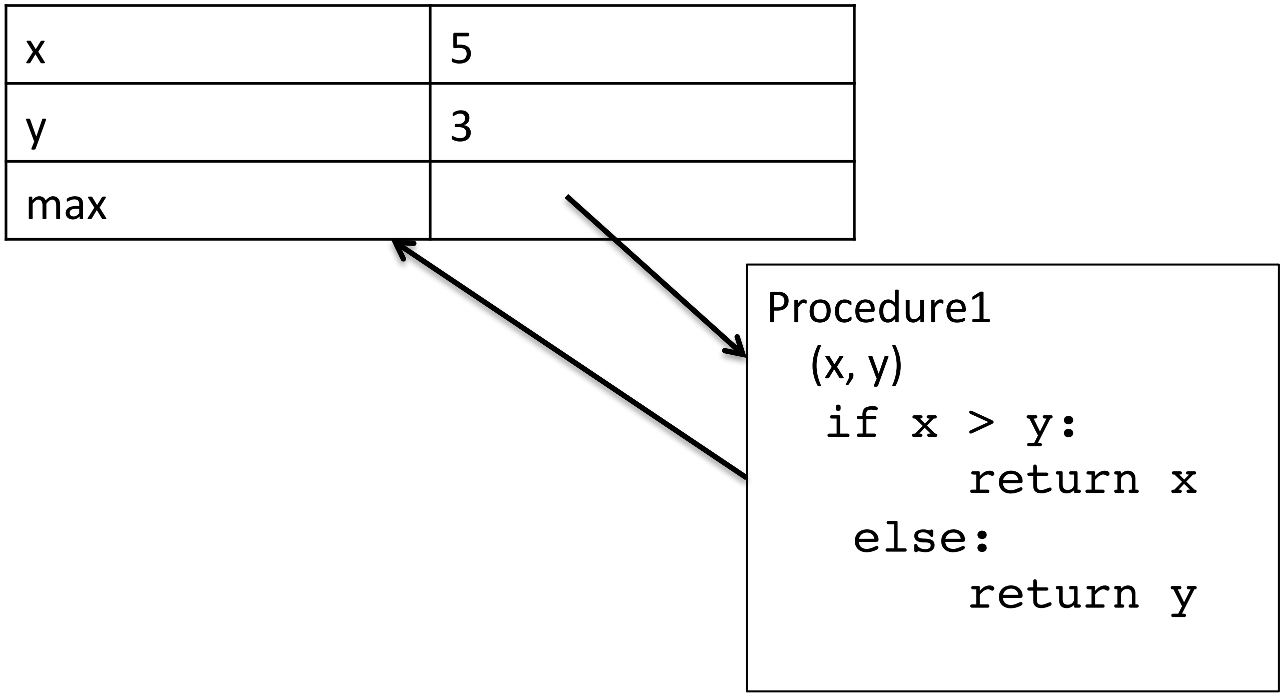
x	5
p	3
result	1

```python
x = 5
p = 3

result = 1

for turn in range(p):
    print('iteration: ' + str(turn) + 'current result: ' +
  str(result))
    result = result * x
```

x	5
p	3
result	125

Back to functions

```
x = 5
y = 3

def max(x, y):
    if x > y:
        return x
    else:
        return y
```



