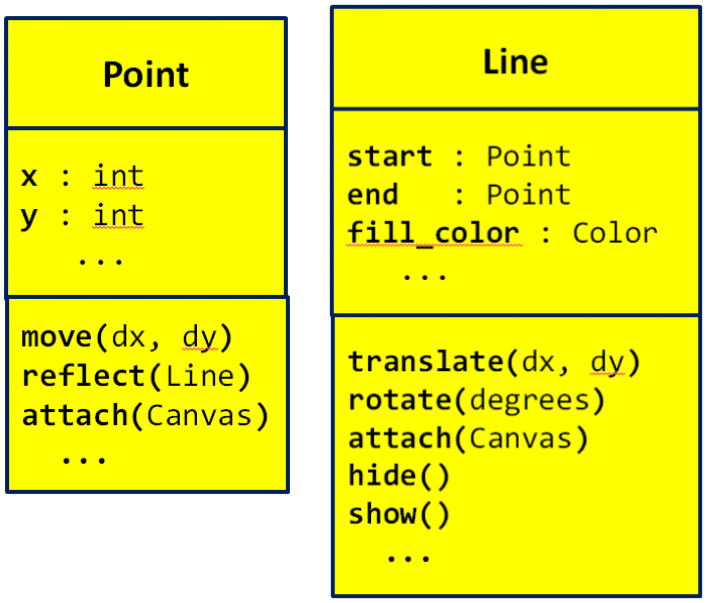
Name: _____

Use this quiz to help make sure you understand the videos/reading. ***Answer all questions.*** Make additional notes as desired. ***Not sure of an answer?*** Ask your instructor to explain in class and revise as needed then.

Throughout, where you are asked to "circle your choice", you can circle or underline it (whichever you prefer).

Video: **Classes – The Concepts**



1. The diagrams to the right are called

   _____ Class Diagrams, where

   _____ stands for Unified Modeling Language. (Fill in both blanks with the (same) 3-letter acronym for ***U***nified ***M***odeling ***L***anguage.)

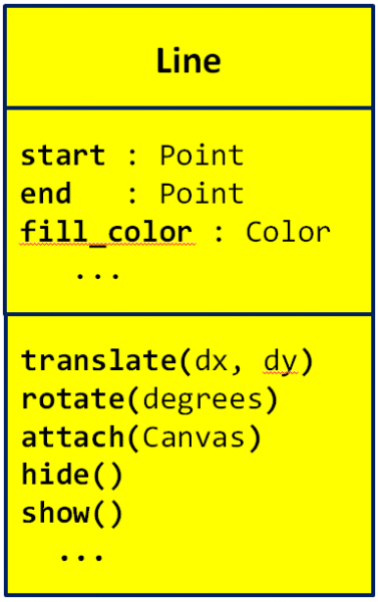
2. Consider the two UML class diagrams shown above and to the right. What are the names of the two ***classes*** shown?

3. Consider the UML class diagram for the ***`Point`*** class shown above. For that class:
   a. What are the names of the two ***instance variables*** (aka ***fields)*** that are shown?

   b. What do you think that those fields represent? (You can't tell this authoritatively from the UML class diagram; just make your best guess based on the names of the fields.)

   c. How many ***methods*** are shown?
   d. How many arguments does the ***`move`*** method require?
   e. How many arguments does the ***`reflect`*** method require?
   f. How many arguments does the ***`attach`*** method require?
   g. What kind of thing is the ***`reflect`*** method's argument?
   h. What kind of thing is the ***`attach`*** method's argument?

4. Consider the UML class diagram for the ***Line*** class shown on the previous page (and repeated to the right for your convenience). For that class:



   a. What are the names of the three ***instance variables*** (aka ***fields)*** that are shown?

   b. What do you think that those fields represent? (You can't tell this authoritatively from the UML class diagram; just make your best guess based on the names of the fields.)

   c. How many ***methods*** are shown?

   d. How many arguments does the ***translate*** method require?

   e. How many arguments does the ***hide*** method require?

