CS1110

Lecture 6: Function calls

Announcements

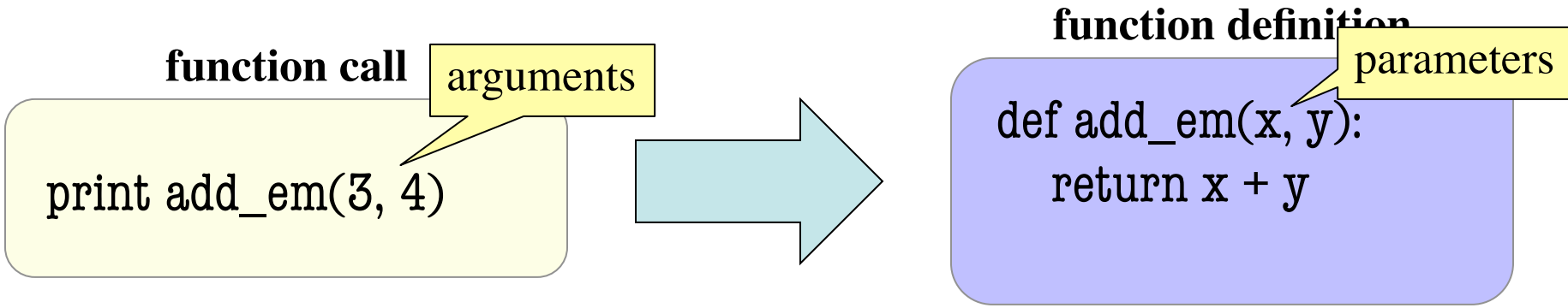
Additional space in labs: We have added some space and staffing to the 12:20 and 1:25 labs on Tuesday. There is still space to move into these labs.

Assignment 1 is out! Grab a printed copy today, and refer to the same text on the website. This assignment is due **February 18**, and you will re-submit until it's all correct.

Printed copies are coming in a few minutes!

Slides by D. Gries, L. Lee, S. Marschner, W. White

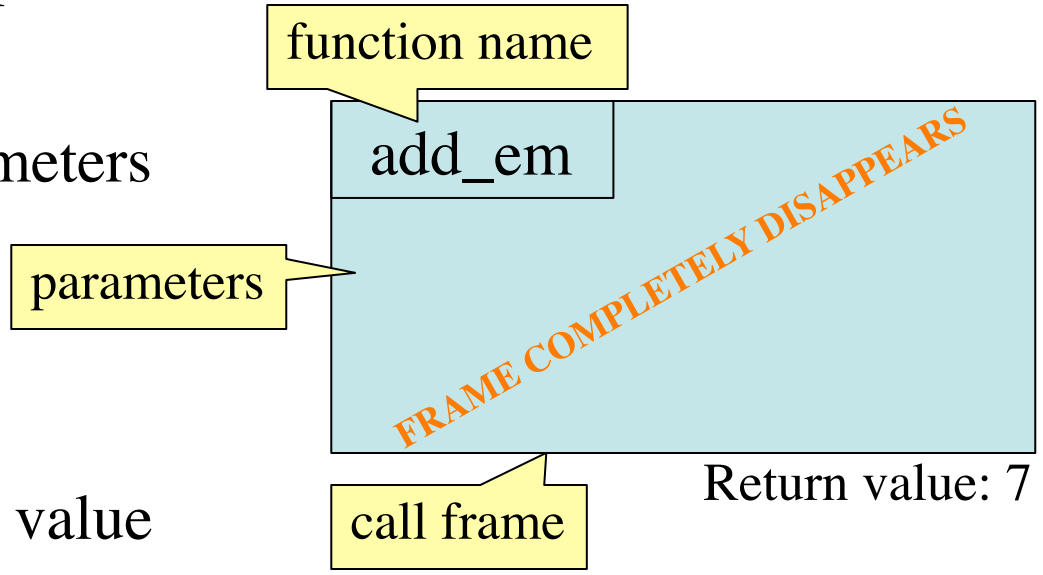
How Do Functions Work?