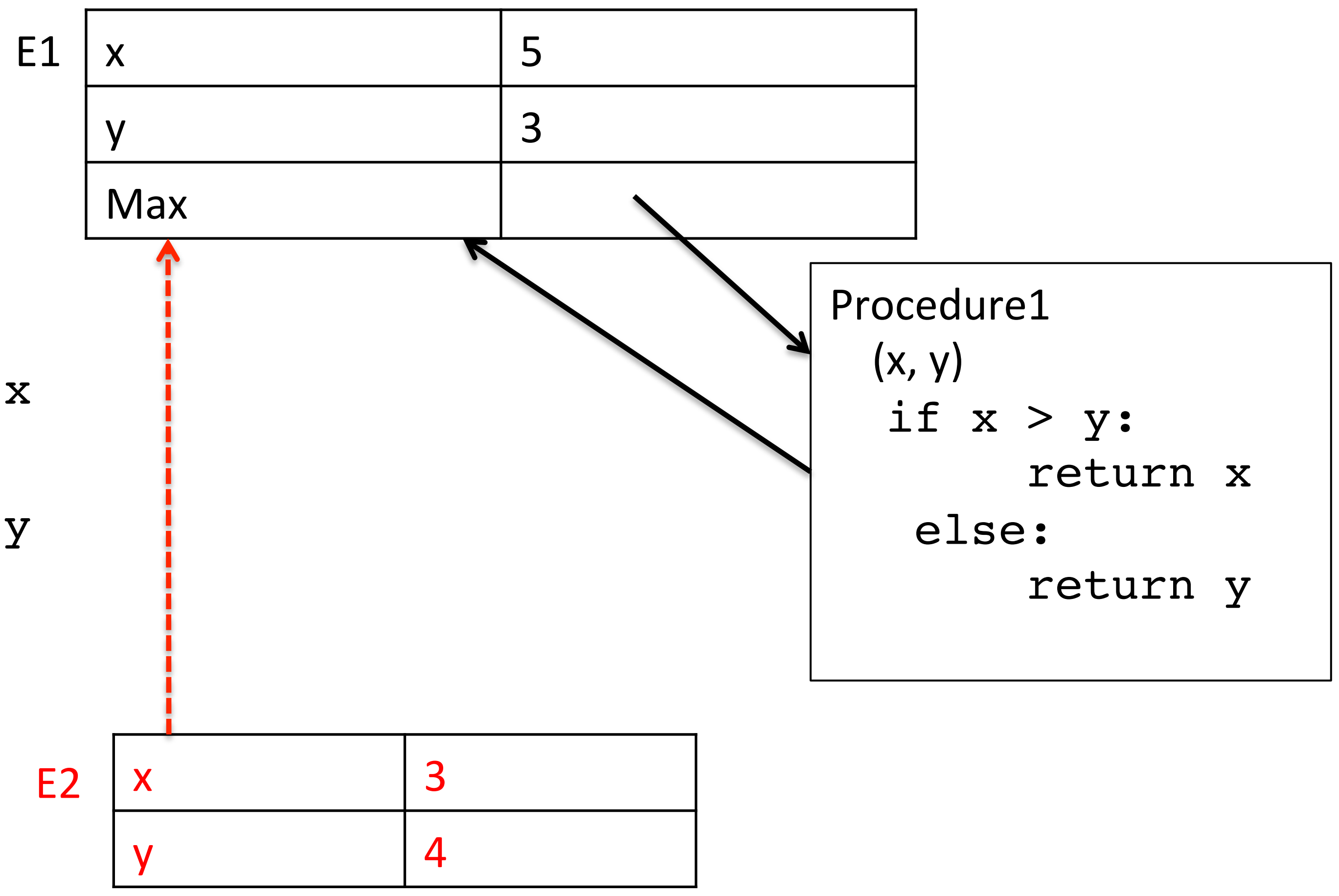
x	5
y	3
max	

When we call a function

- Want to evaluate `<expr0>(<expr1>, …, <exprn>)`
- First evaluate `<expr0>,` which looks up procedure object in environment
- Then evaluate each of the other `<expri>` to get values of parameters
- Bind parameter names in procedure object to values of arguments in a new frame, which has as a parent the environment in which procedure was defined
- Evaluate body of procedure relative to this new frame

When we call the function

```
x = 5
y = 3

def max(x, y):
    if x > y:
        return x
    else:
        return y

z = max(3, 4)
```



E1
x
5
y
3
Max
Procedure1
(x, y)
if x > y:
return x
else:
return y
E2
x
3
y
4