Call Frames
Final Review
Spring 2018

CS 1110

The Big Issue

- Cannot answer questions on this topic without
  - drawing variables
  - drawing frames for function calls
  - drawing objects when they are created
- Learning to do this is useful in general
  - Helps you “think like a computer”
  - Easier to find errors in your programs.

What Do You Need to Know?

- Major topics
  - ***local variables (in a function body)***
  - ***function call (call frames, call stack)***
  - ***class folders, inheritance, and constructor calls***
- Examples from previous exams
  - Question 3 on prelim 1
  - Question 6 on prelim 2

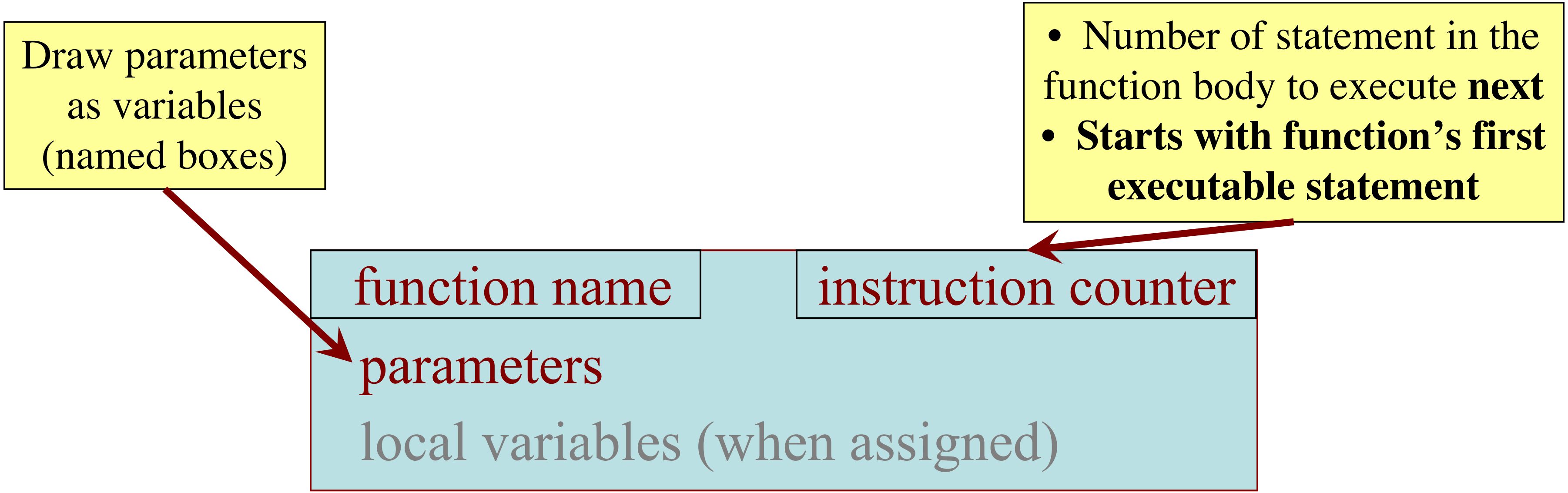
Important

- Code execution is an important part of the final
- You need to know how to
  - draw variables
  - draw call frames
  - draw objects

The purpose of such questions on executing statements with constructs and function calls is to test your understanding of how Python programs are executed.

The Frame (box) for a Function Call

- **Function Frame**: Representation of function call
- A **conceptual model** of Python



Draw parameters as variables (named boxes)

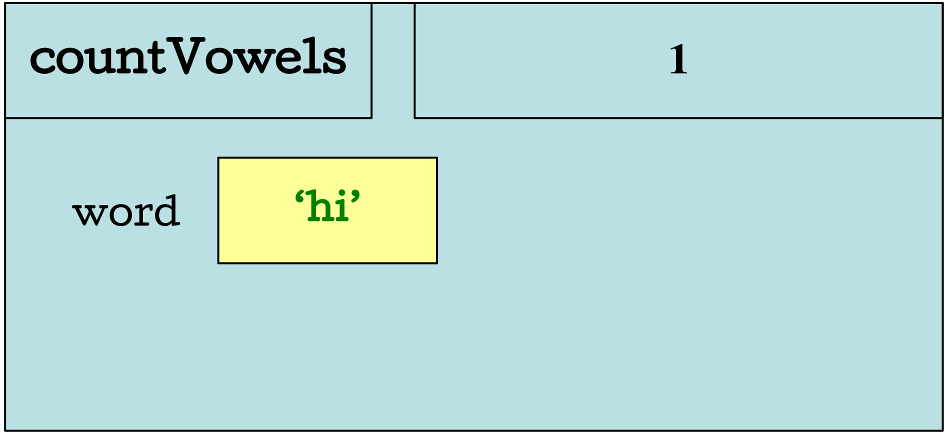
- Number of statement in the function body to execute **next**
- **Starts with function's first executable statement**

Iterative Example

```
def countVowels(word):
    """Returns: The number of vowels
    in the string s.
    Precondition: s is a string"""
1   x = 0
2   count = 0
3   while x < len(word):
4       if word[x] in [‘a’,’e’,’i’,’o’,’u’]:
5           count += 1
6       x += 1
7   return count
```

Call: e = count_vowels(‘hi’)

Call Stack:



countVowels | 1

word ‘hi’

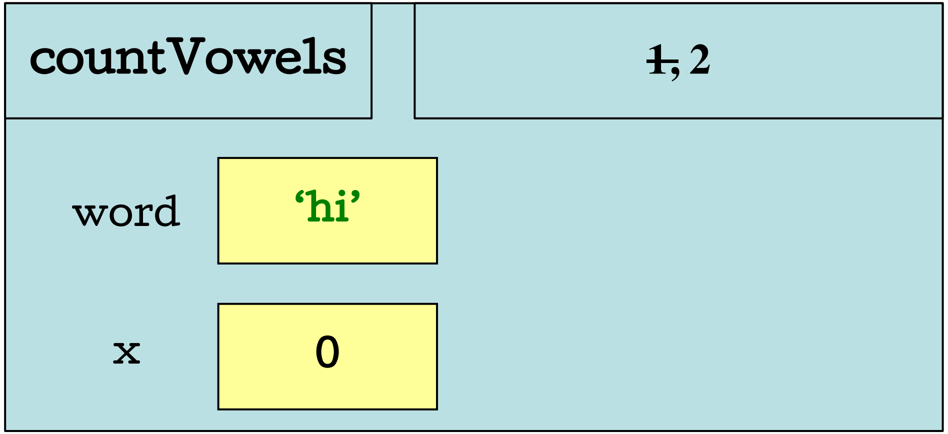
Global Space:

Iterative Example

```
  def countVowels(word):
        """Returns: The number of vowels
        in the string s.
        Precondition: s is a string"""
1       x = 0
➡ 2     count = 0
3       while x < len(word):
4           if word[x] in ['a','e','i','o','u']:
5               count += 1
6           x += 1
7       return count
```

Call: e = count_vowels('hi')

Call Stack:



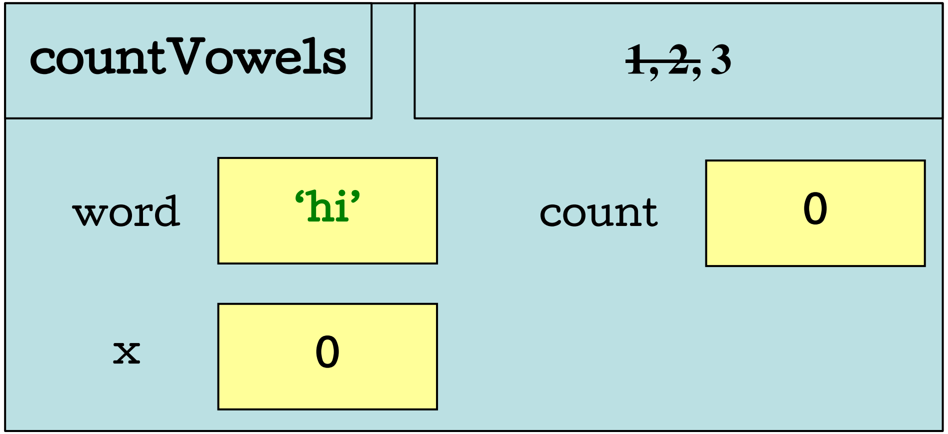
Global Space:

Iterative Example

```
  def countVowels(word):
      """Returns: The number of vowels
      in the string s.
      Precondition: s is a string"""
  1   x = 0
  2   count = 0
➡ 3   while x < len(word):
  4       if word[x] in [‘a’,’e’,’i’,’o’,’u’]:
  5           count += 1
  6       x += 1
  7   return count
```

Call: e = count_vowels(‘hi’)

Call Stack:



countVowels ~~1, 2,~~ 3

word ‘hi’

count 0

x 0

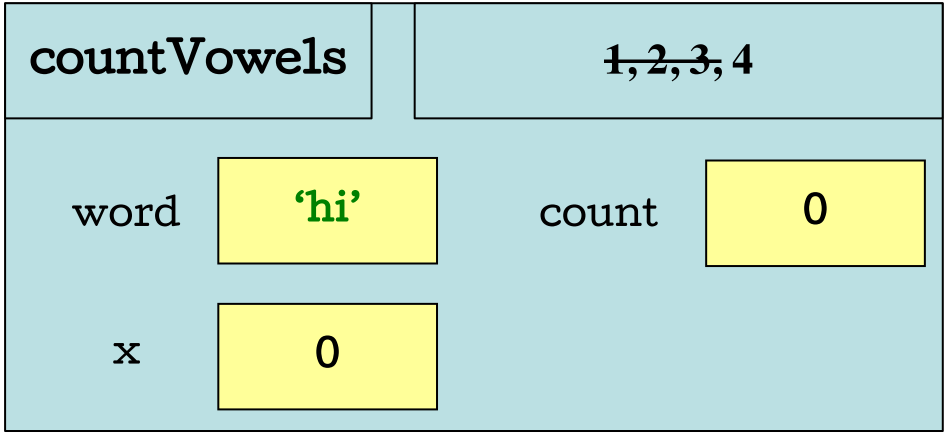
Global Space:

Iterative Example

Call Stack:

```
  def countVowels(word):
      """Returns: The number of vowels
      in the string s.
      Precondition: s is a string"""
1     x = 0
2     count = 0
3     while x < len(word):
4         if word[x] in ['a','e','i','o','u']:
5             count += 1
6         x += 1
7     return count
```

Call: e = count_vowels('hi')



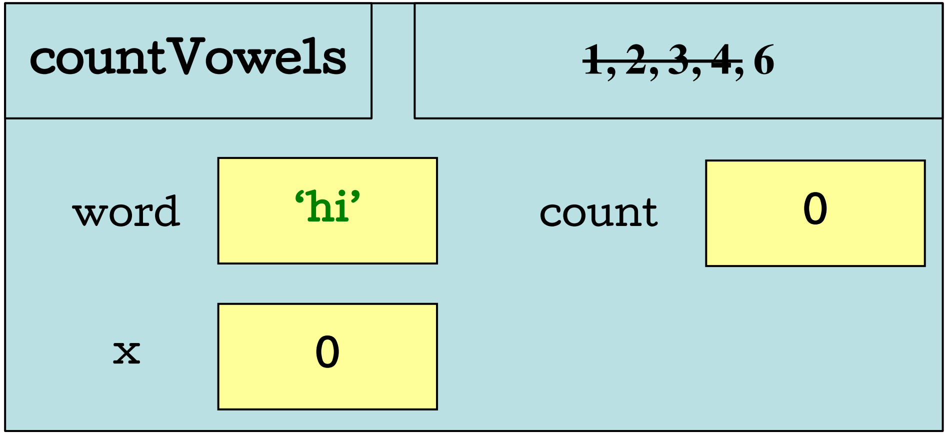
Global Space:

Iterative Example

```
   def countVowels(word):
       """Returns: The number of vowels
       in the string s.
       Precondition: s is a string"""
 1     x = 0
 2     count = 0
 3     while x < len(word):
 4         if word[x] in ['a','e','i','o','u']:
 5             count += 1
→6         x += 1
 7     return count
```

Call: e = count_vowels('hi')

