Presentation 19

Subclasses & Inheritance

Announcements for Today

Assignments

- A4 graded by next week
  - Last call for the Survey
- A5 was posted **Tuesday**
  - Shorter written assignment
  - Due Tuesday at Midnight
- A6 also posted **Tuesday**
  - Due **Sunday** after classes
  - Designed to take two weeks
  - Follow **micro-deadlines!**

Video Lessons

- **Lesson 22** for today
- **Videos 23.1-23.7** next time
- **Video 23.8** is *very* optional

Second Prelim is Coming!

- We are now two weeks out!
  - Thursday November 19th at 9:30 am (Eastern)
  - Will cover up to November 12th
  - **Topics:** Loops, Recursion, and Classes
- **CMS** is now open for conflicts again
  - Time zone conflicts will be resolved *later*
  - Submit if you want/need to move online
  - SDS students should submit again
  - Want to assign groups by next Thurday

Second Prelim is Coming!

- We are now two weeks out!
  - Thursday November 19th at 9:30 am (Eastern)
  - Will cover up to November 12th
  - To
- CMS
  - Time zone conflicts will be resolved *later*
  - Submit if you want/need to move online
  - SDS students should submit again
  - Want to assign groups by next Thursday

Study Guide posted next week

Subclass Constructors

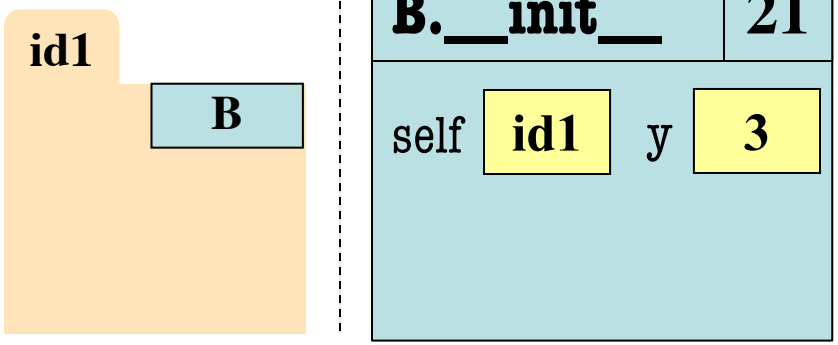
```
class A(object):

12    def __init__(self,x):
13        self.x = x
14        self.y = x

class B(A):

20    def __init__(self,y):
21        super().__init__(y+2)
22        self.x = y
```

```
>>> b = B(3)
```

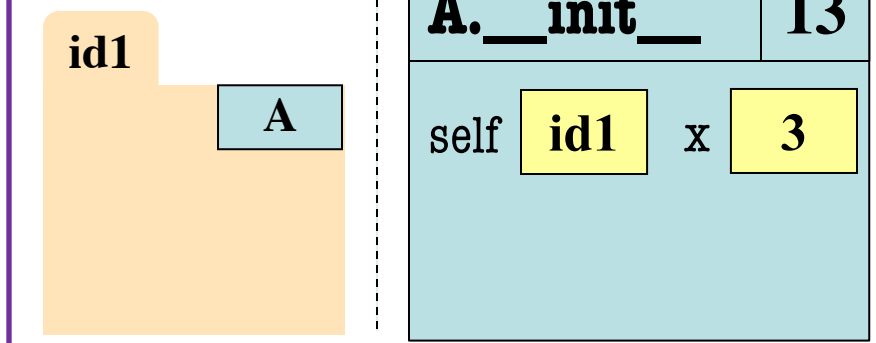
Ignoring the **class folder** what does the **call stack** and the **heap** look like?

Which One is Closest to Your Answer?

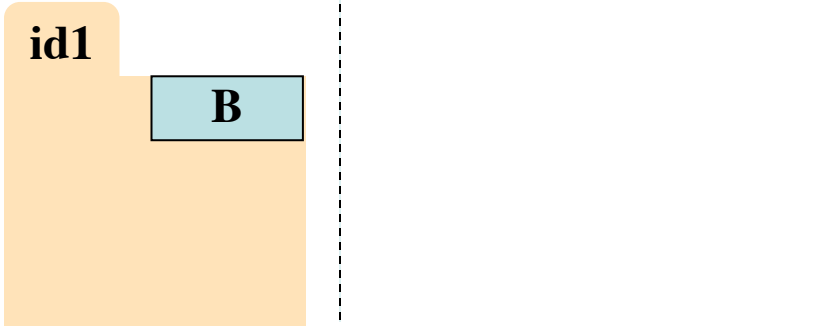
A:



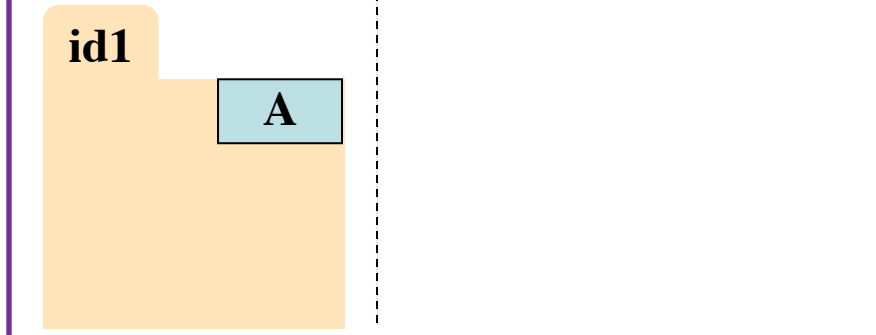
B:



C:



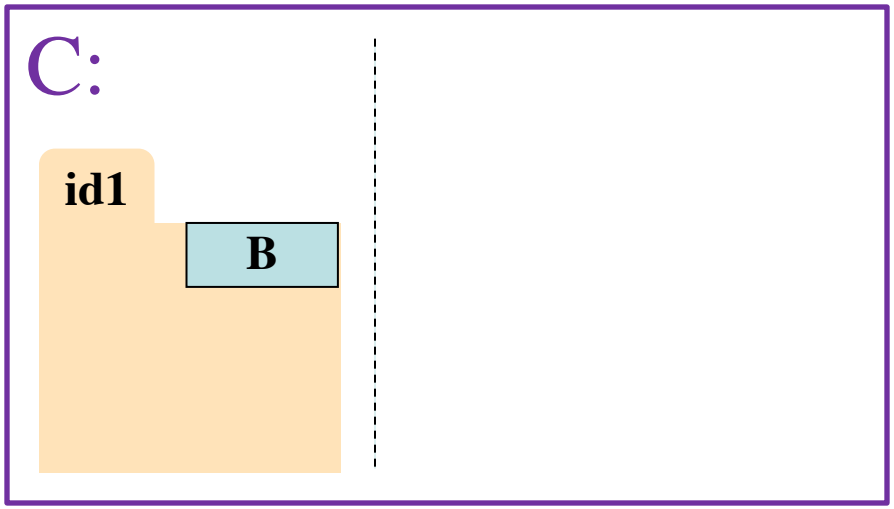
D:



Subclass Constructors

```
class A(object):

    def __init__(self,x):
        self.x = x
        self.y = x

class B(A):

    def __init__(self,y):
        super().__init__(y+2)
        self.x = y
```

