Name: ____ **Section:** 1 2 3 4 5

As you complete each problem, ask a student assistant to check your answer.

Throughout these problems:

- **Use the boxes we supplied**; just ***add labels*** and ***arrows*** for variables and ***data*** for non-container objects.
- Assume the existence of a ***Point*** class with just two instance variables (***x*** and ***y***).
- Assume the existence of a ***Circle*** class with just two instance variables (***center*** and ***radius***, where *center* is a Point object). Assume that a Circle object stores, as its center, a ***reference*** to the Point object that it is given and ***not a copy*** of that Point.

As a reminder, here are the four rules for drawing box-and-pointer diagrams, followed by an example from the video.

Rule 1: Draw a ***NON-container object*** by putting its value inside a box.

Rule 2: Draw a ***variable*** (aka ***name***) using a box labeled with the variable's name and with arrows from the box to the object to which the variable currently refers.

Rule 3: Draw a ***CONTAINER object*** by making a box for it, and then creating sub-boxes that are drawn as if they were variables, but with names for the ***instance variables*** of an object and indices for items of a sequence. (We will talk about sequences later in the course.)

Rule 4: When code RE-assigns a variable, as in `x = blah`:

- Evaluate the expression on the right-hand-side. If it is a new object, draw a box for it.
- Cross through the existing arrow (if any) from the variable.
- Draw a NEW arrow from the variable to the object to which the right-hand-side evaluated.

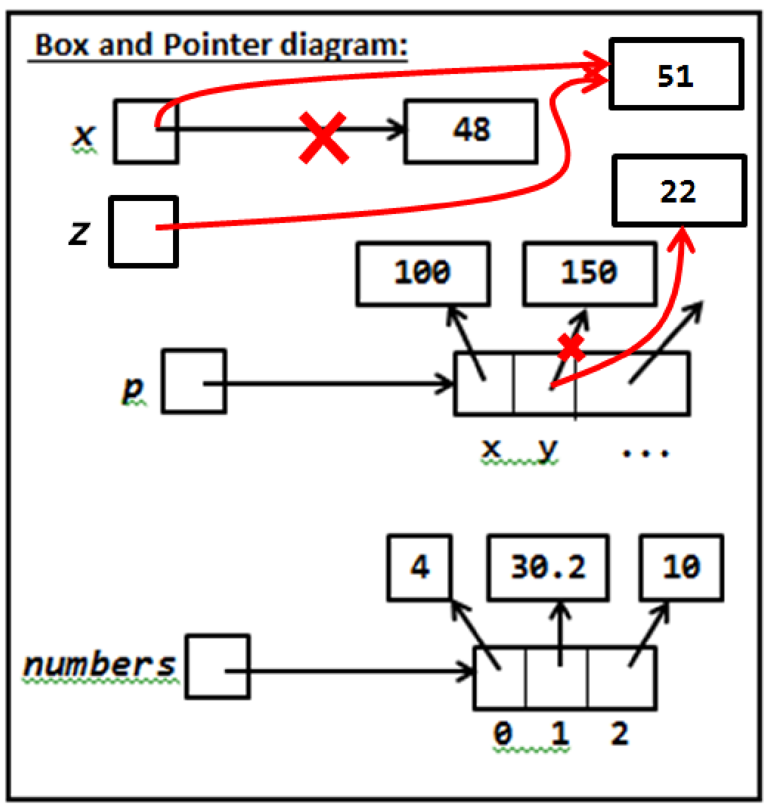
Arrows ALWAYS go:

from a ***variable's*** box

to an ***object's*** box.

Arrows NEVER go from a ***variable's*** box to ***another variable's*** box.

```
x = 48
p = Point(100, 150)
numbers = [4, 30.2, 10]
x = x + 3
p.y = 22
z = x
```



1. Using the diagram at the bottom of this page, ***draw a Box-and-Pointer diagram*** that shows what happens when the following statements execute. ***Then indicate what output is printed***. We already supplied the boxes for the diagram; you label them and draw arrows.

```
x = 33
r = 20
p = Point(80, 50)
c = Circle(p, r)

print('x:',   x)
print('r:',   r)
print('p.x:', p.x)
print('p.y:', p.y)
print('c.center.x:', c.center.x)
print('c.center.y:', c.center.y)
print('c.radius:',   c.radius)
```