Call Stack:



countVowels

~~1, 2, 3, 4,~~ 6

word 'hi'

count 0

x 0

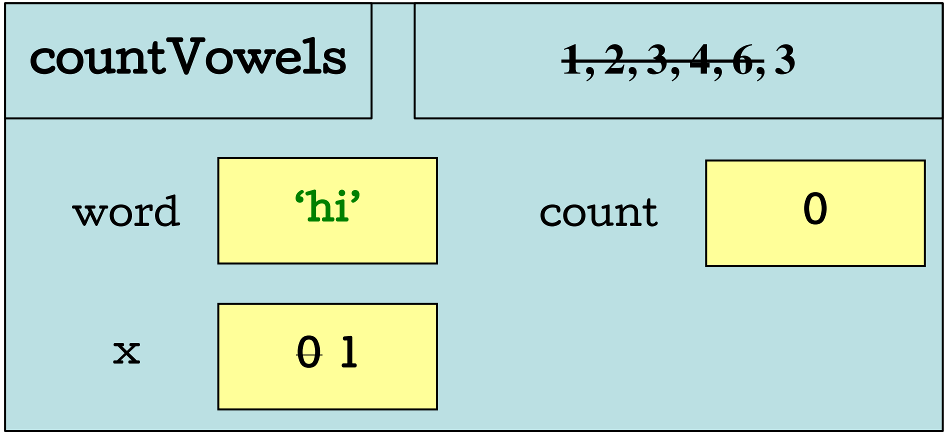
Global Space:

Iterative Example

```
  def countVowels(word):
      """Returns: The number of vowels
      in the string s.
      Precondition: s is a string"""
1     x = 0
2     count = 0
➡3    while x < len(word):
4         if word[x] in ['a','e','i','o','u']:
5             count += 1
6         x += 1
7     return count
```

Call: e = count_vowels('hi')

Call Stack:



countVowels

~~1, 2, 3, 4, 6,~~ 3

word 'hi'

count 0

x ~~0~~ 1

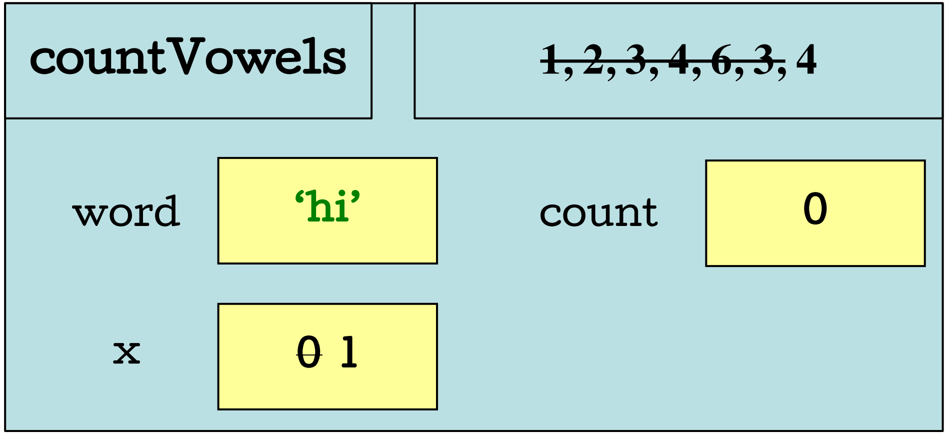
Global Space:

Iterative Example

```
  def countVowels(word):
      """Returns: The number of vowels
      in the string s.
      Precondition: s is a string"""
  1   x = 0
  2   count = 0
  3   while x < len(word):
➡ 4       if word[x] in ['a','e','i','o','u']:
  5           count += 1
  6       x += 1
  7   return count
```

Call: e = count_vowels('hi')

Call Stack:



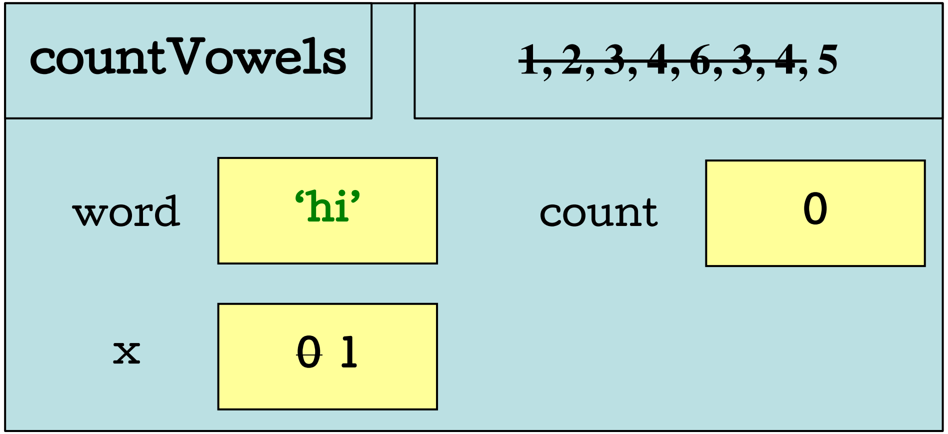
Global Space:

Iterative Example

```
  def countVowels(word):
      """Returns: The number of vowels
      in the string s.
      Precondition: s is a string"""
1     x = 0
2     count = 0
3     while x < len(word):
4         if word[x] in ['a','e','i','o','u']:
5             count += 1
6         x += 1
7     return count
```

Call: e = count_vowels('hi')

Call Stack:



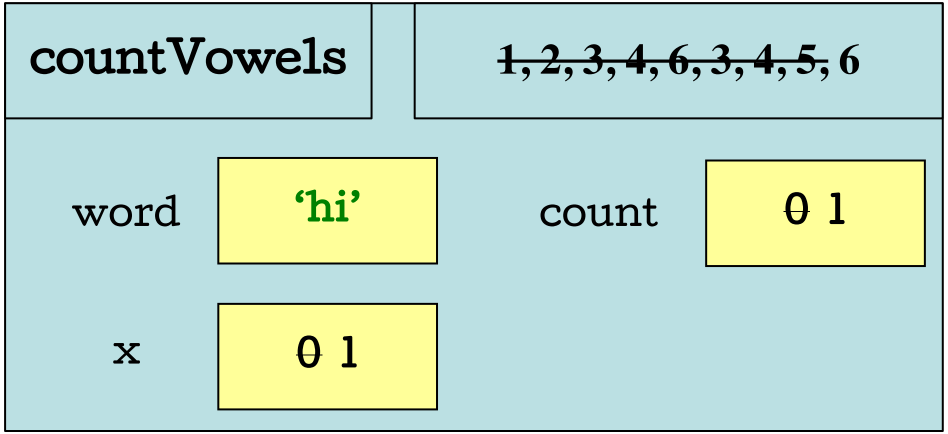
Global Space:

Iterative Example

```
def countVowels(word):
      """Returns: The number of vowels
      in the string s.
      Precondition: s is a string"""
1     x = 0
2     count = 0
3     while x < len(word):
4         if word[x] in ['a','e','i','o','u']:
5             count += 1
6         x += 1
7     return count
```

Call: e = count_vowels('hi')

Call Stack:



countVowels

~~1, 2, 3, 4, 6, 3, 4, 5,~~ 6

word 'hi'

count ~~0~~ 1

x ~~0~~ 1

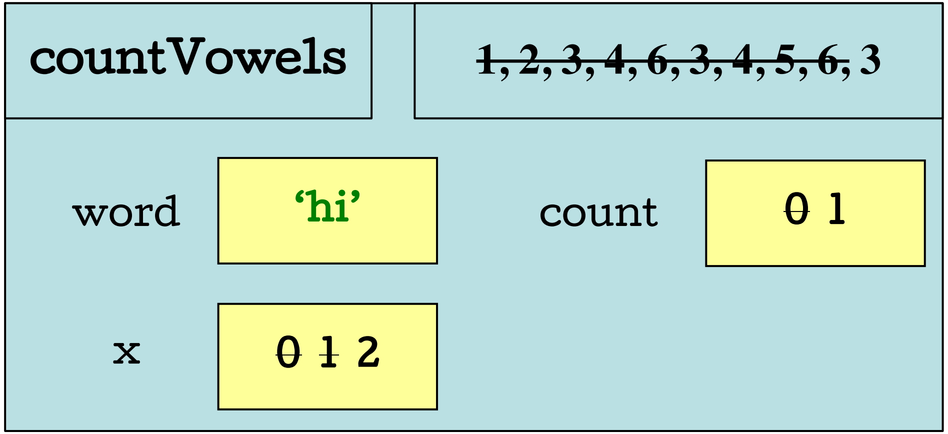
Global Space:

Iterative Example

```
  def countVowels(word):
        """Returns: The number of vowels
        in the string s.
        Precondition: s is a string"""
  1     x = 0
  2     count = 0
➡ 3     while x < len(word):
  4         if word[x] in [‘a’,’e’,’i’,’o’,’u’]:
  5             count += 1
  6         x += 1
  7     return count
```

Call: e = count_vowels(‘hi’)

Call Stack:



countVowels

~~1, 2, 3, 4, 6, 3, 4, 5, 6,~~ 3

word ‘hi’

count ~~0~~ 1

x ~~0~~ ~~1~~ 2

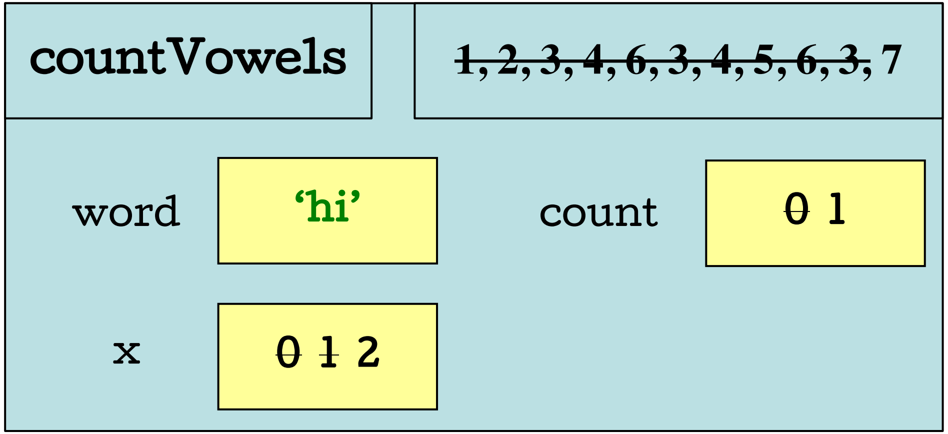
Global Space:

Iterative Example

```
  def countVowels(word):
      """Returns: The number of vowels
      in the string s.
      Precondition: s is a string"""
1     x = 0
2     count = 0
3     while x < len(word):
4         if word[x] in ['a','e','i','o','u']:
5             count += 1
6         x += 1
➡ 7   return count
```

Call: e = count_vowels('hi')

Call Stack:



countVowels | ~~1, 2, 3, 4, 6, 3, 4, 5, 6, 3,~~ 7

word: 'hi'

count: ~~0~~ 1

x: ~~0~~ ~~1~~ 2

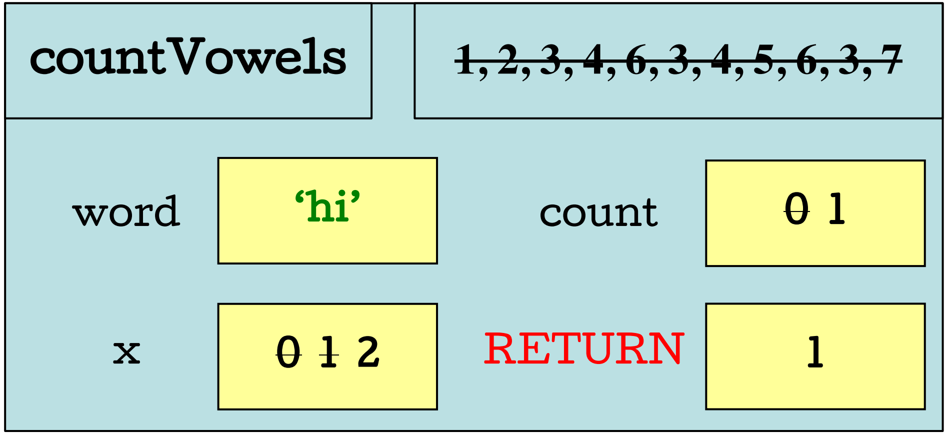
Global Space:

Iterative Example

```
  def countVowels(word):
      """Returns: The number of vowels
      in the string s.
      Precondition: s is a string"""
1     x = 0
2     count = 0
3     while x < len(word):
4         if word[x] in [‘a’,’e’,’i’,’o’,’u’]:
5             count += 1
6         x += 1
7     return count
```

Call: e = count_vowels(‘hi’)

Call Stack:



countVowels ~~1, 2, 3, 4, 6, 3, 4, 5, 6, 3, 7~~

word ‘hi’

count ~~0~~ 1

x ~~0~~ ~~1~~ 2

RETURN 1

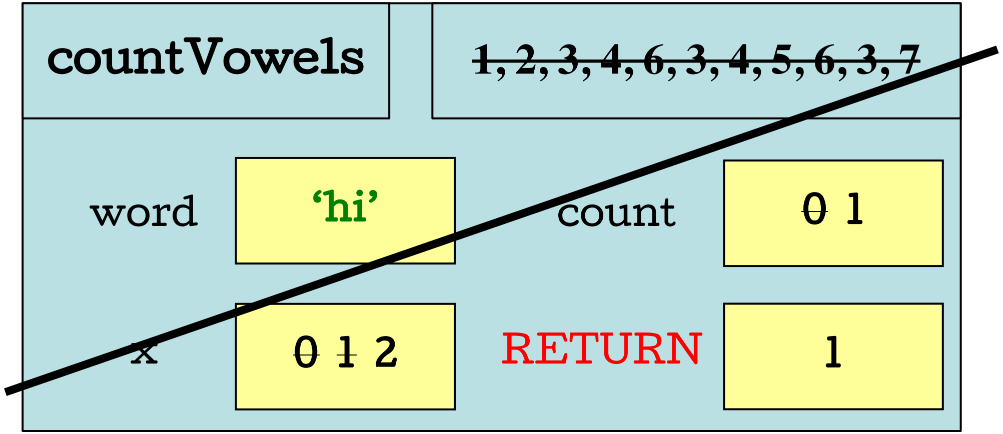
Global Space:

Iterative Example

```
  def countVowels(word):
       """Returns: The number of vowels
       in the string s.
       Precondition: s is a string"""
1      x = 0
2      count = 0
3      while x < len(word):
4          if word[x] in [‘a’,’e’,’i’,’o’,’u’]:
5              count += 1
6          x += 1
7      return count
```

Call: e = count_vowels(‘hi’)

Call Stack:



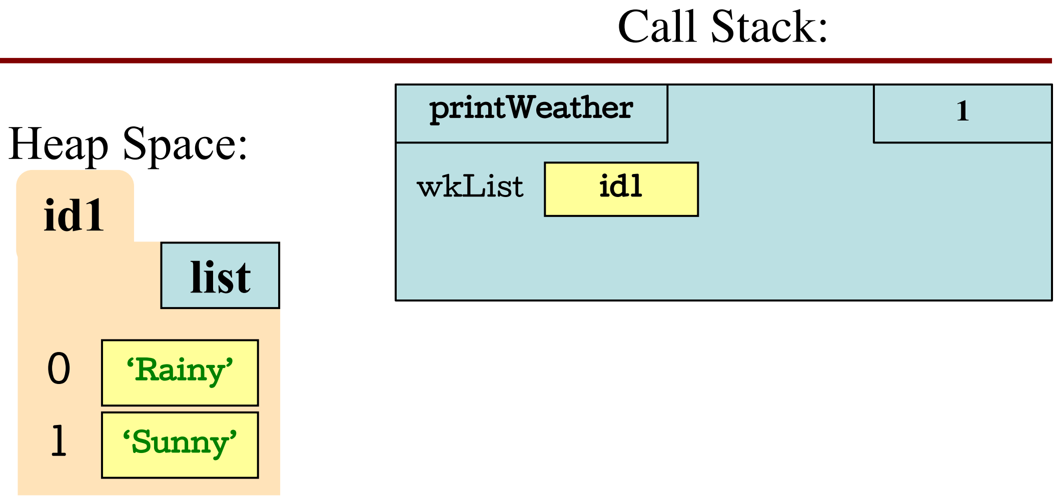
Global Space: e 1

Subcall Example

Global Space: a idl

```
def printWeather(wkList):
1       for item in wkList:
2           dayWeather(item)
def dayWeather(day):
3       if day == ‘Sunny’:
4           print (‘Time for a picnic!’)
5       if day == ‘Rainy’:
6           print (‘Grab your umbrella!’)
```

Call: printWeather(a)


Heap Space:

idl

list

0 ‘Rainy’

1 ‘Sunny’

Call Stack:

printWeather 1

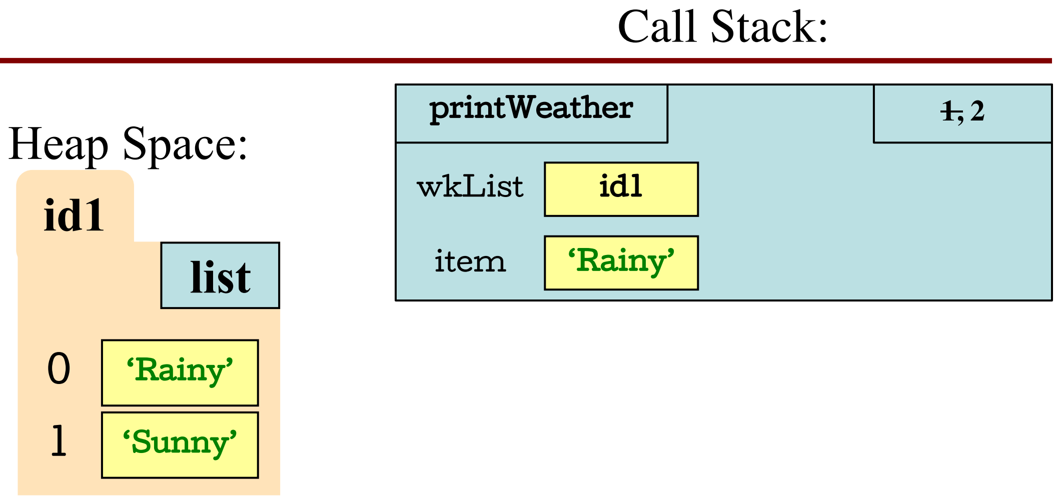
wkList idl

Subcall Example

Global Space: a id1

```
  def printWeather(wkList):
1     for item in wkList:
2         dayWeather(item)
  def dayWeather(day):
3     if day == ‘Sunny’:
4         print (‘Time for a picnic!’)
5     if day == ‘Rainy’:
6         print (‘Grab your umbrella!’)
```

Call: printWeather(a)



Heap Space:

id1 list

0 ‘Rainy’

1 ‘Sunny’

Call Stack:

printWeather ~~1~~, 2

wkList id1

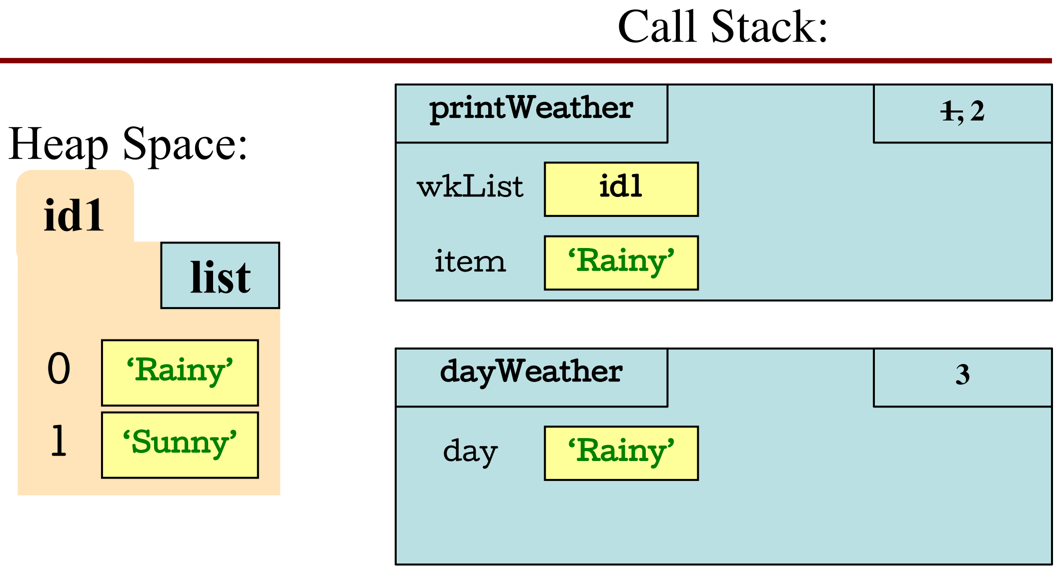
item ‘Rainy’

Subcall Example

Global Space: a id1

```
  def printWeather(wkList):
1     for item in wkList:
2         dayWeather(item)
  def dayWeather(day):
3     if day == 'Sunny':
4         print ('Time for a picnic!')
5     if day == 'Rainy':
6         print ('Grab your umbrella!')
```

Call: printWeather(a)

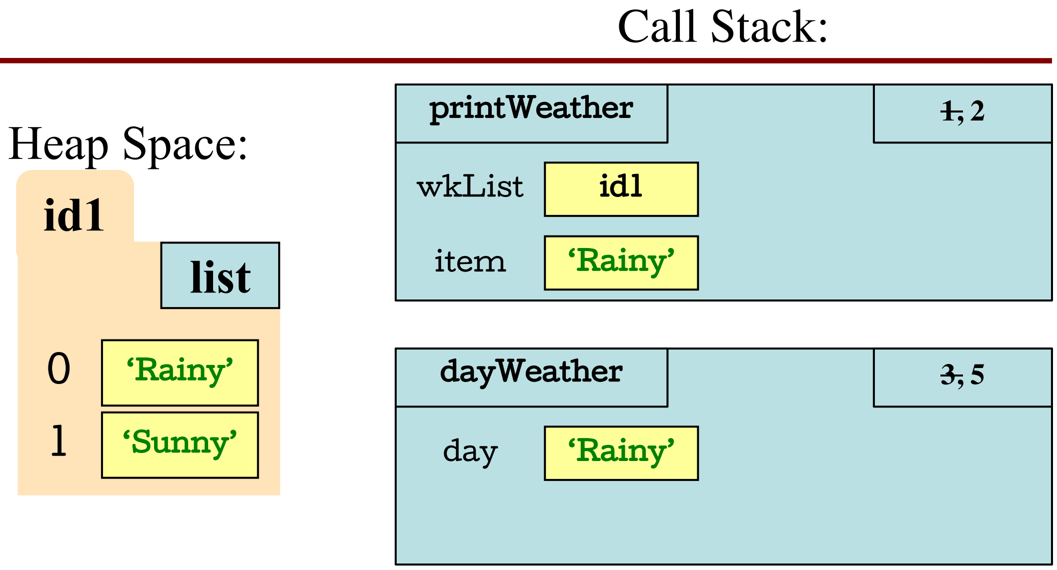


Subcall Example

Global Space: a id1

```
   def printWeather(wkList):
1      for item in wkList:
2          dayWeather(item)
   def dayWeather(day):
3      if day == 'Sunny':
4          print ('Time for a picnic!')
5      if day == 'Rainy':
6          print ('Grab your umbrella!')
```

Call: printWeather(a)



Heap Space:

id1 list

0 'Rainy'

1 'Sunny'