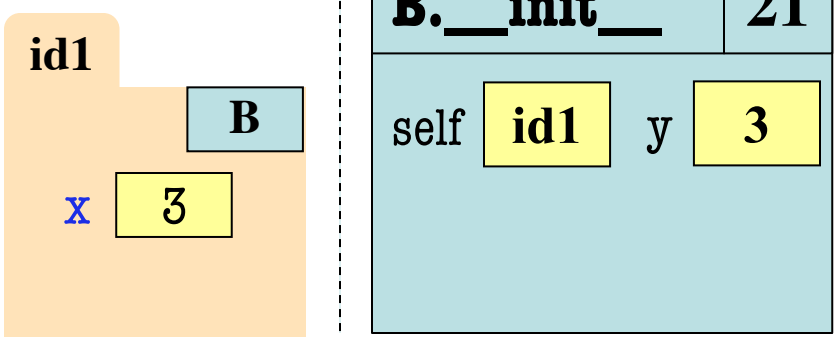
```
>>> b = B(3)
```



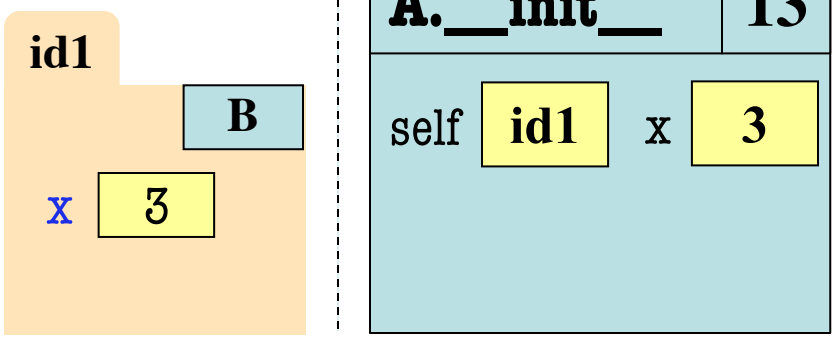
What is the **next step**?

Which One is Closest to Your Answer?

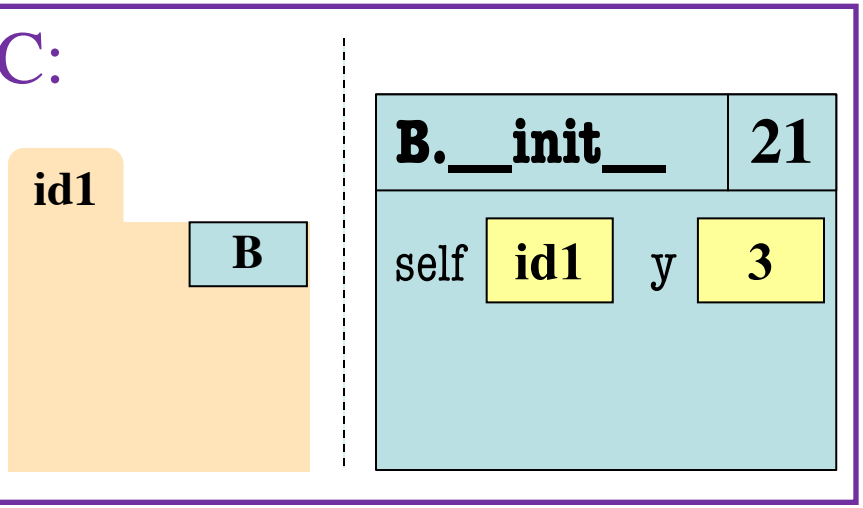
A:



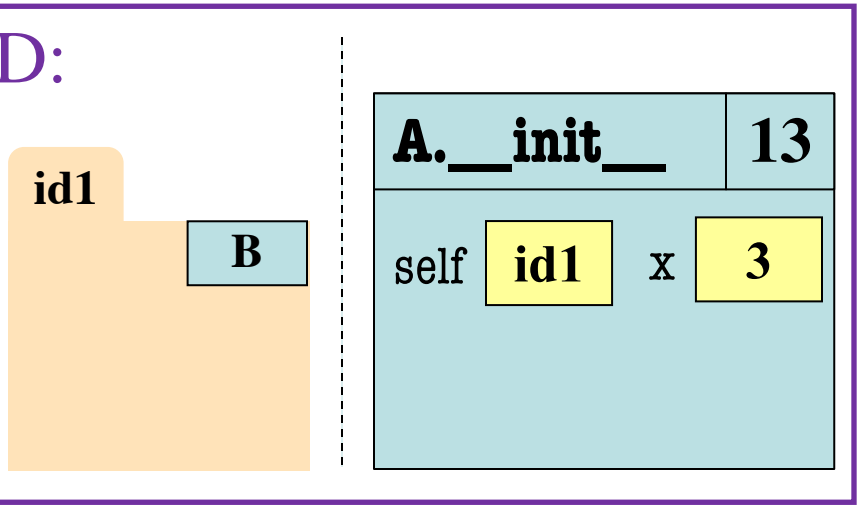
B:



C:



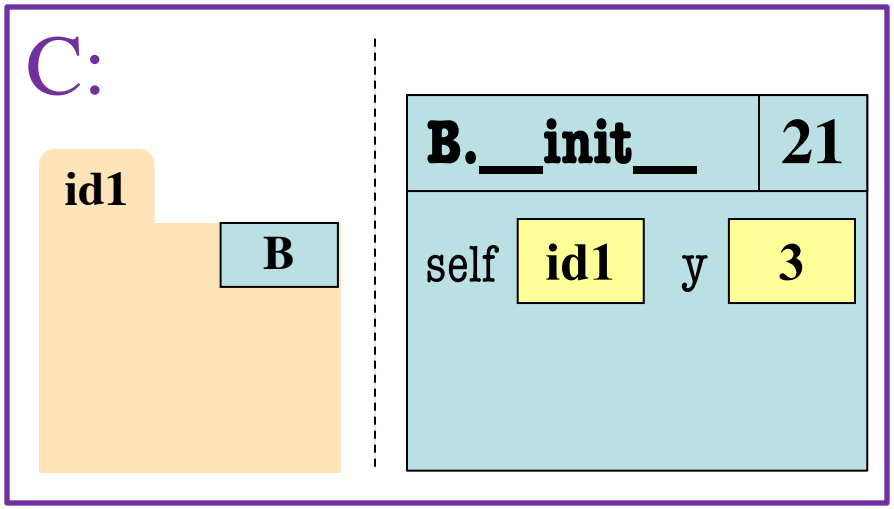
D:



Subclass Constructors

```
class A(object):


    def __init__(self,x):
        self.x = x
        self.y = x

class B(A):


    def __init__(self,y):
        super().__init__(y+2)
        self.x = y
```

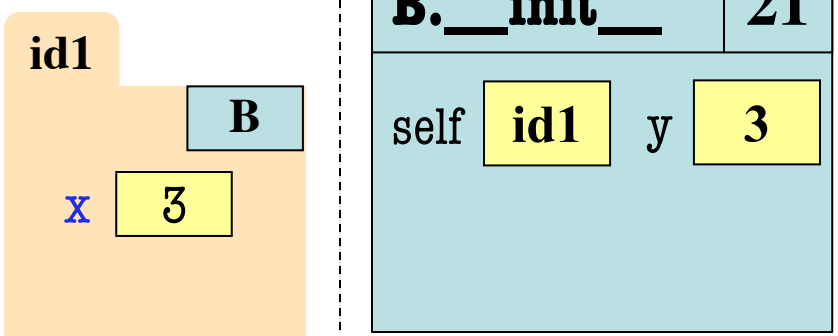
```
>>> b = B(3)
```



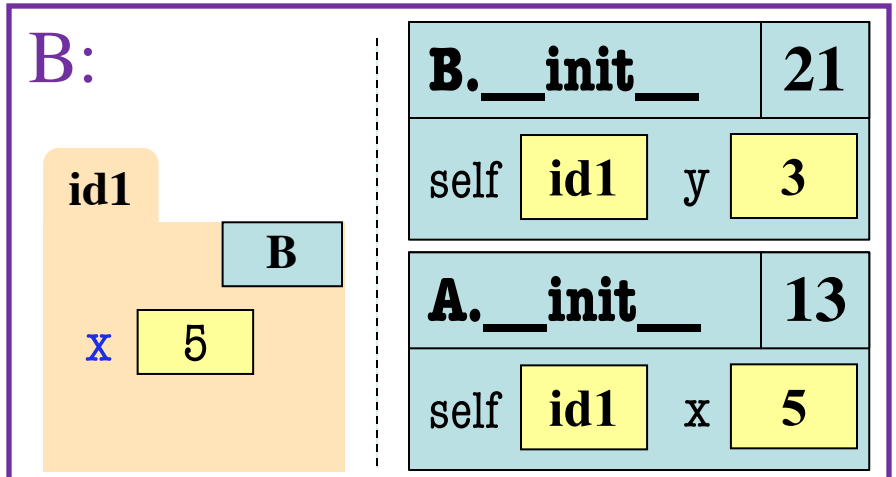
What is the **next step**?