5. True or false: A single class can have many instances of that class. **True** **False** (circle your choice)

6. True or false: For any particular class (say, the Point class), any instance of that class uses the same ***NAMES*** for its ***instance variables*** as any ***other*** instance of that class. **True** **False** (circle your choice)

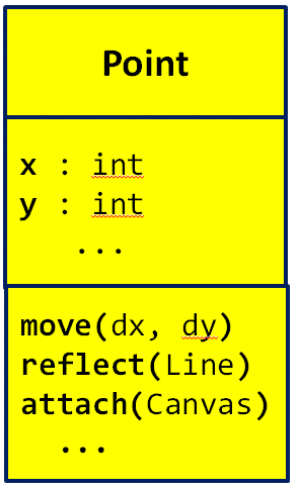
7. Consider two objects **`alpha`** and **`beta`**, both of which are instances of the same class. Suppose that both have an instance variable called **`time_of_day`**. True or false: The VALUE of **`alpha's time_of_day`** is necessarily the same as the VALUE of **`beta's time_of_day`** . **True** **False** (circle your choice)

8. True or false: For any particular class (say, the Point class), any instance of that class uses the same ***NAMES*** for its ***methods*** as any ***other*** instance of that class. **True** **False** (circle your choice)

9. Consider two objects **`alpha`** and **`beta`**, both of which are instances of the same class. Suppose that both have a method called **`sick_days_left()`** that returns a number. True or false: The value returned by a call to **`alpha's sick_days_left()`** is necessarily the same as the value returned by a call to **`beta's sick_days_left()`**. **True** **False** (circle your choice)

Video: **Classes – Notation**

In the following questions, we are referring to Point and Line classes per the UML class diagrams shown below. These classes are NOT part of RoseGraphics. That is why **you will not see (and should not put) any "rg dots".**



10. The following examples show two different attempts by two students to construct a new Point at (4, 3). Based on what you have learned about constructing objects in Python, which student(s) are correct and why? (Circle the ones that are CORRECT.)

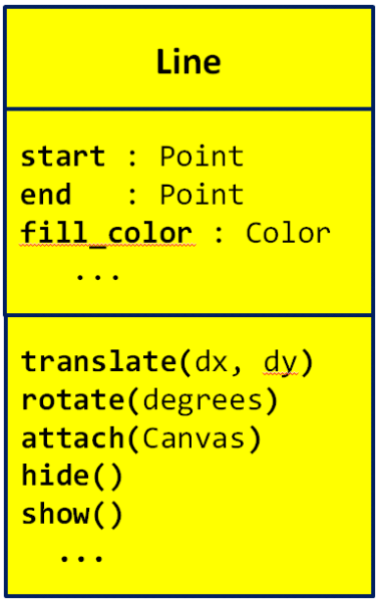
```
# Anna
point_anna = Point
point_anna.x = 4
point_anna.y = 3
```

```
# Dalton
point_dalton = Point(0,0)
point_dalton.x = 4
point_dalton.y = 3
```

11. A Point's constructor takes an initial x and a y. Thus, both of the above examples are needlessly long. How would you instead construct the point using just **one** line of code?

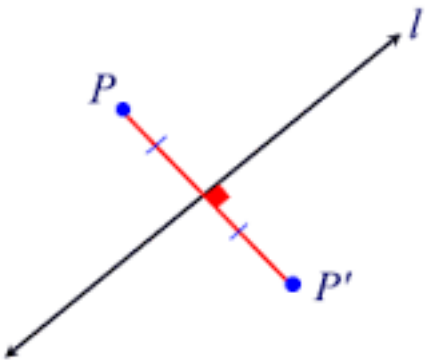
(continues on the next page)