Call Stack:

printWeather ~~1~~, 2

wkList id1

item 'Rainy'

dayWeather ~~3~~, 5

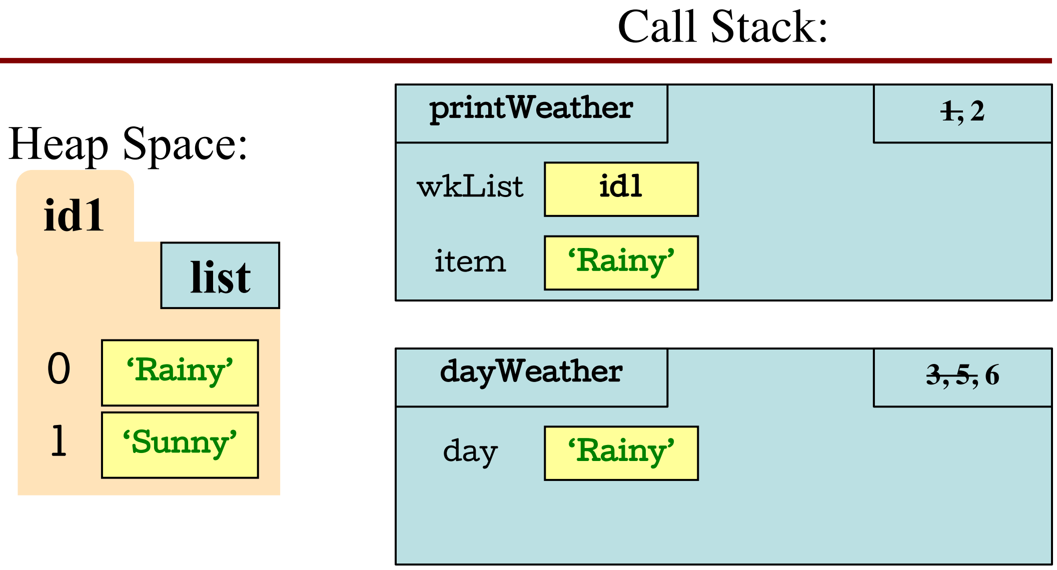
day 'Rainy'

Subcall Example

Global Space: a idl

```
    def printWeather(wkList):
1       for item in wkList:
2           dayWeather(item)
    def dayWeather(day):
3       if day == 'Sunny':
4           print ('Time for a picnic!')
5       if day == 'Rainy':
6           print ('Grab your umbrella!')
```

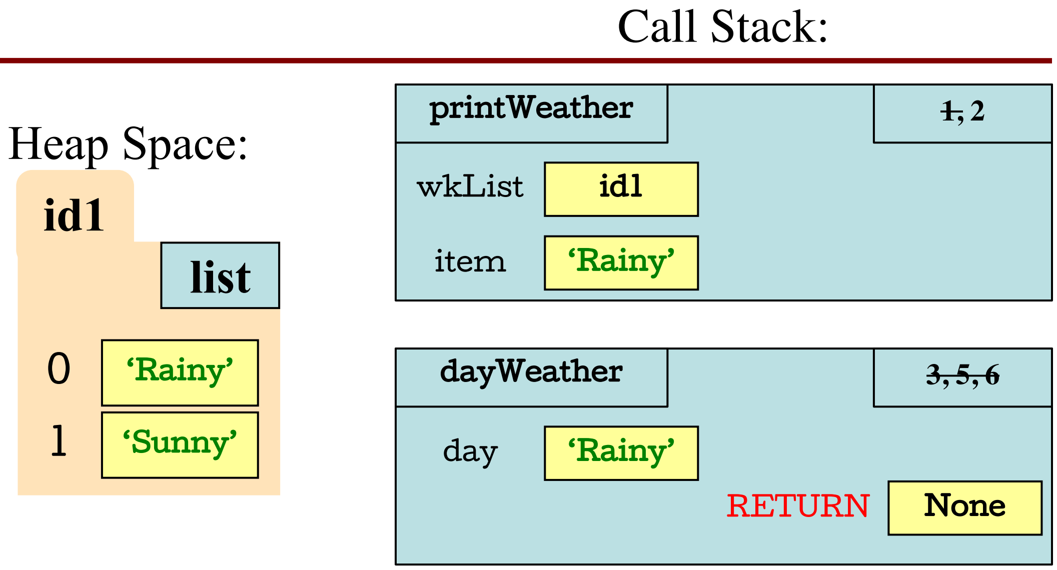
Call: printWeather(a)



Subcall Example

Global Space: a id1

```
  def printWeather(wkList):
1     for item in wkList:
2         dayWeather(item)
  def dayWeather(day):
3     if day == ‘Sunny’:
4         print (‘Time for a picnic!’)
5     if day == ‘Rainy’:
6         print (‘Grab your umbrella!’)
```



Heap Space:

id1 list

0	‘Rainy’
1	‘Sunny’

Call Stack:

printWeather ~~1~~, 2

wkList id1

item ‘Rainy’

dayWeather ~~3, 5, 6~~

day ‘Rainy’

RETURN None

Call: printWeather(a)

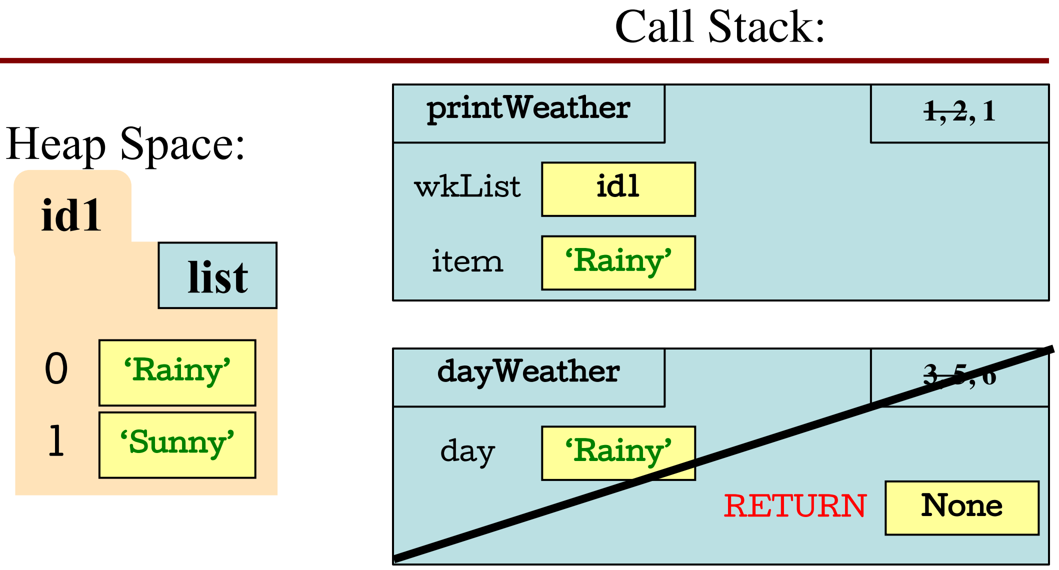
Output:
Grab your umbrella!

Subcall Example

Global Space: a id1

```
  def printWeather(wkList):
1     for item in wkList:
2         dayWeather(item)
  def dayWeather(day):
3     if day == ‘Sunny’:
4         print (‘Time for a picnic!’)
5     if day == ‘Rainy’:
6         print (‘Grab your umbrella!’)
```

Call: printWeather(a)



Output:
Grab your umbrella!

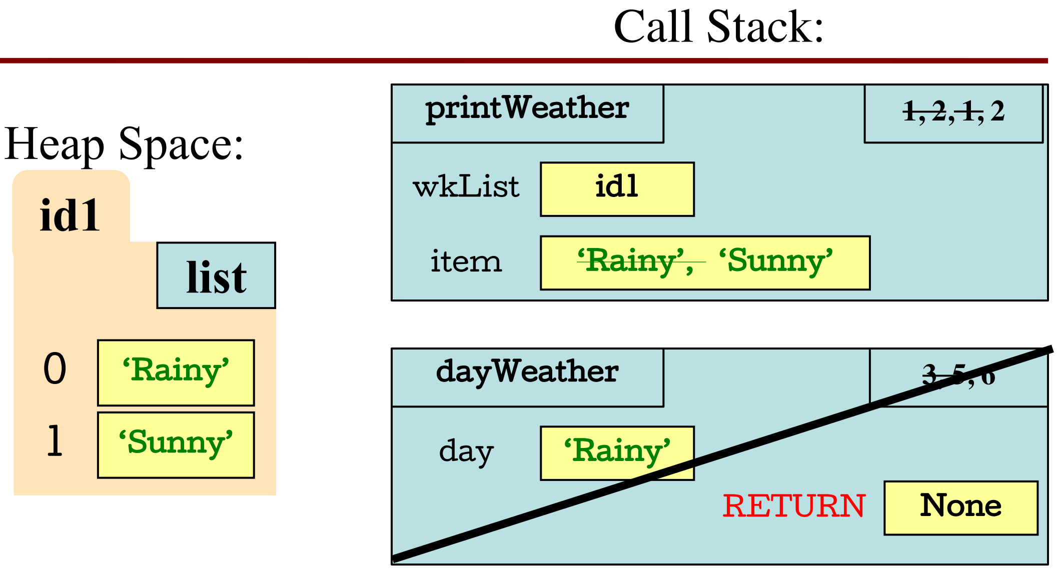
Subcall Example

Global Space: a id1

```
def printWeather(wkList):
1       for item in wkList:
2           dayWeather(item)
def dayWeather(day):
3       if day == 'Sunny':
4           print ('Time for a picnic!')
5       if day == 'Rainy':
6           print ('Grab your umbrella!')
```

Call: printWeather(a)

Output:
Grab your umbrella!



Subcall Example

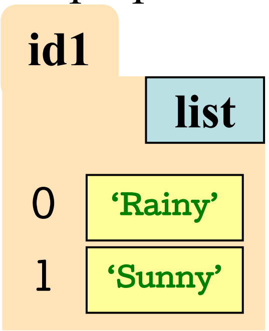
Global Space: a id1

```
def printWeather(wkList):
1       for item in wkList:
2           dayWeather(item)
def dayWeather(day):
3       if day == ‘Sunny’:
4           print (‘Time for a picnic!’)
5       if day == ‘Rainy’:
6           print (‘Grab your umbrella!’)
```

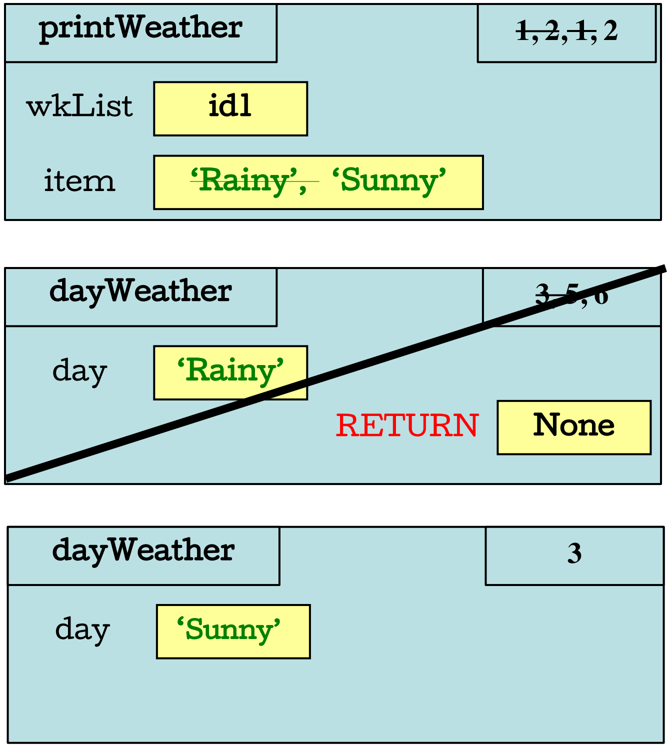
Call: printWeather(a)

Output:
Grab your umbrella!