Which One is Closest to Your Answer?

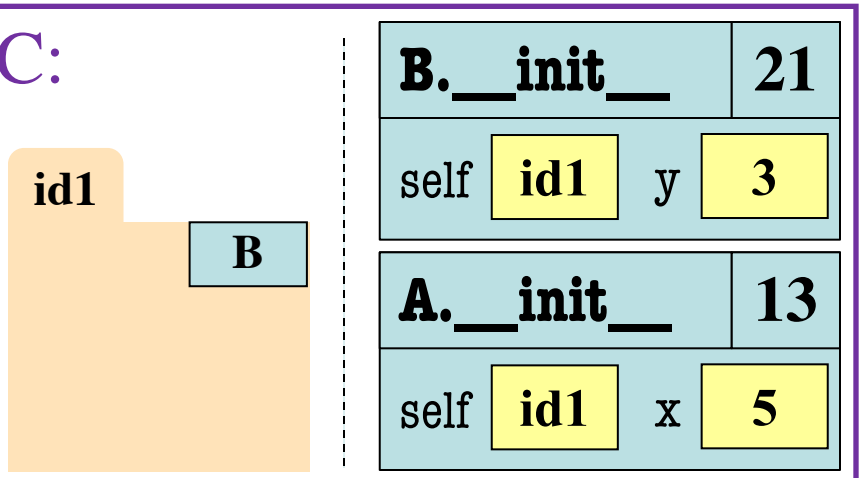
A:



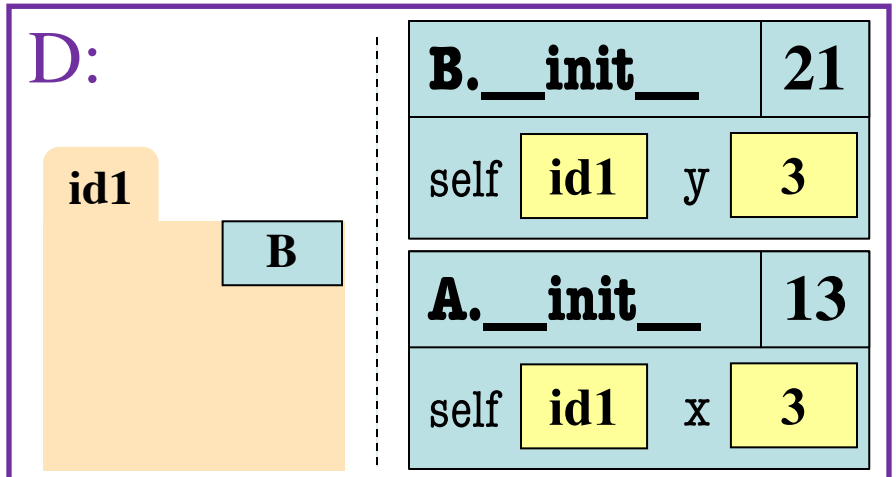
B:



C:



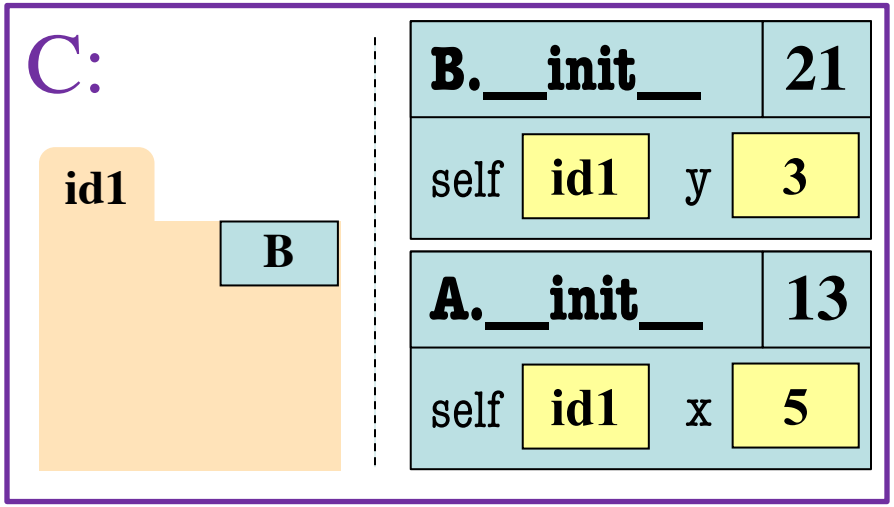
D:



Subclass Constructors

```
class A(object):

12    def __init__(self,x):
13        self.x = x
14        self.y = x

class B(A):

20    def __init__(self,y):
21        super().__init__(y+2)
22        self.x = y
```

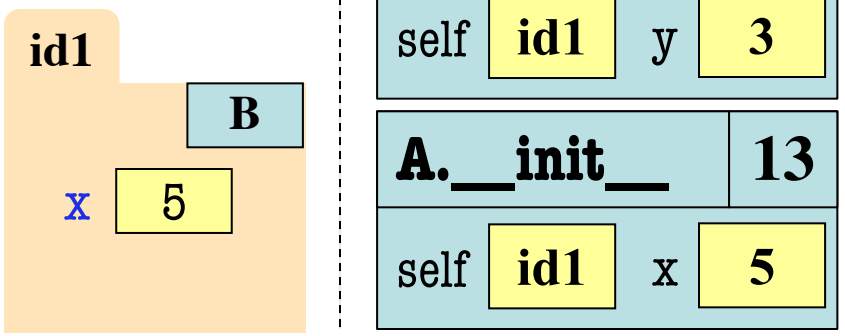
```
>>> b = B(3)
```



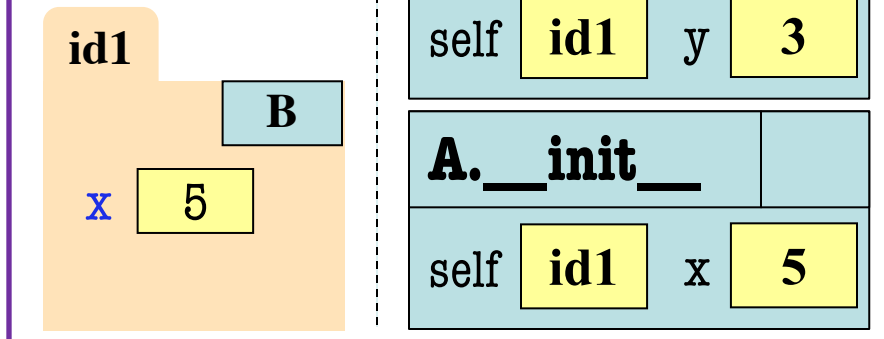
What is the **next step**?

Which One is Closest to Your Answer?