To evaluate a function call expression:

1. Create a frame for the call
2. Assign arguments to parameters
3. Execute function body
4. Erase the frame

The value of the function call expression is the returned value



How Do Functions Work?

function call

```
print add_em(3, 4)
```

arguments

function definition

```
def add_em(x, y):
1   sum = x + y
2   return sum
```

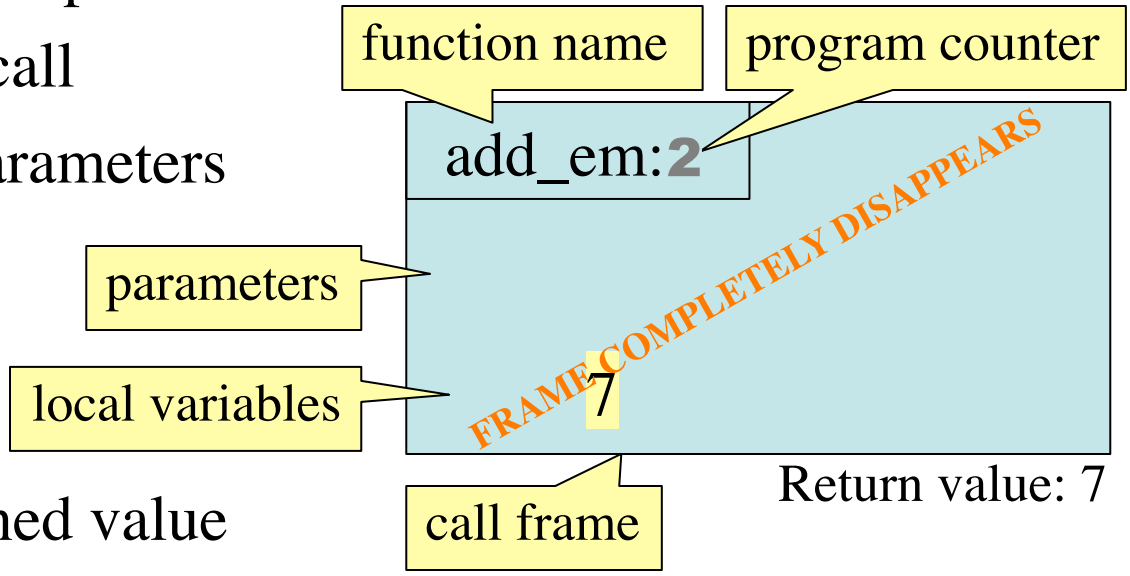
parameters

statement numbers

To evaluate a function call expression:

1. Create a frame for the call
2. Assign arguments to parameters
3. Execute function body
4. Erase the frame

The value of the function call expression is the returned value



function call

```
a = 1
b = 2
swap(a, b)
```

function definition

```
def swap(x, y):
1   x = y
2   y = x
3   print 'x, y:', x, y
```

with print, use commas to print several values

To evaluate a function call expression:

1. Create a frame for the call
2. Assign arguments to parameters
3. Execute function body
4. Erase the frame

The value of the function call expression is the returned value

Execute this call on paper. What gets printed out?

A: x, y: 1 2
B: x, y: 2 1
C: x, y: 2 2
D: x, y: 1 1
E: I don't know

function call

```
a = 1
b = 2
swap(a, b)
```

arguments

module (global) variables

a 1

b 2

function definition

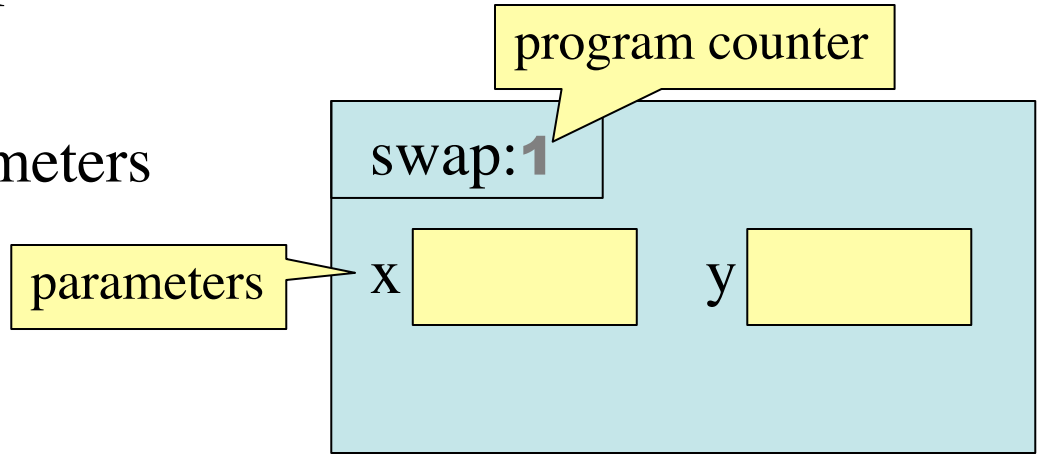
parameters

```
def swap(x, y):
1   x = y
2   y = x
3   print 'x, y:', x, y
```