Heap Space:



Call Stack:



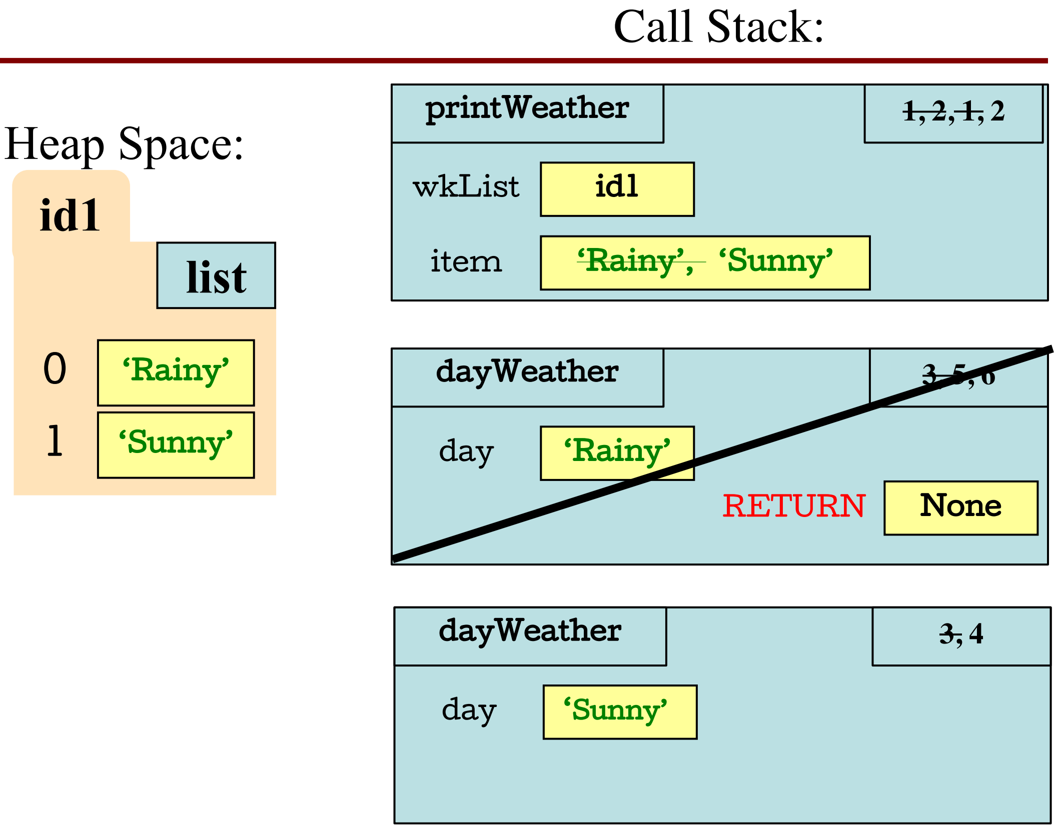
Subcall Example

Global Space: a idl

```
  def printWeather(wkList):
1     for item in wkList:
2         dayWeather(item)
  def dayWeather(day):
3     if day == 'Sunny':
4         print ('Time for a picnic!')
5     if day == 'Rainy':
6         print ('Grab your umbrella!')
```

Call: printWeather(a)

Output:
Grab your umbrella!



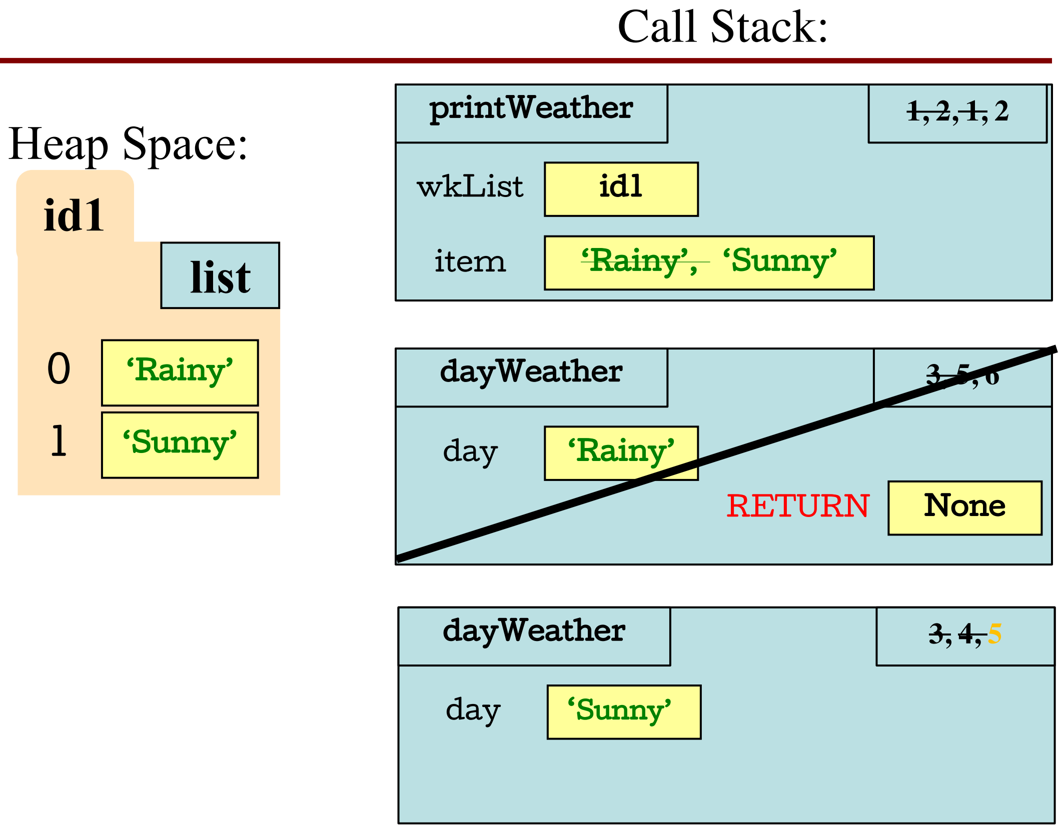
Subcall Example

Global Space: a id1

```
    def printWeather(wkList):
1       for item in wkList:
2           dayWeather(item)
    def dayWeather(day):
3       if day == 'Sunny':
4           print ('Time for a picnic!')
5       if day == 'Rainy':
6           print ('Grab your umbrella!')
```

Call: printWeather(a)

Output:
Grab your umbrella!



Heap Space:

id1 list

0 'Rainy'

1 'Sunny'

Call Stack:

printWeather ~~1, 2, 1,~~ 2

wkList id1

item ~~'Rainy',~~ 'Sunny'

dayWeather ~~3, 5,~~ 6

day 'Rainy'

RETURN None

dayWeather ~~3, 4,~~ 5

day 'Sunny'

Subcall Example

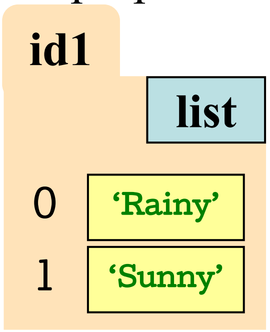
Global Space: a idl

```
   def printWeather(wkList):
1      for item in wkList:
2          dayWeather(item)
   def dayWeather(day):
3      if day == 'Sunny':
4          print ('Time for a picnic!')
5      if day == 'Rainy':
6          print ('Grab your umbrella!')
```

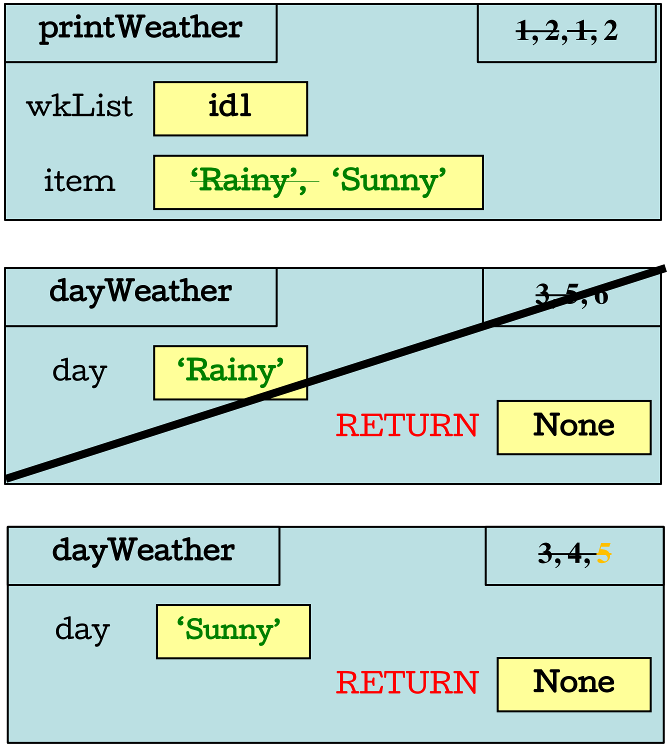
Call: printWeather(a)

Output:
Grab your umbrella!
Time for a picnic!

Heap Space:



Call Stack:

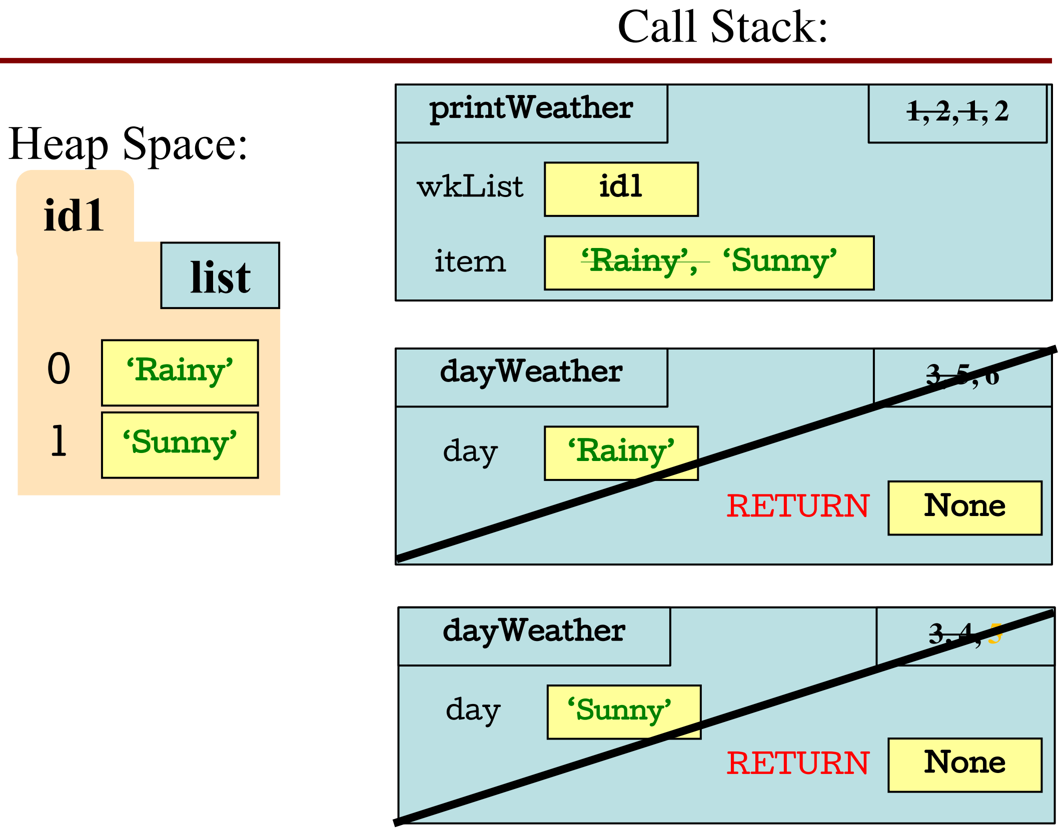


Subcall Example

Global Space: a idl

```
  def printWeather(wkList):
1     for item in wkList:
2         dayWeather(item)
  def dayWeather(day):
3     if day == ‘Sunny’:
4         print (‘Time for a picnic!’)
5     if day == ‘Rainy’:
6         print (‘Grab your umbrella!’)
```

Call: printWeather(a)



Output:
Grab your umbrella!
Time for a picnic!

Subcall Example

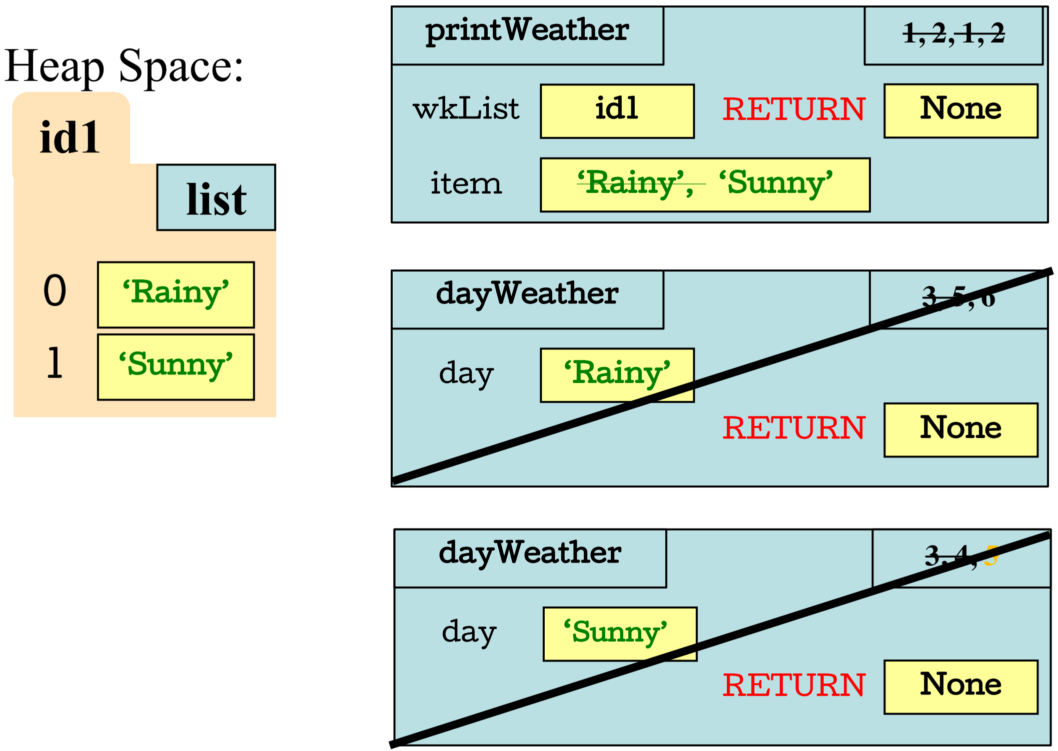
Global Space: a idl

```
def printWeather(wkList):
1       for item in wkList:
2           dayWeather(item)
def dayWeather(day):
3       if day == 'Sunny':
4           print ('Time for a picnic!')
5       if day == 'Rainy':
6           print ('Grab your umbrella!')
```

Call: printWeather(a)

Output:
Grab your umbrella!
Time for a picnic!