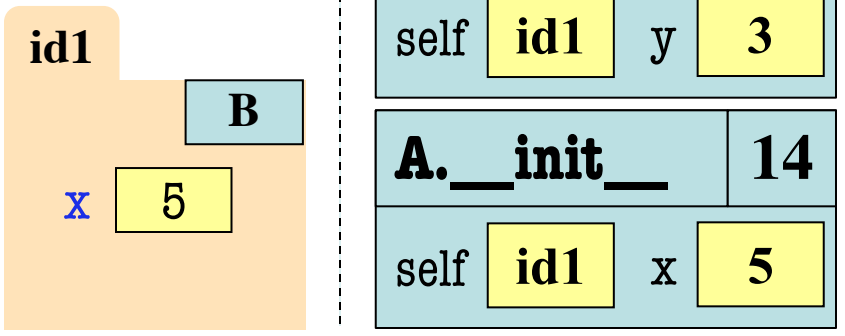
A:



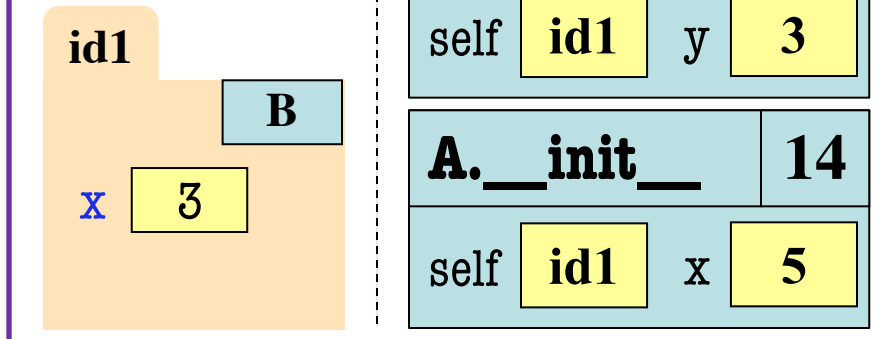
B:



C:



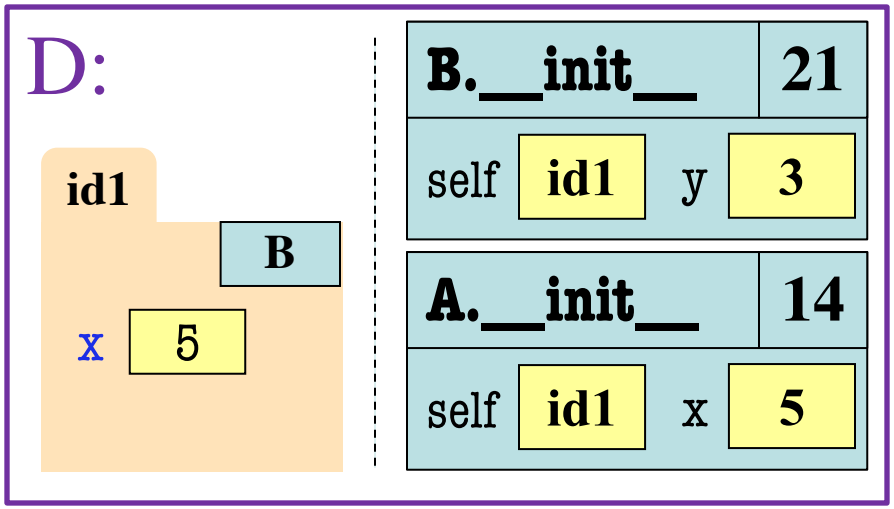
D:



Subclass Constructors

```
class A(object):

    def __init__(self,x):
        self.x = x
        self.y = x

class B(A):

    def __init__(self,y):
        super().__init__(y+2)
        self.x = y
```

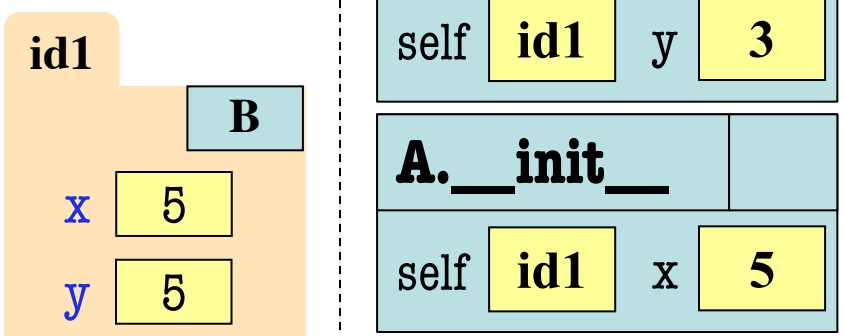
```
>>> b = B(3)
```



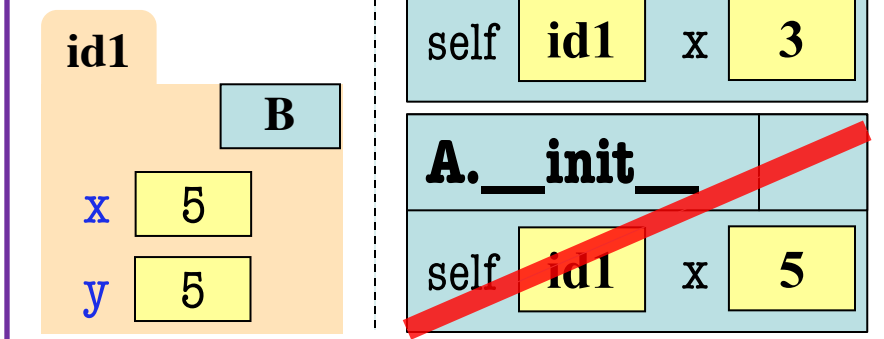
What is the **next step**?

Which One is Closest to Your Answer?

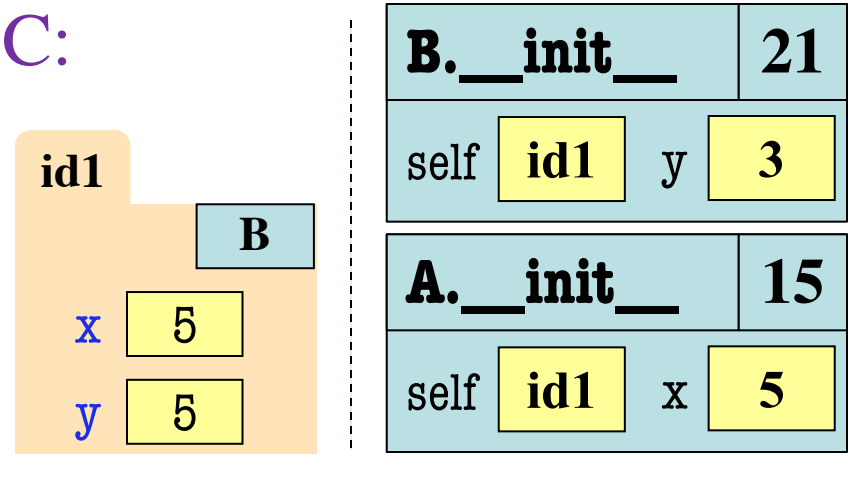
A:



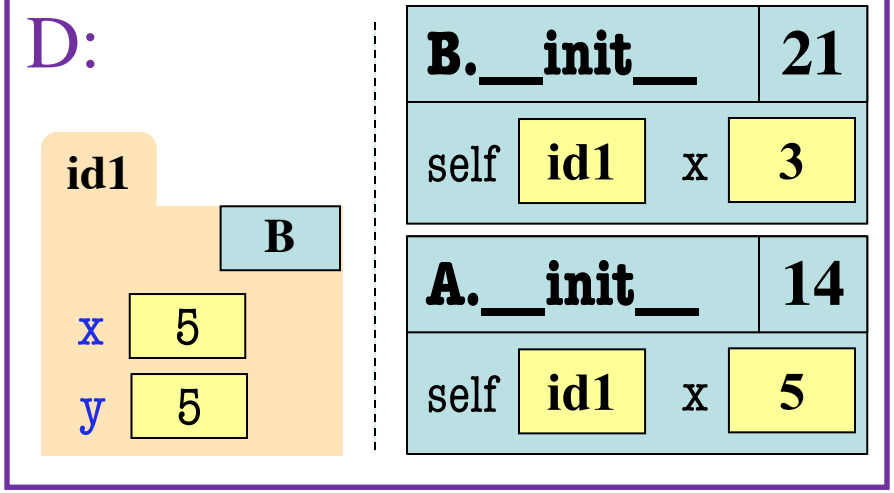
B:



C:



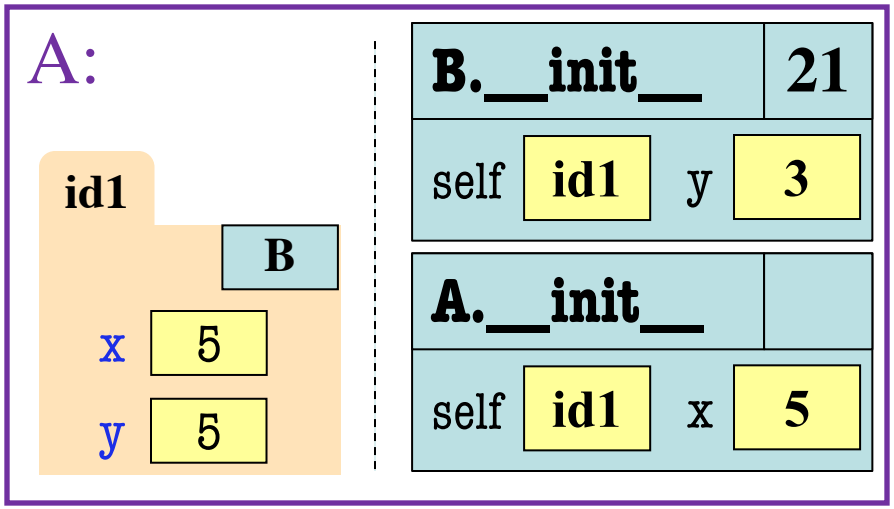
D:



Subclass Constructors

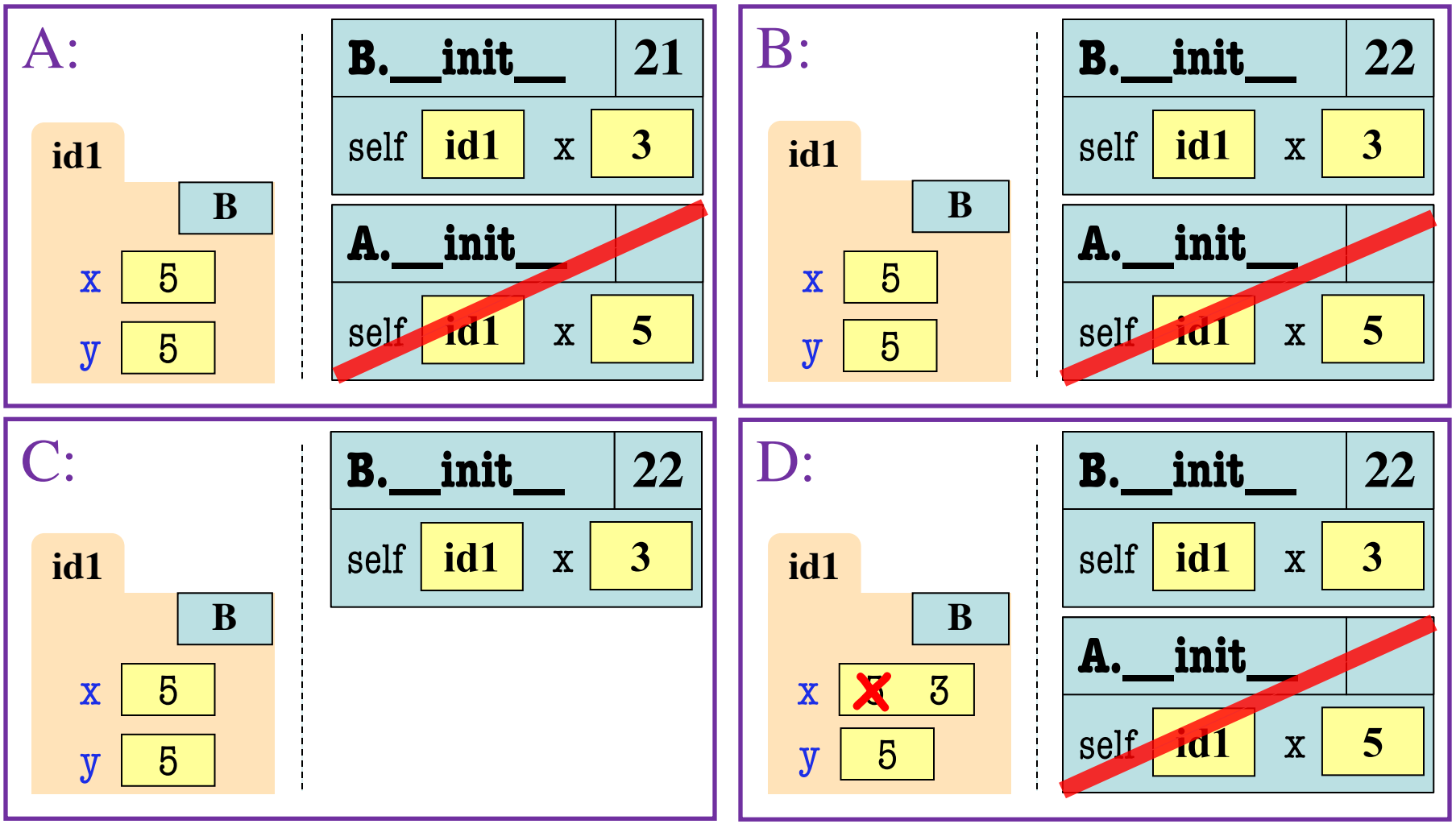
```
class A(object):

    def __init__(self,x):
        self.x = x
        self.y = x

class B(A):

    def __init__(self,y):
        super().__init__(y+2)
        self.x = y
```

```
>>> b = B(3)
```



What is the **next step**?

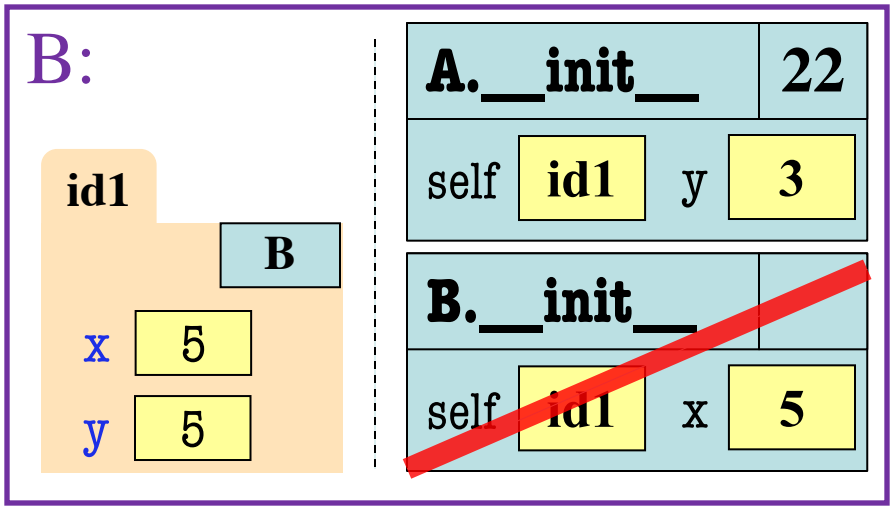
Which One is Closest to Your Answer?