To evaluate a function call expression:

1. Create a frame for the call
2. Assign arguments to parameters
3. Execute function body
4. Erase the frame

The value of the function call expression is the returned value



function call

```
a = 1
b = 2
swap(a, b)
print 'a, b:', a, b
```

arguments

module (global) variables

a 1

b 2

function definition

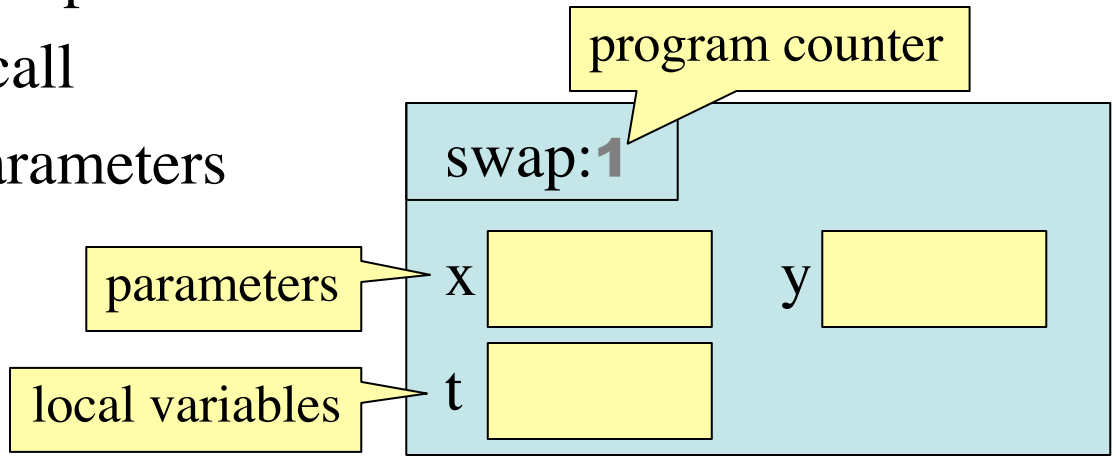
parameters

```
def swap(x, y):
1   t = x
2   x = y
3   y = t
4   print 'x, y:', x, y
```

To evaluate a function call expression:

1. Create a frame for the call
2. Assign arguments to parameters
3. Execute function body
4. Erase the frame

The value of the function call expression is the returned value

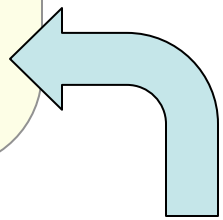


function call

```
import point
p = point.Point(1,2,3)
q = point.Point(3,4,5)
swap_x(p, q)
print 'p:', p
print 'q:', q
```

function definition

```
def swap_x(p, q):
1   t = p.x
2   p.x = q.x
3   q.x = t
```



Execute this code on paper.
You will draw 2 objects and a frame.
What is in p.x at the end?