Call Stack:



Subcall Example

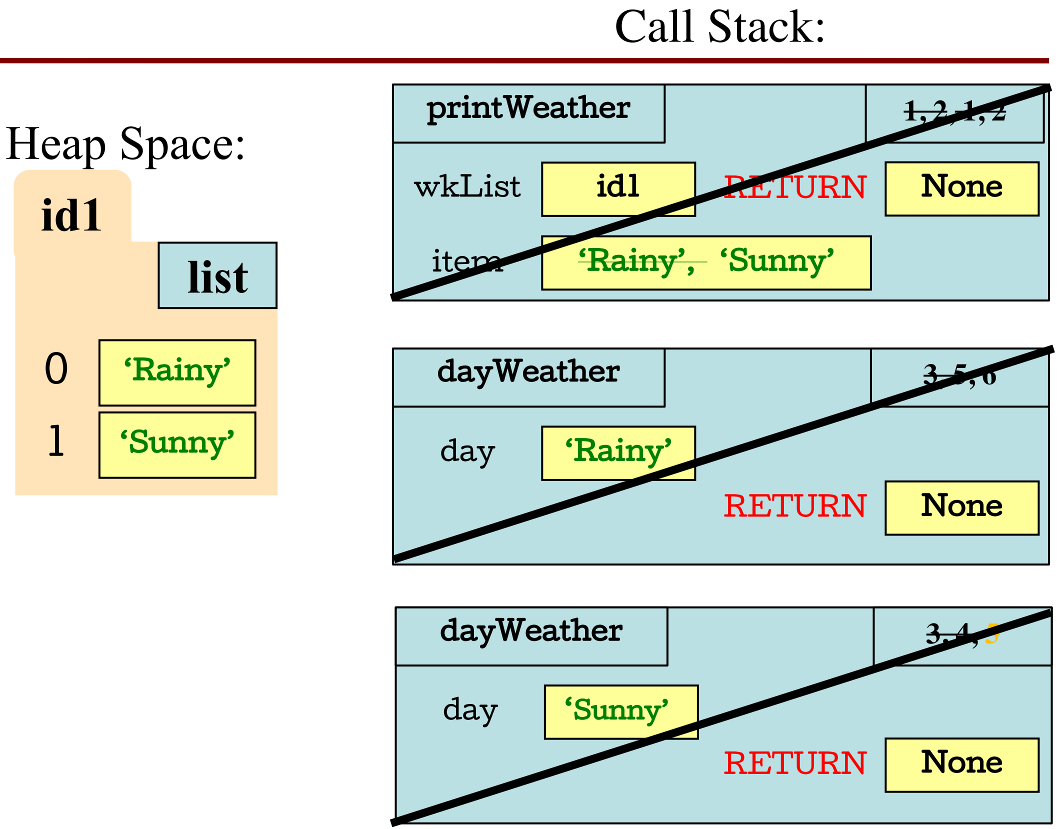
Global Space: a idl

```
  def printWeather(wkList):
1     for item in wkList:
2         dayWeather(item)
  def dayWeather(day):
3     if day == 'Sunny':
4         print ('Time for a picnic!')
5     if day == 'Rainy':
6         print ('Grab your umbrella!')
```

Call: printWeather(a)

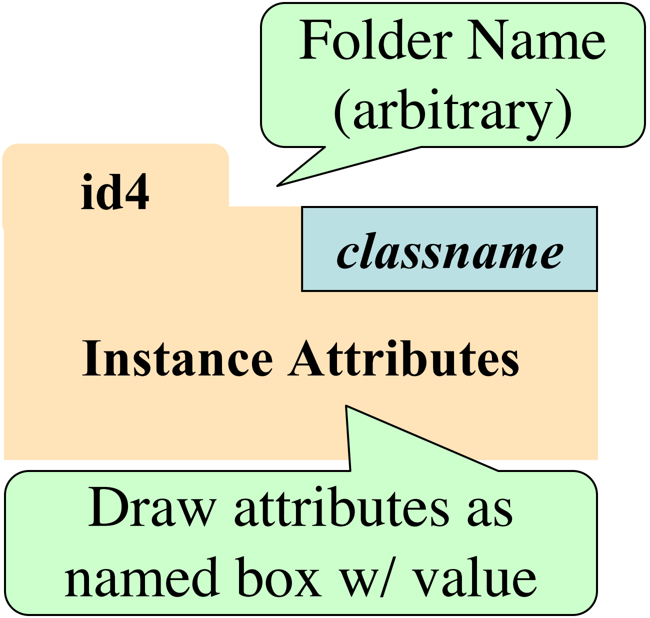
Output:

```
Grab your umbrella!
Time for a picnic!
```



Diagramming Objects (Folders)

Object Folder



Folder Name (arbitrary)

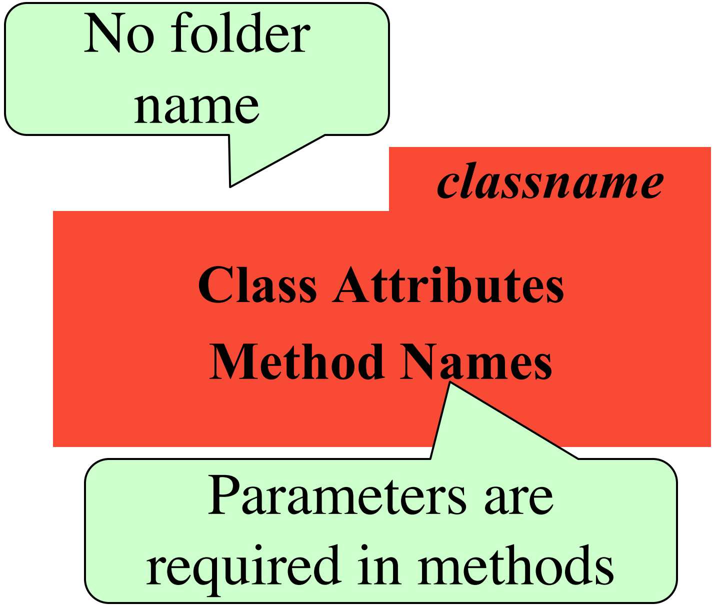
id4

classname

Instance Attributes

Draw attributes as named box w/ value

Class Folder



No folder name

classname

Class Attributes

Method Names

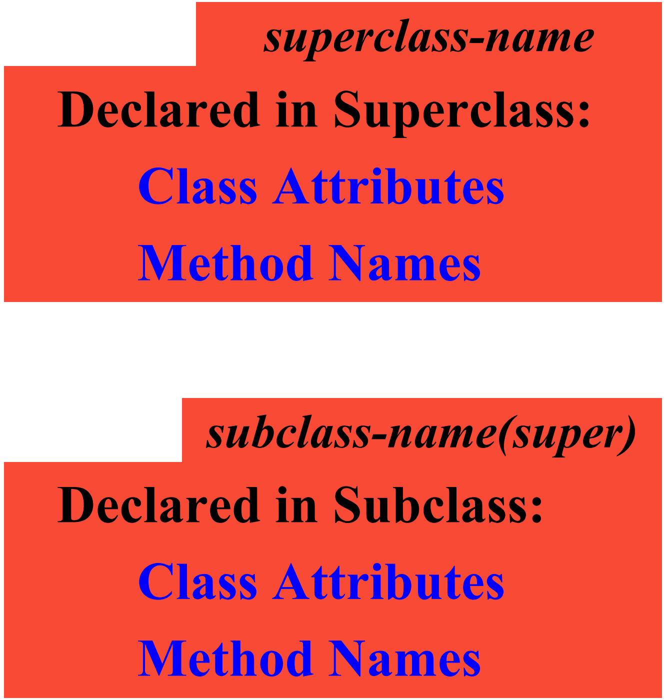
Parameters are required in methods

Evaluation of a Constructor Call

3 steps to evaluating the call C(args)

- ***Create a new folder*** (object) of class C
  - Give it with a unique name (any number will do)
  - Folder goes into heap space
- Execute the ***method*** __init__(args)
- Yield ***the name*** of the object as ***the value***
  - A constructor call is an *expression*, not a command
  - Does not put name in a variable unless you **assign it**

Diagramming Subclasses



superclass-name

Declared in Superclass:

Class Attributes

Method Names

subclass-name(super)

Declared in Subclass:

Class Attributes

Method Names

Important Details:

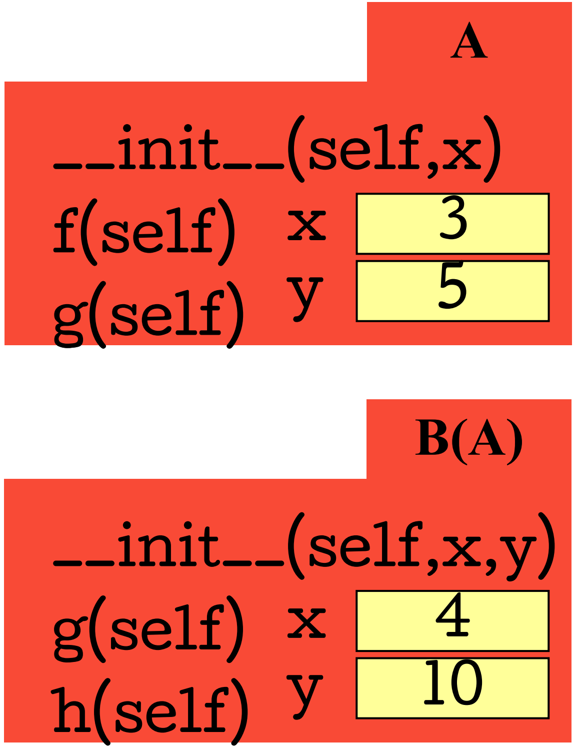
- Make sure you put the superclass-name in parentheses
- Do not duplicate inherited methods and attributes
- Include initializer and and other special methods (as applicable)
- Method parameters are required
- Class attributes are a box with (current) value

Two Example Classes

```
class A(object):
    x=3
    y=5
    def __init__(self,y):
        self.y = y

    def f(self):
        return self.g()

    def g(self):
        return self.x+self.y
```

```
class B(A):
    y=4
    z=10
    def __init__(self,x,y):
        super().__init__(y)
        self.x = x

    def g(self):
        return self.x+self.z
    def h(self):
        return 42
```

Class Folders


A

__init__(self,x)

f(self) x 3

g(self) y 5

B(A)

__init__(self,x,y)

g(self) x 4

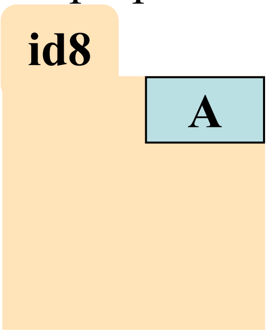
h(self) y 10

These folders will still exist in the following slides, but will not be redrawn; they exist in the heap space along with the object folders.