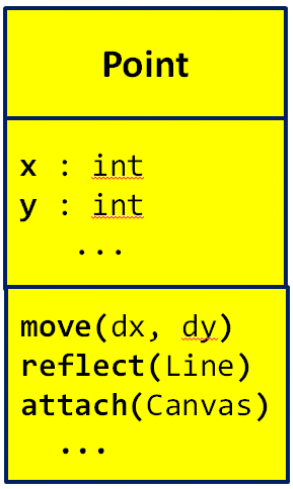
13. Suppose you are told that a Line's constructor takes a start and end point as its constructor arguments, as suggested by the UML class diagram to the right. What line of code would you write to create a Line between a point called ***point_mine*** and a ***new*** **Point** at (0, 5), putting the constructed object into a variable called ***line1***.



14. Assume that you have a Line object named ***line2*** and a Point object named ***another_point.*** Write the statement that would cause the Point class' **reflect** method to reflect ***another_point*** about ***line2*** to obtain the reflected point. For your convenience, the Point class is repeated to the right.

That is, if ***another_point*** is the point labelled ***P*** in the diagram to the right, and if ***line2*** is the line labelled ***l*** in the diagram, you are to write the statement that would obtain the point labelled **P'** in the diagram, by using the ***reflect*** method of the Point class.





a. First assume that the **reflect** method RETURNS a new Point that is the reflected Point. Write the statement that obtains the reflected Point under that assumption:

b. Now assume that the **reflect** method MUTATES its given Point to be reflected about the given Line. Write the statement that obtains the reflected Point under that assumption:

15. Suppose that you have two ***Line*** objects in variables ***line1*** and ***line2***, respectively. Write statements that set ***line1***'s fill color to ***'red'*** and ***line2***'s fill color to ***'blue'***.

Video: **Classes – Implementation**

16. Write the complete definition of a class called **Dog** that (so far) has nothing and does nothing.

17. What is the name of the special method of the **RoseWindow** class that is called by the following statement:

```
window = rg.RoseWindow()
```

18. Suppose that you have a class called **Dog** that has the following **__init__** method:

```
class Dog(object):

    def __init__(self, dog_name, age):
        self.name = dog_name
        self.age = age
```

When the following statements run, what will ***self.name*** and ***self.age*** be set to when **__init__** runs?

a. `fido = Dog('buster', 3)`

self.name is set to _____ **self.age** is set to _____

b. `fluffy = Dog('brownie', 5)`

self.name is set to _____ **self.age** is set to _____

19. Continuing the previous problem, what does each of the following PRINT statements print (or, if the statement has any error, just write ERROR), given the statements in the box below:

```
class Dog(object):

    def __init__(self, dog_name, age):
        self.name = dog_name
        self.age = age
```

```
fido = Dog('buster', 3)

fluffy = Dog('brownie', 5)
```

`print(fido.name)` prints _____

`print(fido.dog_name)` prints _____

`print(fido.age)` prints _____

`print(buster.age)` prints _____

`print(fluffy.name)` prints _____

`print(fluffy.dog_name)` prints _____

`print(fluffy.age)` prints _____

`print(brownie.age)` prints _____

20. Continuing the Dog class example, suppose we added a third line to **`__init__`**, as shown to the right.

```
class Dog(object):

    def __init__(self, dog_name, age):
        self.name = dog_name
        self.age = age
        self.breed = 'poodle'
```

a. Is that last line legal? If so, what is its effect?

b. If after constructing ***`fido`*** (per the previous problem), a statement included **`fido.breed,`** would that result in an error? If not, what would it evaluate to?

In the video you saw the definition of a **`Point`** class method called **`move_by`** that took two arguments and moved the **`Point`**'s coordinates (**`x`** and **`y`** instance variables) by those values. Keeping that example in mind:

21. Write the definition of a **`Point`** class method called **`move_to`** that takes two arguments and moves the Point TO (not BY) the given coordinates.

22. Write the definition of a **`Point`** class method called **`clone`** that takes NO arguments and returns a ***new*** **`Point`** whose **`x`** and **`y`** coordinates are the same as the Point's coordinates.