Subclass Constructors

```
class A(object):

    def __init__(self,x):
        self.x = x
        self.y = x

class B(A):

    def __init__(self,y):
        super().__init__(y+2)
        self.x = y
```

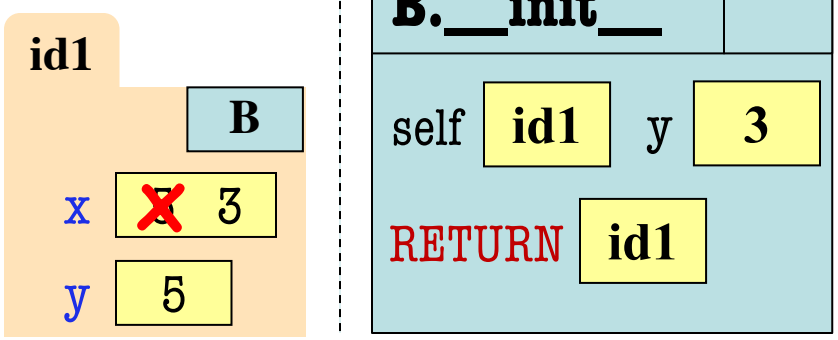
```
>>> b = B(3)
```



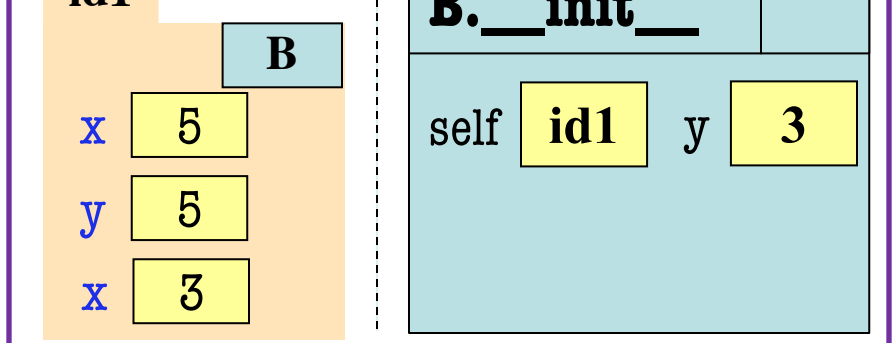
What is the **next step**?

Which One is Closest to Your Answer?

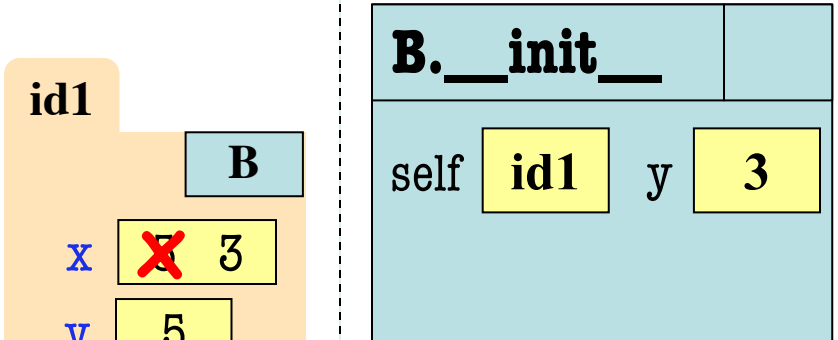
A:



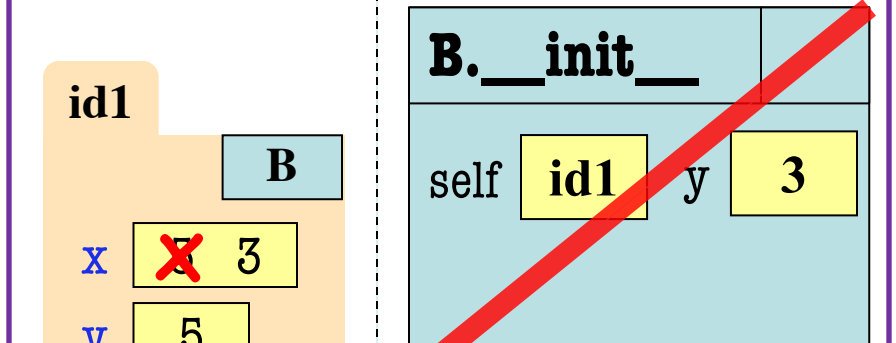
B:



C:



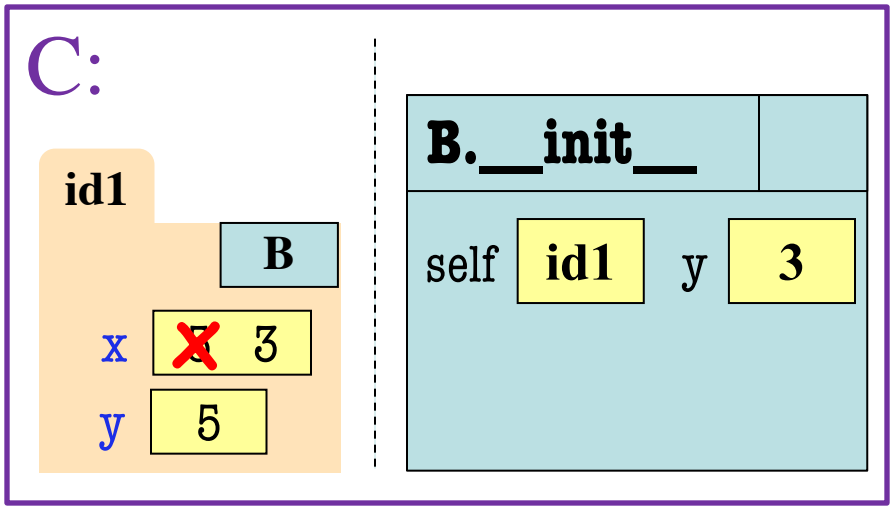
D:



Subclass Constructors

```
class A(object):

    def __init__(self,x):
        super().__init__(x+2)
        self.x = x

class B(A):

    def __init__(self,y):
        self.x = y
        self.y = y
```

```
>>> b = B(3)
```



Final step is **erase frame**

Name Resolution

```
class A(object):
    x = 3 # Class Attribute
    y = 5 # Class Attribute

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

    def g(self):
        return 10

class B(A):
    y = 4    # Class Attribute
    z = 42   # Class Attribute

    def g(self):
        return 14

    def h(self):
        return 18
```

- Execute the following:

```
>>> a = A()
>>> b = B()
```

- What is value of a.f()?

A: 10
B: 14
C: 5
D: **ERROR**
E: I don't know

Name Resolution

```
class A(object):
    x = 3 # Class Attribute
    y = 5 # Class Attribute

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

    def g(self):
        return 10

class B(A):
    y = 4    # Class Attribute
    z = 42   # Class Attribute

    def g(self):
        return 14

    def h(self):
        return 18
```

- Execute the following:

```
>>> a = A()
>>> b = B()
```

- What is value of a.f()?

A: 10 **CORRECT**
B: 14
C: 5
D: **ERROR**
E: I don't know

Name Resolution

```
class A(object):
    x = 3 # Class Attribute
    y = 5 # Class Attribute

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

    def g(self):
        return 10

class B(A):
    y = 4    # Class Attribute
    z = 42   # Class Attribute

    def g(self):
        return 14

    def h(self):
        return 18
```

- Execute the following:

```
>>> a = A()
>>> b = B()
```

- What is value of b.f()?

