Section #2: Function, Control

Control

Call Expression:

$\underbrace{\text{add}}_{\text{operator}}\,(\underbrace{2,\ 3}_{\text{operand}})$

1. evaluate operator and operand
2. apply function to values

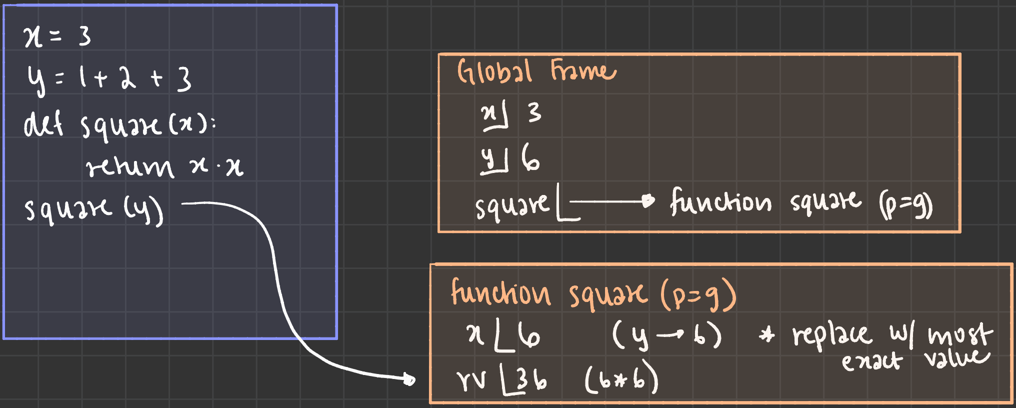
Define Function:

def $\underbrace{\text{<name>}}_{\text{function name}}$ (< $\underbrace{\text{formal parameter}}_{\text{function parameters}}$ >)

return < $\underbrace{\text{return expressions}}_{\text{what function returns}}$ >

Environment Diagrams:

- first start with Global Frame outlining just variable and function names
- only if call to function is made, do you open a new frame



```
x = 3
y = 1 + 2 + 3
def square(x):
    return x · x
square(y)
```

Print vs. Return

Print: will display value but will return None

Return: returns the proper value

understand this!

print (4) → although prints 4, return value is None!

If:

```
if blah:          ← checks this ①
    do this
elif blah2:       ← if above isn't true checks this ②
    do that
else:             ← if all cases aren't true, does this ③
    do thisback;
```

while:

```
while (a conditional):    ← checks if this conditional
                            is correct
  ~ does all of this      ← will keep doing
    multiple times ~        everything inside until
return blah                 while condition is true
```

* can create infinite loops
if not careful! (when conditional always true)