Output:

x: ____________

r: ____________

p.x: ____________

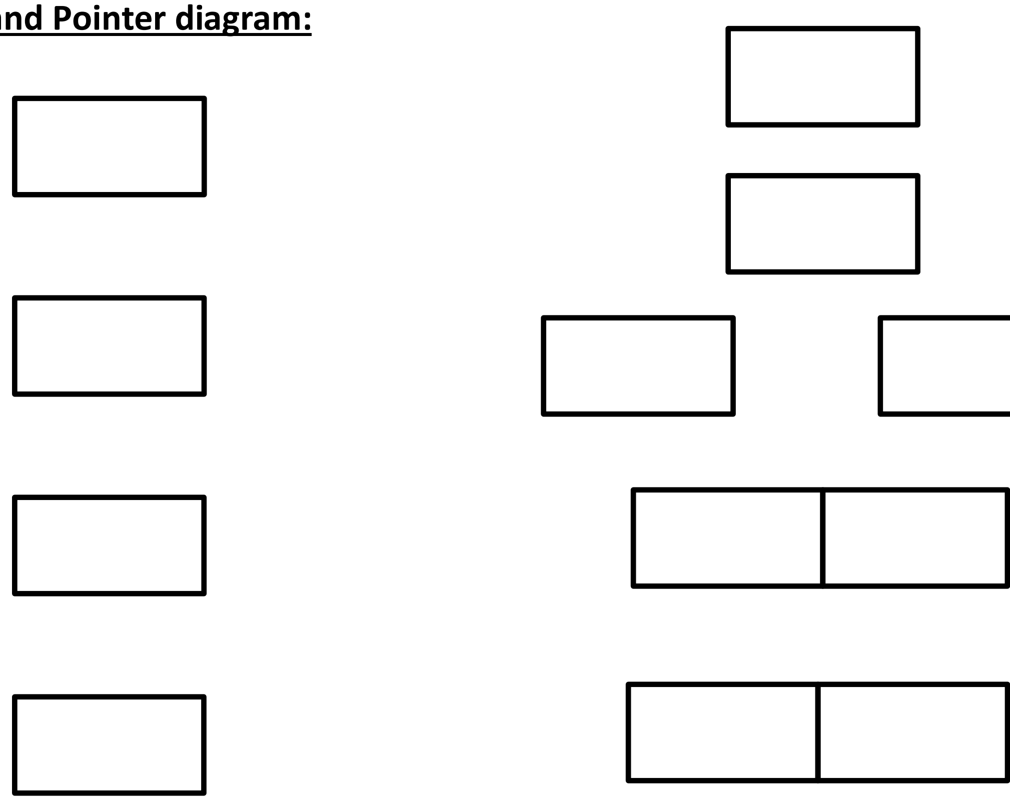
p.y: ____________

c.center.x: ____________

c.center.y: ____________

c.radius: ____________

Box and Pointer diagram:



2. This problem continues the previous one. We have drawn a ***SOLUTION*** to the previous problem below. Use it to check your answer to the previous problem. Then augment the box-and-pointer diagram below to include the new statements in the code below. Also indicate what output is printed by the *print* statements that follow that new code.

```
x = 33
r = 20
p = Point(80, 50)
c = Circle(p, r)
<same print statements as in problem 1>

r = 77
p.x = 44

<same print statements as in problem 1,
repeated here>
```

Previous problem printed these numbers.

New code is here

33
20
80
50
80
50
20

Output from 2nd set of *print* statements:

x: ____________

r: ____________

p.x: ____________

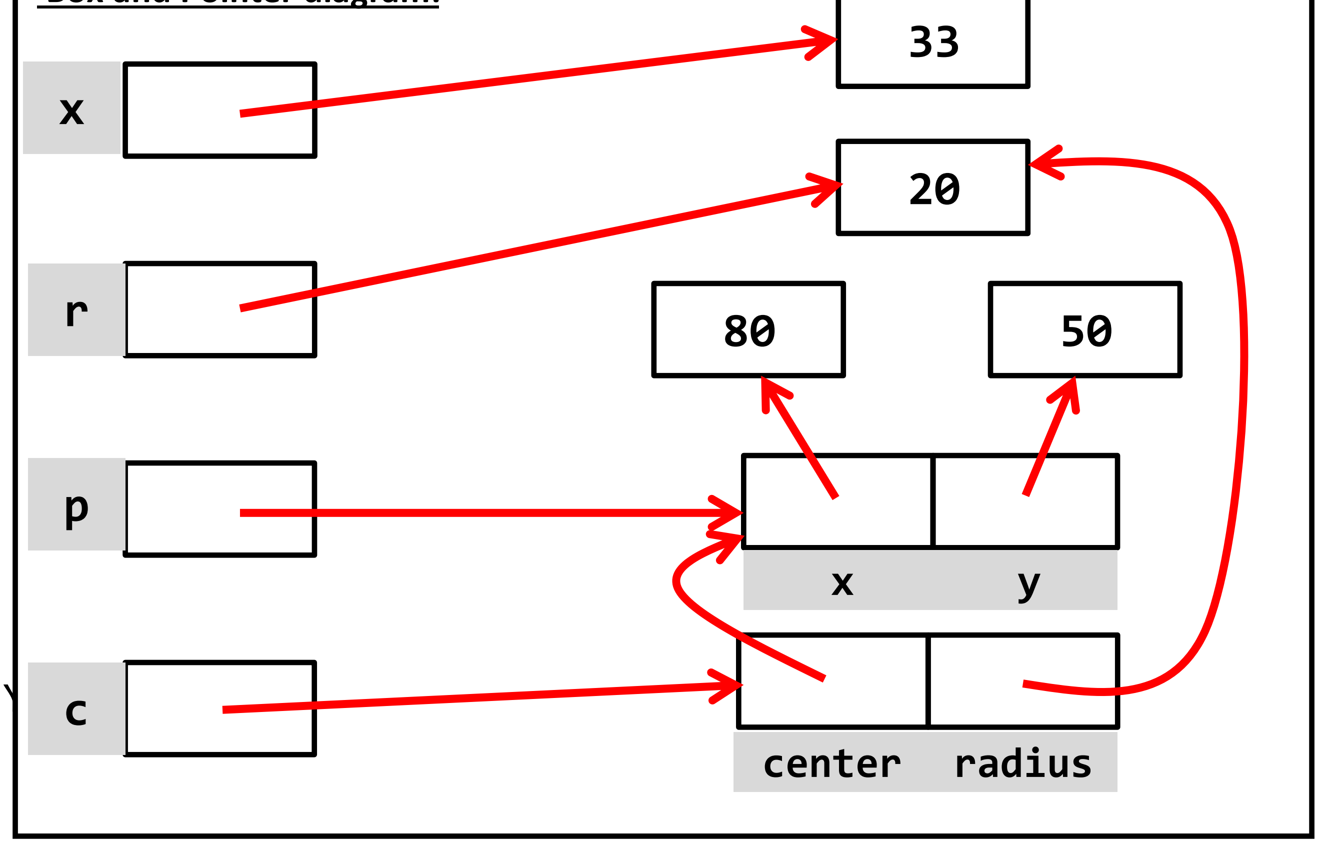
p.y: ____________

c.center.x: ________

c.center.y: ________

c.radius: ________

Box and Pointer diagram:



READ THIS page carefully, asking questions as needed!

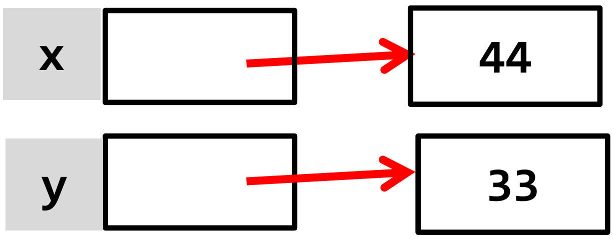
```
def main():
    x = 44
    y = 33

    foo(100, x)


def foo(a, b):
    ...
    x = 70

main()
```

Consider the code to the right. **A *function call* creates a new *namespace* in which the function will run.** Hence, when ***main*** is called, a namespace is created and then names (variables) ***x*** and ***y*** are created and assigned values. The box-and-pointer diagram after the assignments to ***x*** and ***y*** (but before the call to ***foo***) is:

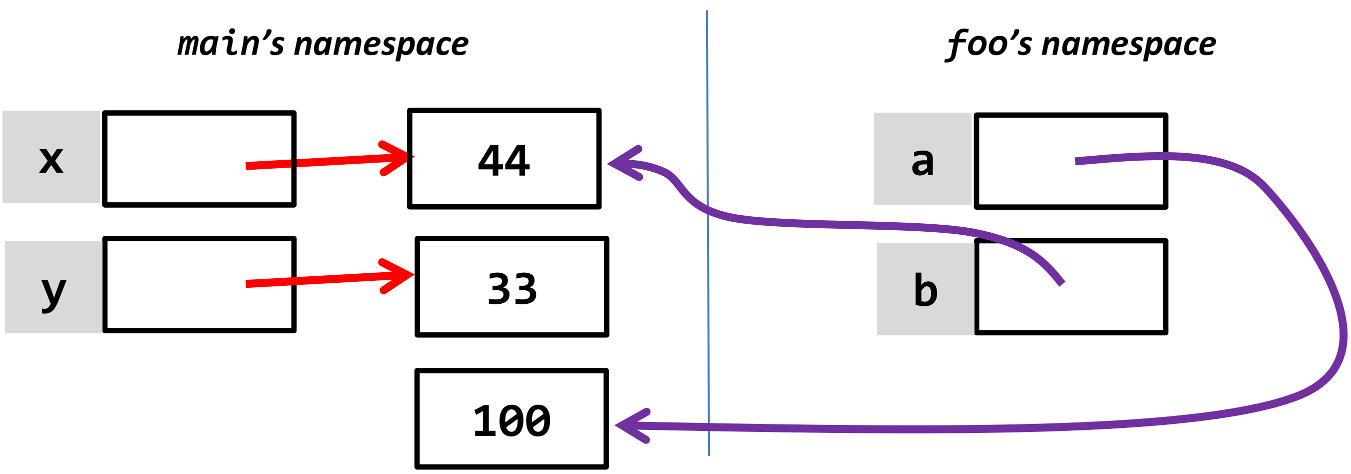


When a function is called, the function's parameters are added to the function's namespace. Each parameter is assigned the ***value*** of the corresponding actual argument. For example, when the call to function ***foo*** occurs in the code to the right, it is as if the following assignments occur:

`a` (in ***foo***) `=` `100` (in ***main***)

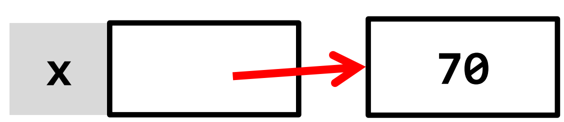
`b` (in ***foo***) `=` `x` (in ***main***) which is ***44***

So, after the call to ***foo***, the box and pointer has TWO parts (for the TWO namespaces), as shown below:



Note that the* variables *in foo point to* values *in main.

Also, note that the constant `100` appears in ***main***, so we have drawn it in ***main***'s namespace.



When the statement

`x = 70`

in ***foo*** runs, ***foo***'s namespace acquires its own variable ***x***, as shown to the right.

3. Draw a Box-and-Pointer diagram that shows what happens when ***main*** executes. Then indicate what output is printed, assuming appropriate ***print*** statements.

```
def main():
    a = 44
    b = 33
    z = 22
    p1 = Point(100, 200)

    foo(a, b, z, p1)

    <print statements here>

def foo(x, y, z, p):
    x = 10 * x
    y = 88
    p.x = 1
    p = Point(300, 400)
    p.y = 2
```

Output:

a: __________

b: __________

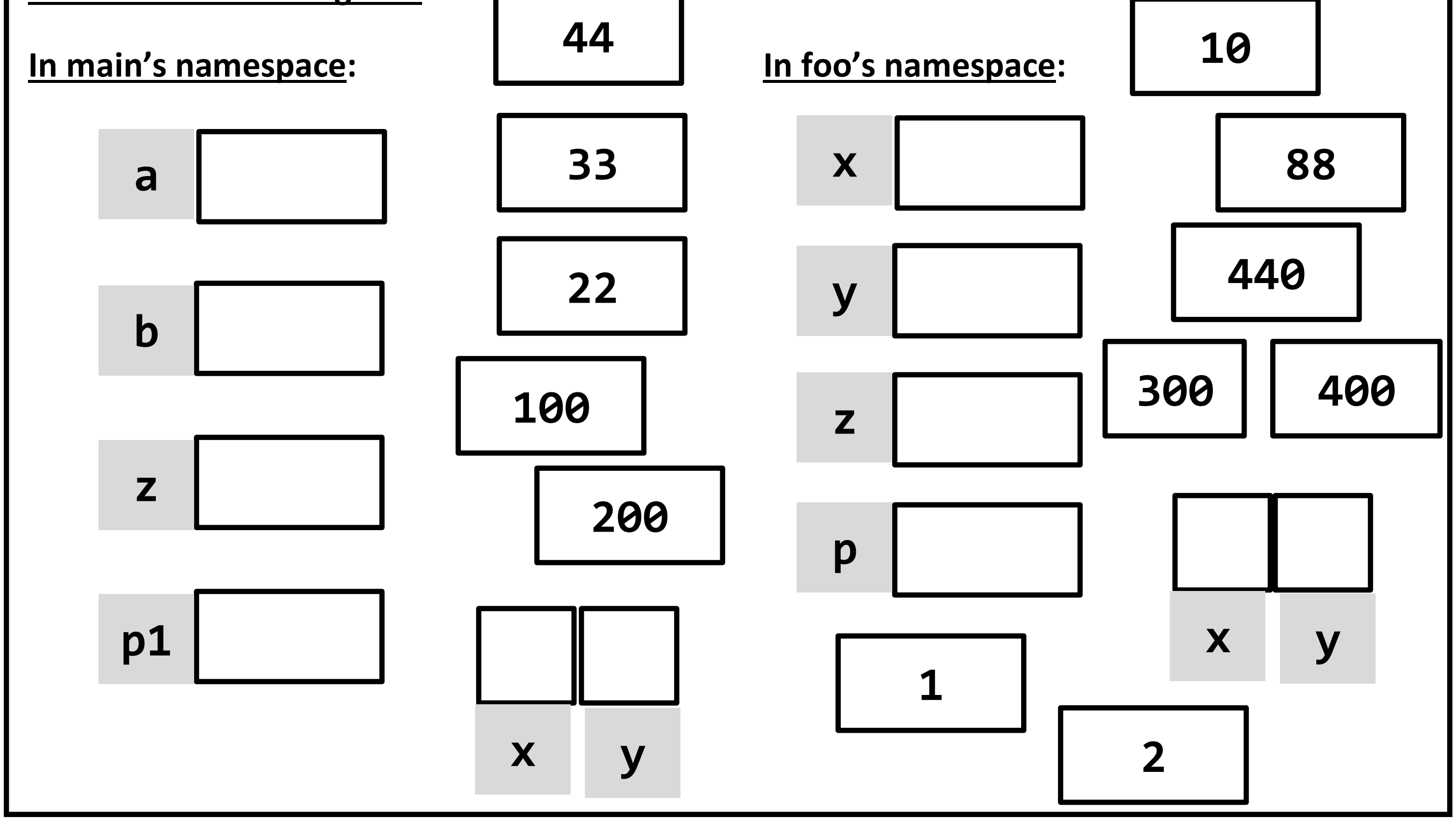
z: __________

p1.x: __________

p1.y: __________

We have already drawn all the boxes that you need. Just draw arrows (and eventually X's).

Box and Pointer diagram:



4. Draw a Box-and-Pointer diagram that shows what happens when ***main*** executes. Then indicate what output is printed, assuming appropriate ***print*** statements.

```
def main():
    a = 88
    b = 55
    p1 = Point(b, 66)
    p2 = Point(77, a)

    a = foo(p1, p2, a, b)

    <print statements here>


def foo(p, p1, a, b):
    p.y = 100
    p1.y = b
    p = Point(300, 400)
    p.x = 200
    b = 99

    <print statements here>

    return a + 10


main()
```

Output from printing in *foo*:

a: ___________

b: ___________

p.x: ____________

p.y: ____________

p1.x: ____________

p1.y: ____________

Output from printing in *main*:

a: ___________

b: ___________

p1.x: ____________

p1.y: ____________

p2.x: ____________

p2.y: ____________

Draw the entire box-and-pointer diagram on a *separate* sheet of paper, then staple that sheet to this handout.