A: 10
B: 14
C: 5
D: **ERROR**
E: I don't know

Name Resolution

```
class A(object):
    x = 3 # Class Attribute
    y = 5 # Class Attribute

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

    def g(self):
        return 10

class B(A):
    y = 4    # Class Attribute
    z = 42   # Class Attribute

    def g(self):
        return 14

    def h(self):
        return 18
```

- Execute the following:

```
>>> a = A()
>>> b = B()
```

- What is value of b.f()?

A: 10
B: 14 **CORRECT**
C: 5
D: **ERROR**
E: I don't know

Name Resolution

```
class A(object):
    x = 3 # Class Attribute
    y = 5 # Class Attribute

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

    def g(self):
        return 10

class B(A):
    y = 4    # Class Attribute
    z = 42   # Class Attribute

    def g(self):
        return 14

    def h(self):
        return 18
```

- Execute the following:

```
>>> a = A()
>>> b = B()
```

- What is value of b.x?

A: 4
B: 3
C: 42
D: **ERROR**
E: I don't know

Name Resolution

```
class A(object):
    x = 3 # Class Attribute
    y = 5 # Class Attribute

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

    def g(self):
        return 10

class B(A):
    y = 4    # Class Attribute
    z = 42   # Class Attribute

    def g(self):
        return 14

    def h(self):
        return 18
```

- Execute the following:

```
>>> a = A()
>>> b = B()
```

- What is value of b.x?

A: 4
B: 3 **CORRECT**
C: 42
D: **ERROR**
E: I don't know

Name Resolution

```
class A(object):
    x = 3 # Class Attribute
    y = 5 # Class Attribute

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

    def g(self):
        return 10

class B(A):
    y = 4    # Class Attribute
    z = 42   # Class Attribute

    def g(self):
        return 14

    def h(self):
        return 18
```

- Execute the following:

  ```
  >>> a = A()
  >>> b = B()
  ```

- What is value of a.z?

A: 4
B: 3
C: 42
D: **ERROR**
E: I don't know

Name Resolution

```
class A(object):
    x = 3 # Class Attribute
    y = 5 # Class Attribute

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

    def g(self):
        return 10

class B(A):
    y = 4    # Class Attribute
    z = 42   # Class Attribute

    def g(self):
        return 14

    def h(self):
        return 18
```

- Execute the following:

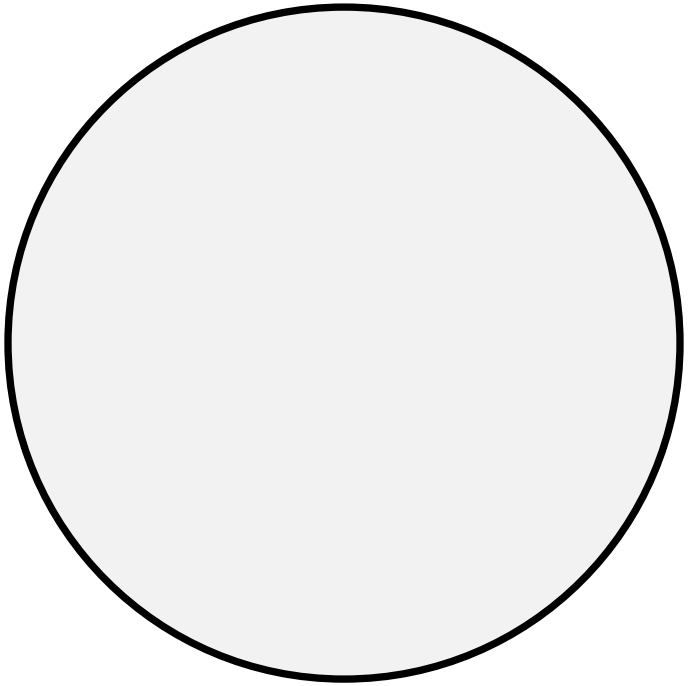
  >>> a = A()

  >>> b = B()

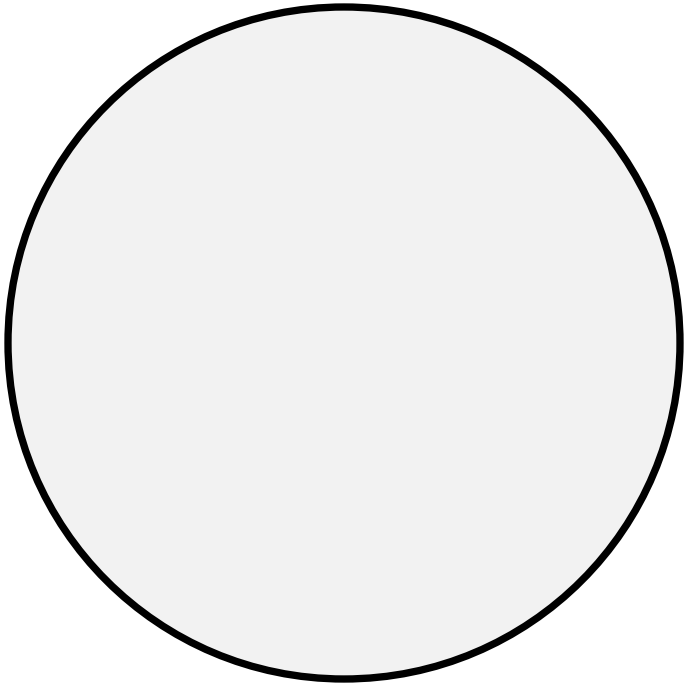
- What is value of a.z?

A: 4
B: 3
C: 42
D: **ERROR** **CORRECT**
E: I don't know

Design Time: A (2D) Circle

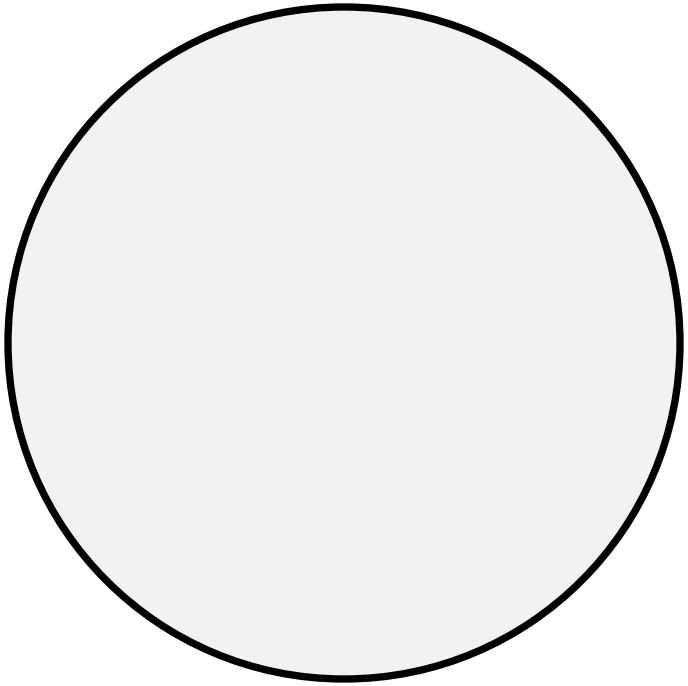


Design Time: A (2D) Circle



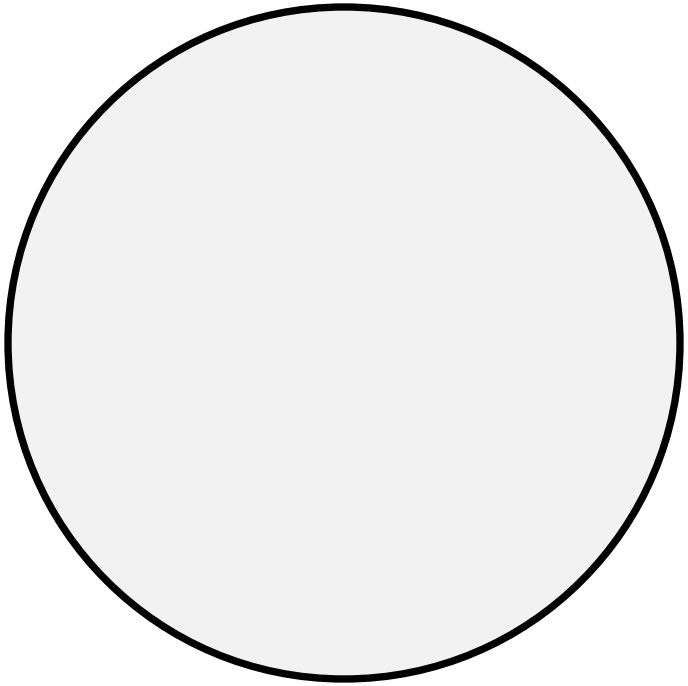
What are the **Attributes**?