A: 1
B: 2
C: 3
D: I don't know

function call

```
import point
p = point.Point(1,2,3)
q = point.Point(3,4,5)
swap_x(p, q)
print 'p:', p
print 'q:', q
```

module (global) variables

p **id1**

q **id2**

function definition

```
def swap_x(p, q):
1   t = p.x
2   p.x = q.x
3   q.x = t
```

id1

Point

x 1.0

y 2.0

z 3.0

id2

Point

x 3.0

y 4.0

z 5.0

swap_x:1

p

q

t

(9:05 version)

function call

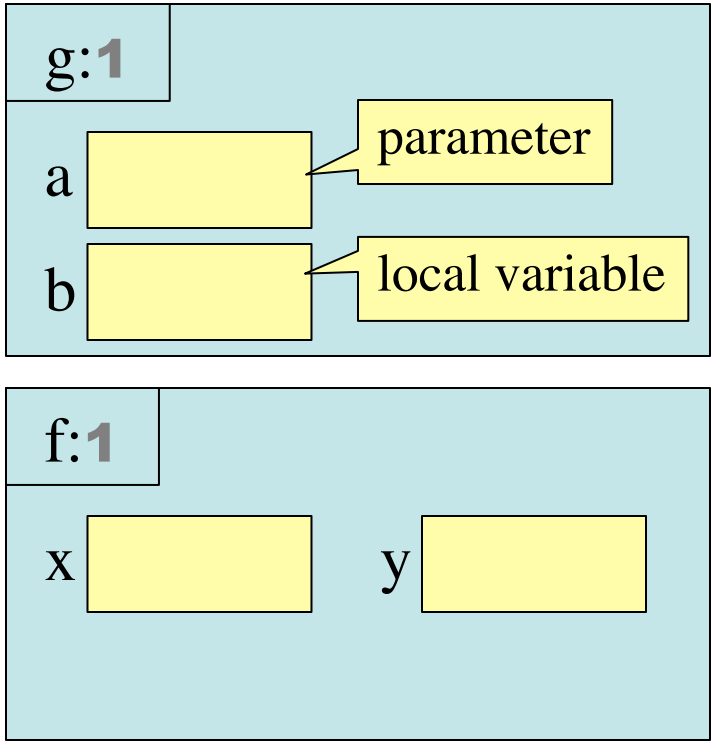
```
c = 2
print g(3)
```

c 2

function definitions

```
def f(x, y):
1   return 3*x + y
```

```
def g(a):
1   b = f(a, c)
2   return f(b, a)
```



1. Create a frame for the call
2. Assign arguments to parameters
3. Execute function body
4. Erase the frame

(11:15 version)

function call

```
lt_speed = 3e8
print g(3)
```

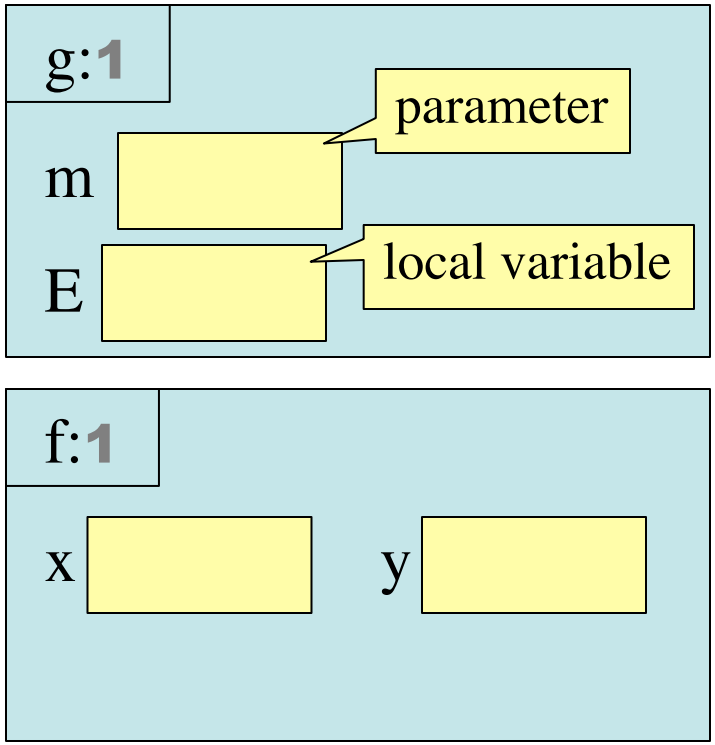
function definitions

```
def g(m):
    E = f(m, lt_speed)
    return E
```