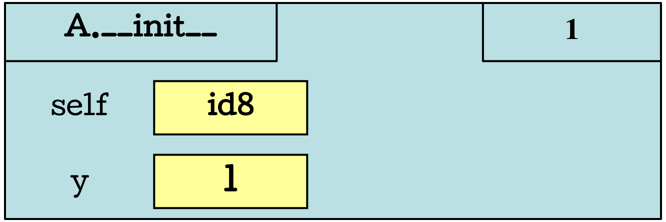
Constructor Examples

Global Space:

Heap Space:



Call Stack:



```
  class A(object):
      x = 3
      y = 5
      def __init__(self,y):
1         self.y = y

  class B(A):
      y = 4
      z = 10
      def __init__(self,x,y):
2         super().__init__(y)
3         self.x = x
```

Call: a = A(1)
b = B(7, 3)

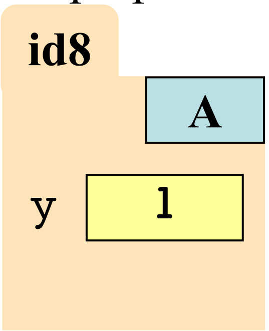
Constructor Examples

Global Space:

```
  class A(object):
      x = 3
      y = 5
      def __init__(self,y):
1         self.y = y

  class B(A):
      y = 4
      z = 10
      def __init__(self,x,y):
2         super().__init__(y)
3         self.x = x
```

Call: a = A(1)
b = B(7, 3)

Heap Space:

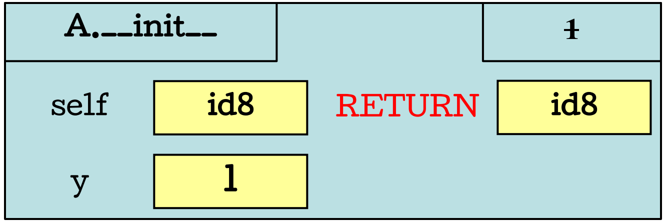


id8

A

y 1

Call Stack:



A.__init__ ~~1~~

self id8

RETURN id8

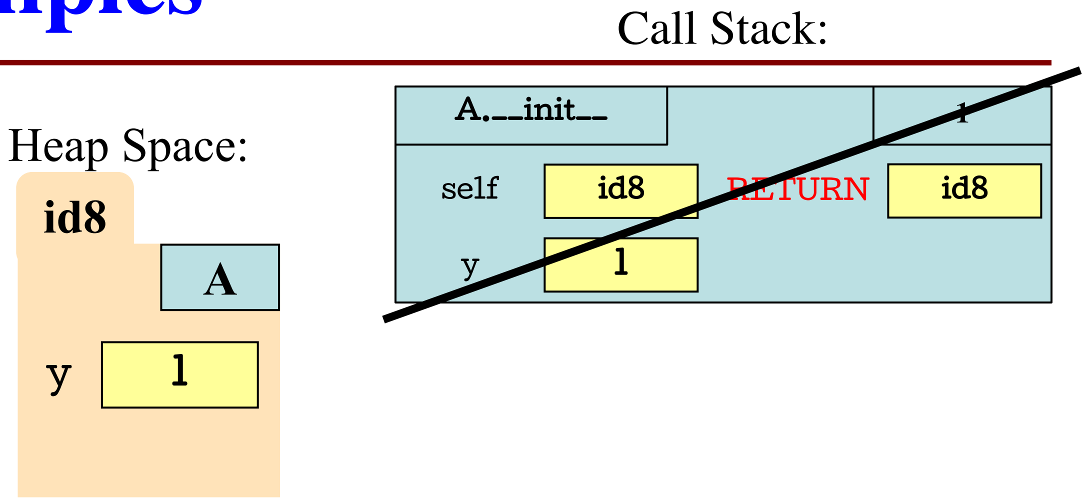
y 1

Constructor Examples

Global Space: a id8

```
  class A(object):
      x = 3
      y = 5
      def __init__(self,y):
1         self.y = y

  class B(A):
      y = 4
      z = 10
      def __init__(self,x,y):
2         super().__init__(y)
3         self.x = x
```

Call: a = A(1)
b = B(7, 3)

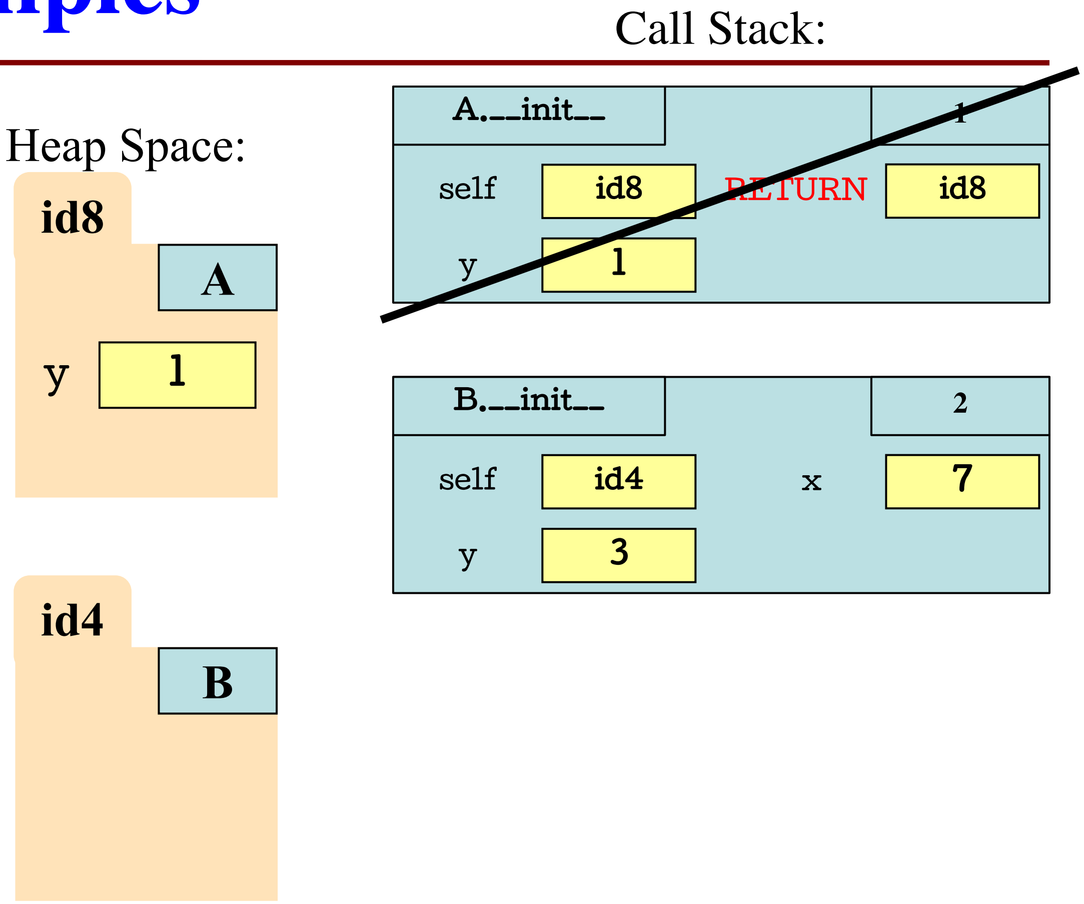


Constructor Examples

Global Space: a id8

```
  class A(object):
      x = 3
      y = 5
      def __init__(self,y):
1         self.y = y

  class B(A):
      y = 4
      z = 10
      def __init__(self,x,y):
2         super().__init__(y)
3         self.x = x
```

Call: a = A(1)
b = B(7, 3)

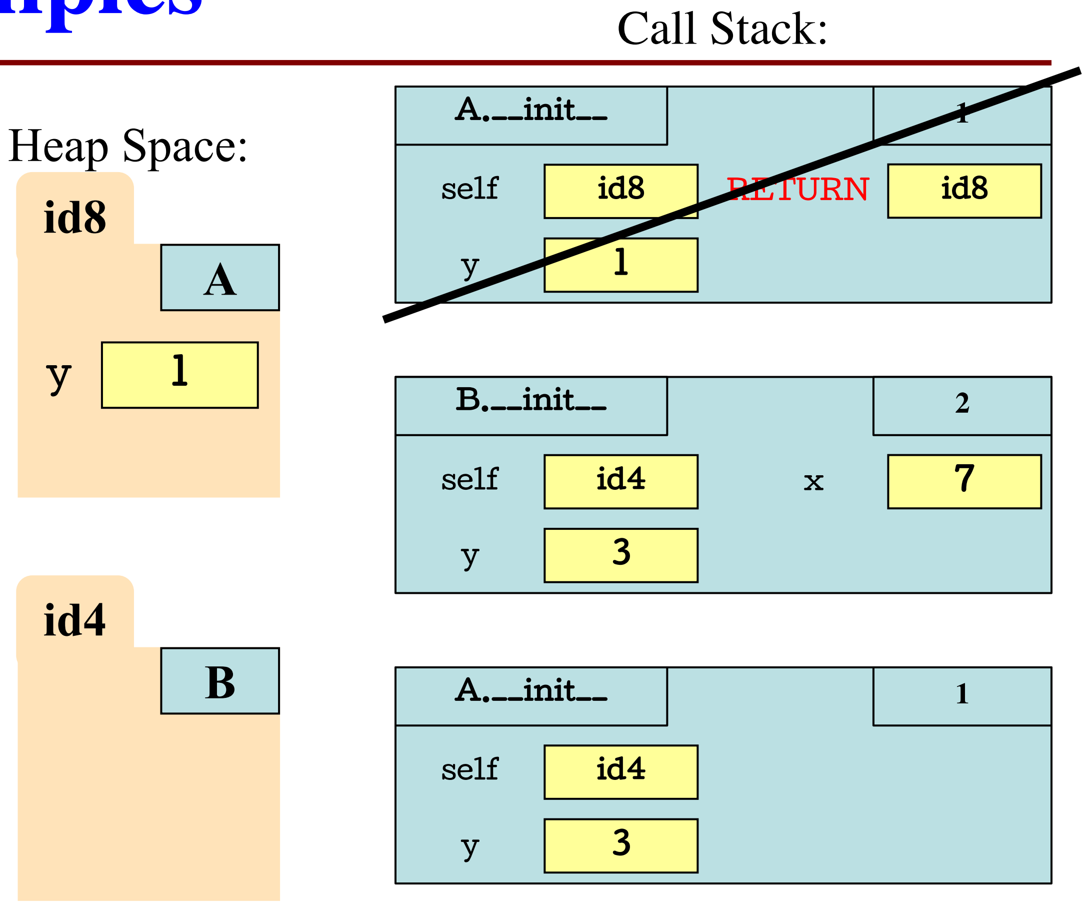


Constructor Examples

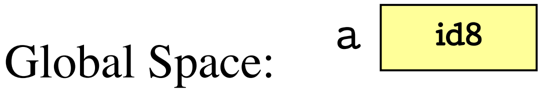
Global Space: a id8

```
  class A(object):
      x = 3
      y = 5
      def __init__(self,y):
1         self.y = y

  class B(A):
      y = 4
      z = 10
      def __init__(self,x,y):
2         super().__init__(y)
3         self.x = x
```

Call: a = A(1)
b = B(7, 3)