Design Time: A (2D) Circle



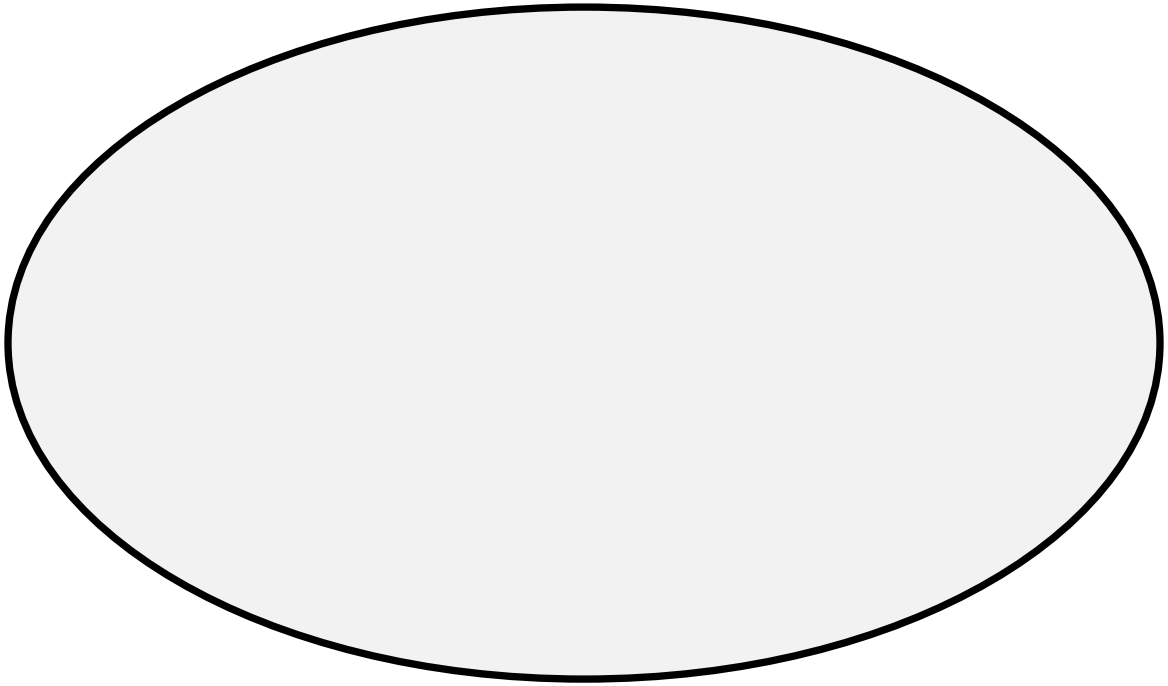
What are the **Methods**?

Design Time: A (2D) Circle

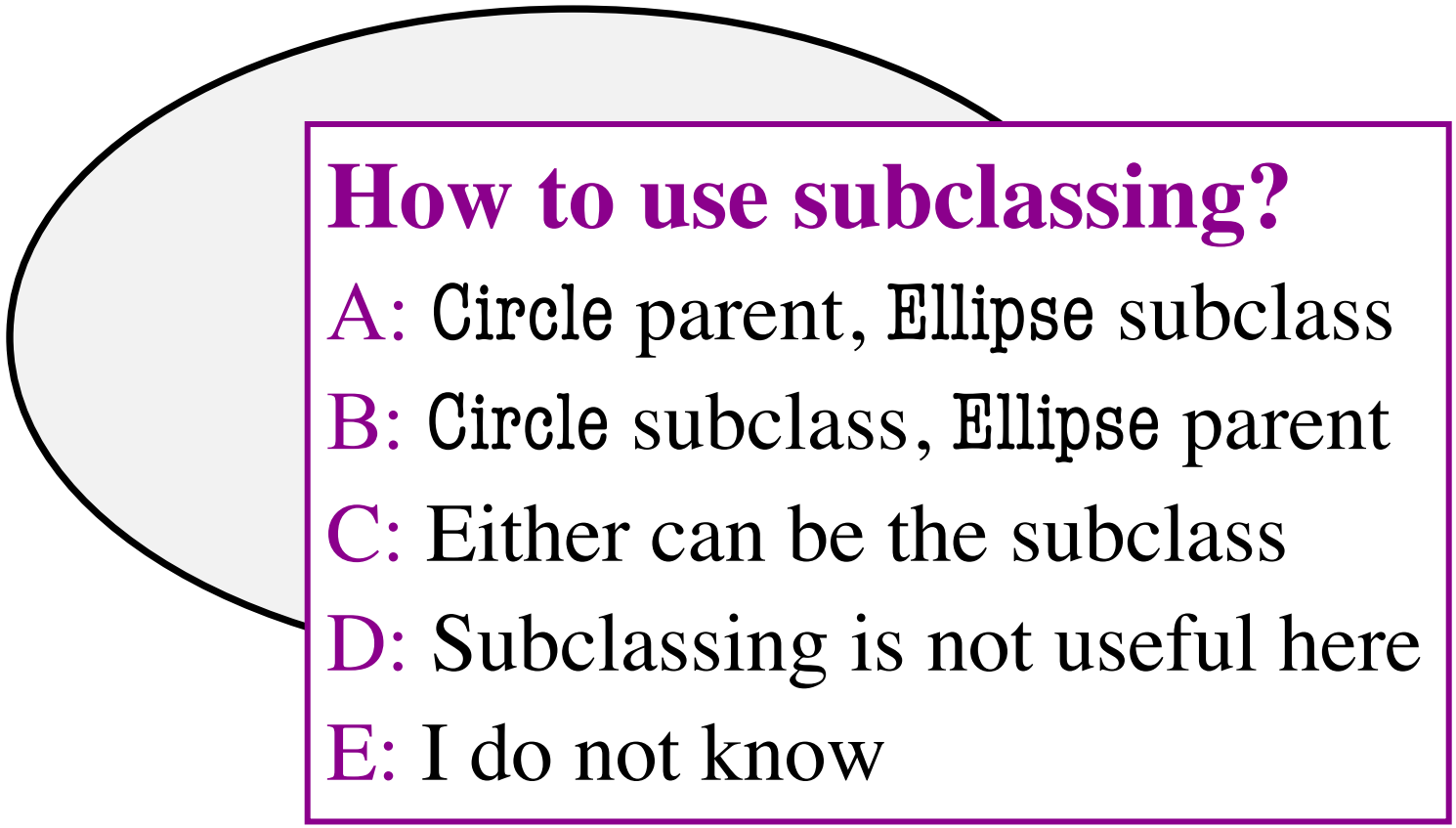


What are the **Invariants**?

Design Time: A (2D) ~~Circle~~ Ellipse



Design Time: A (2D) ~~Circle~~ Ellipse



How to use subclassing?

A: Circle parent, Ellipse subclass

B: Circle subclass, Ellipse parent

C: Either can be the subclass

D: Subclassing is not useful here

E: I do not know

Questions?