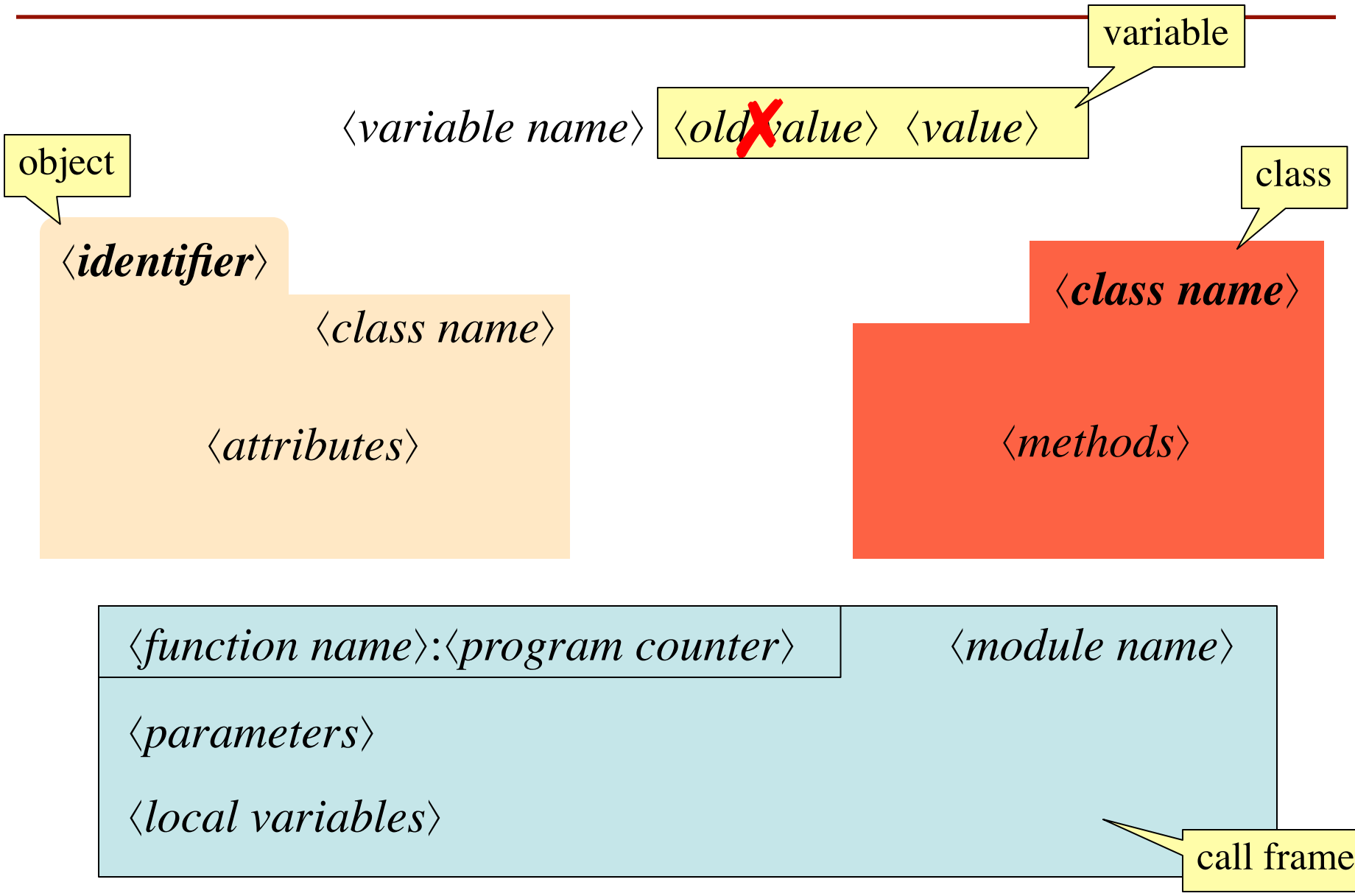
```
def f(x, y):
    return x * (y**2)
```