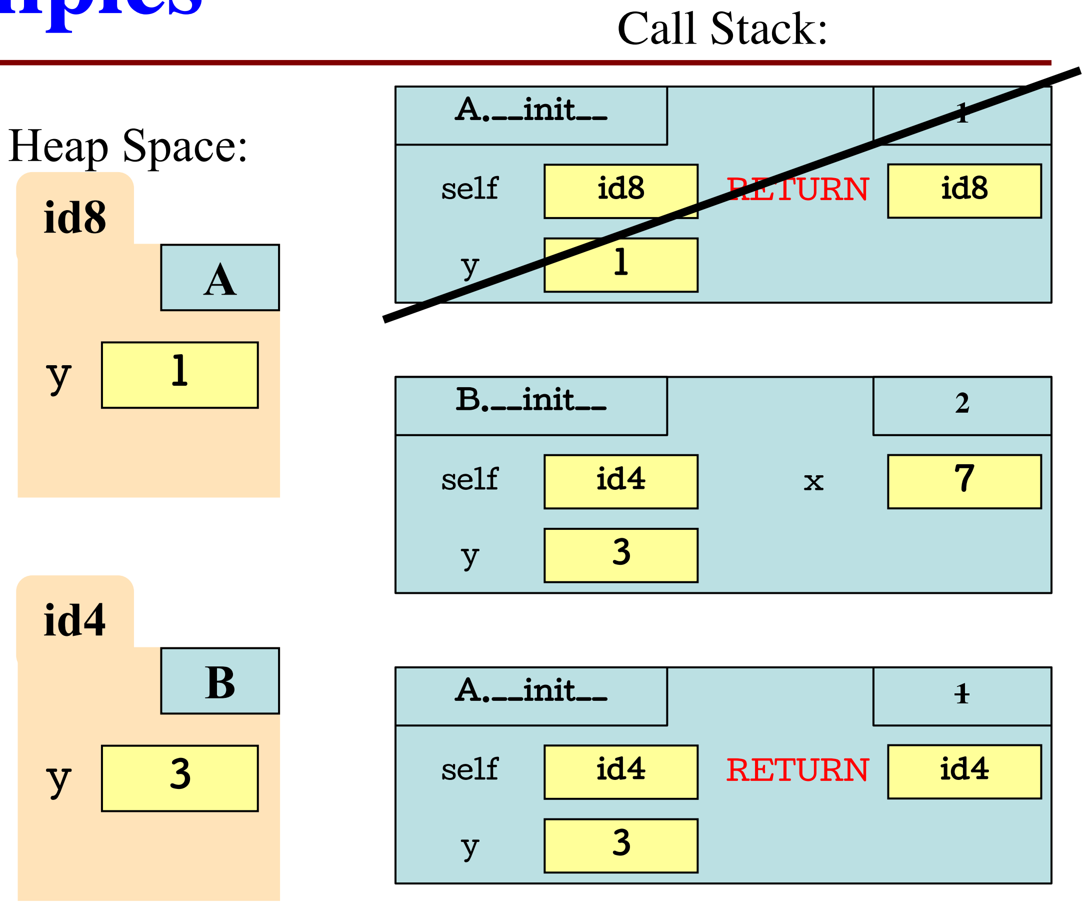


Constructor Examples



```
  class A(object):
      x = 3
      y = 5
      def __init__(self,y):
1         self.y = y

  class B(A):
      y = 4
      z = 10
      def __init__(self,x,y):
2         super().__init__(y)
3         self.x = x
```

Call: a = A(1)
b = B(7, 3)

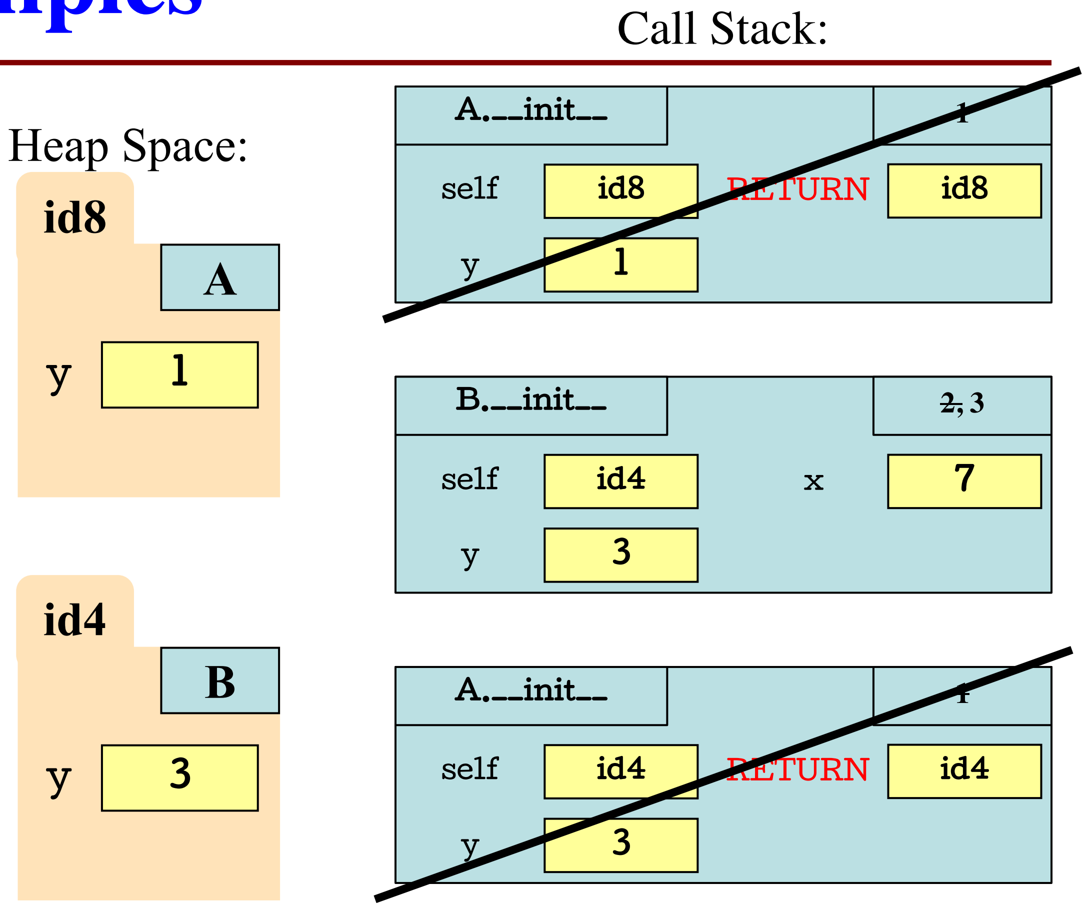


Constructor Examples

Global Space: a id8

```
class A(object):
    x = 3
    y = 5
    def __init__(self,y):
1       self.y = y

class B(A):
    y = 4
    z = 10
    def __init__(self,x,y):
2       super().__init__(y)
3       self.x = x
```

Call: a = A(1)
b = B(7, 3)

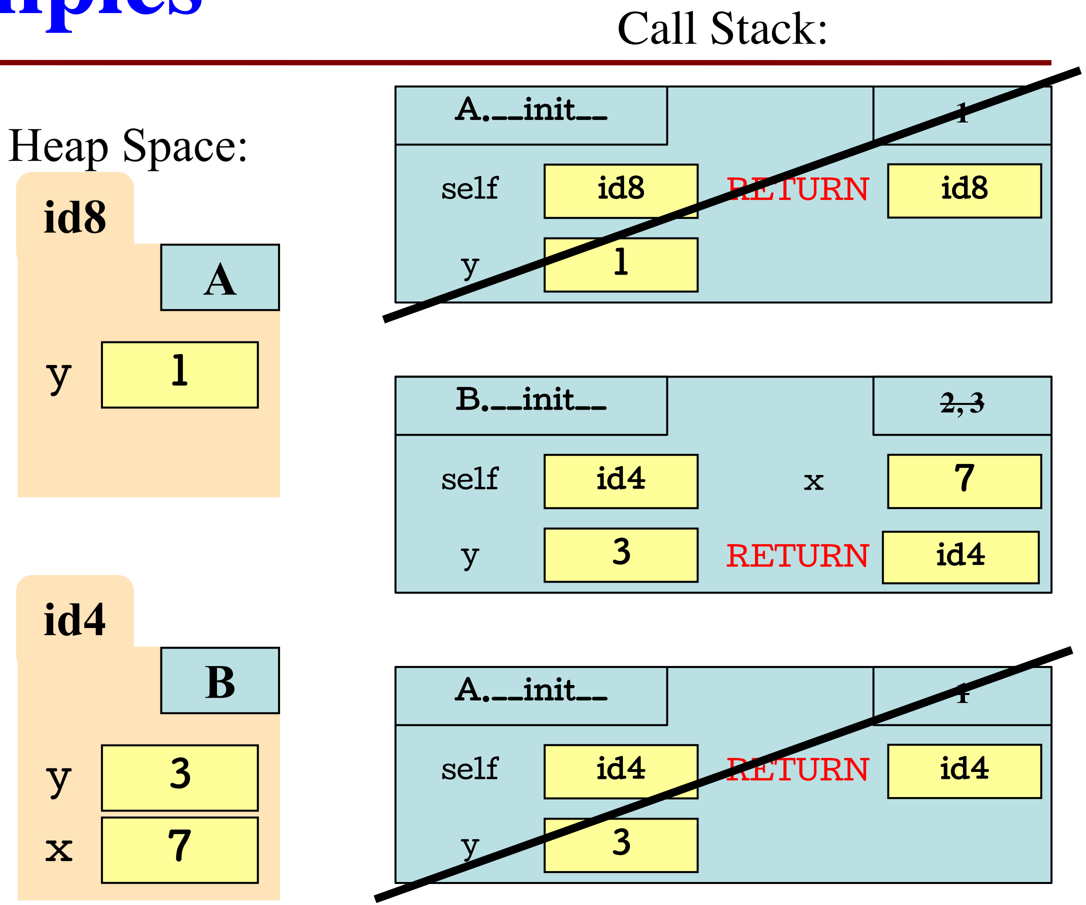


Constructor Examples

Global Space: a id8

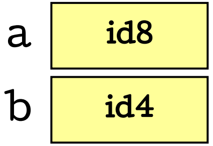
```
  class A(object):
      x = 3
      y = 5
      def __init__(self,y):
1         self.y = y

  class B(A):
      y = 4
      z = 10
      def __init__(self,x,y):
2         super().__init__(y)
3         self.x = x
```

Call: a = A(1)
b = B(7, 3)



Constructor Examples

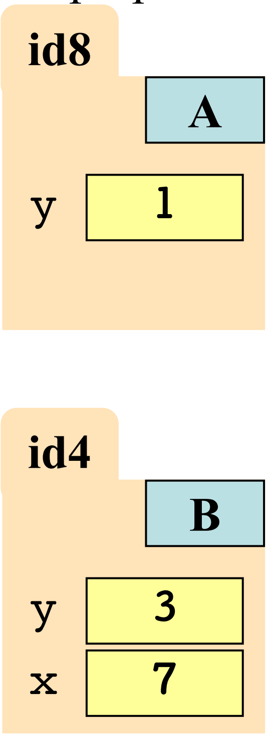
Global Space:



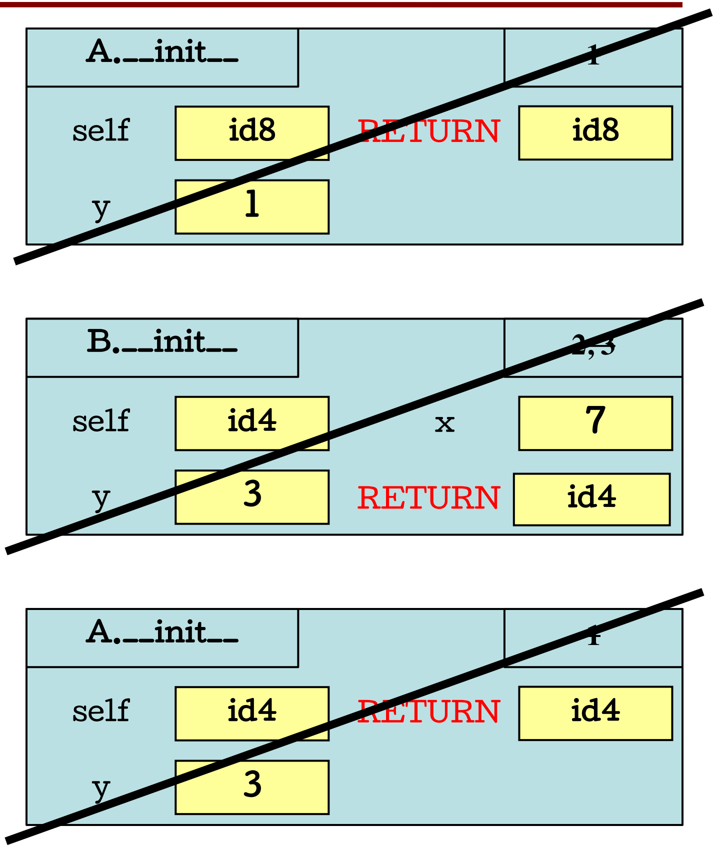
```
  class A(object):
      x = 3
      y = 5
      def __init__(self,y):
1             self.y = y

  class B(A):
      y = 4
      z = 10
      def __init__(self,x,y):
2             super().__init__(y)
3             self.x = x
```

Call: a = A(1)
b = B(7, 3)

Heap Space:



Call Stack:



A.__init__
1
self
id8
RETURN
id8
y
1
B.__init__
2,3
self
id4
x
7
y
3
RETURN
id4
A.__init__
4
self
id4
RETURN
id4
y
3