lt_speed 3×10^8

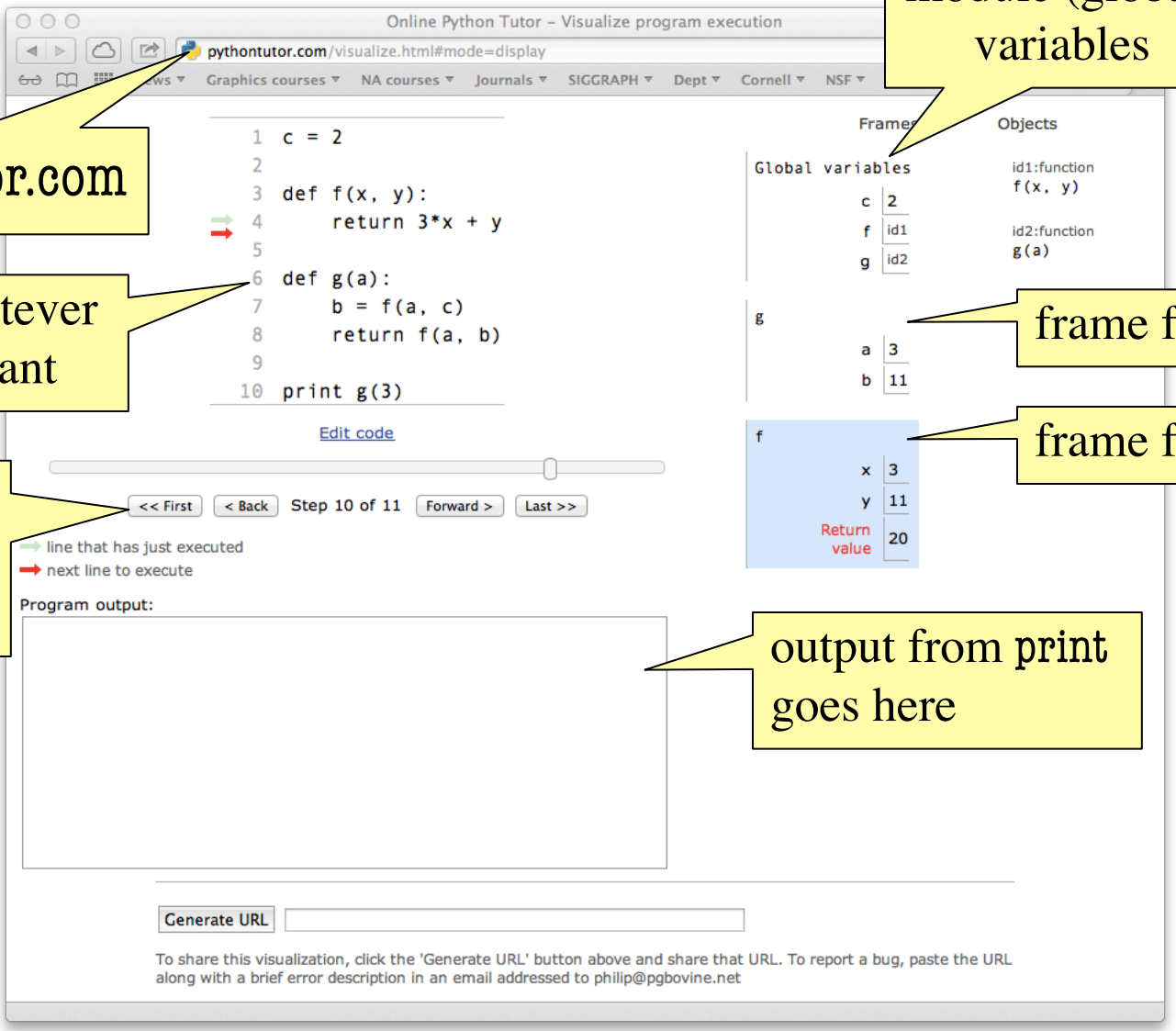
1. Create a frame for the call
2. Assign arguments to parameters
3. Execute function body
4. Erase the frame



How to Draw Things



Online Python Tutor



pythontutor.com

module (global) variables

type in whatever code you want

frame for call of g

frame for call of f

controls for stepping through code

output from print goes here