1 Indicate whether each of the following statements is `true` or `false`. (3)

a. The `super` keyword allows a class' methods to access attributes and methods of its super-class.

b. Subclasses can access the `private` attributes and methods of their their superclasses.

c. Subclass constructors must *explicitly* call a superclass constructor if the superclass does not contain a constructor which takes no parameters.

2 Suppose `Employee` contains the `calculatePay()` method and that `Manager` is a subclass of `Employee` which overrides `calculatePay()`. Consider the following code fragment. (3)

```
public void payIndividual(Employee emp) {
  double pay = emp.calculatePay();
  //  further implementation details not shown
}
```

If `manager1` is an instance of the `Manager` class, which copy of `calculatePay()` is evaluated when `payIndividual(manager1)` is called? Explain your reasoning.

3 Suppose the `Pencil` class is a subclass of `WritingUtensil` which adds the `erase()` method and that `pencil1` is an instance of the `Pencil` class. Consider the following code fragment. (5)

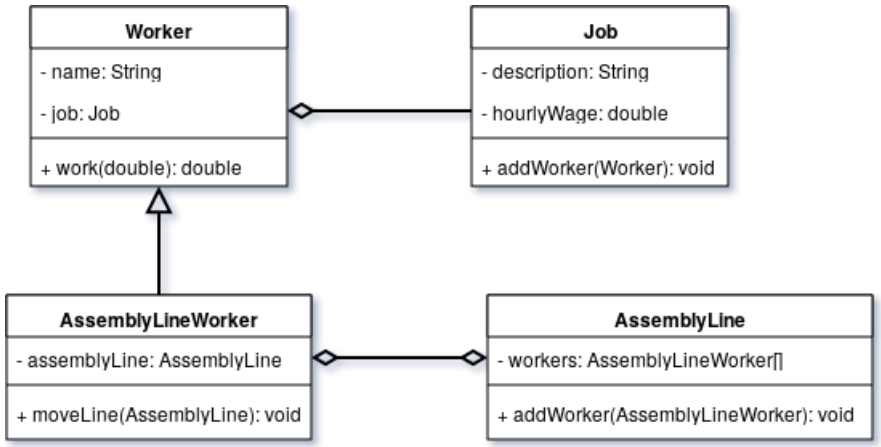
```
public void removeMistake(WritingUntensil unt) {
  unt.erase();
}
```

Explain why the method call, `removeMistake(pencil1)` causes an error. How would you fix this error?

4 For each pair of classes, indicate whether they should exhibit an "is-a" or "has-a" relationship. Write "neither" if neither relationship appears to work. (5)

1. `Cat`, `Animal`
2. `Zoo`, `Animal`
3. `Business`, `Employee`
4. `Cat`, `Dog`
5. `Manager`, `Employee`

5 Create classes representing an implementation of the following UML diagram. (10)
Note: Your methods do not need to represent actual working implementations for each class.



6 Use the `Point`, `Line`, and `Geometry` classes implemented in the previous assignment to complete each of the following tasks. (20)

(a) Create the `Polygon`, `Triangle`, and `Rectangle` classes with the following specifications.

`Polygon`

- Contains an attribute to hold a collection of points.
- Contains the appropriate constructor for accepting a collection of points.
- Contains the `calcPerimeter()` and `calcArea()` methods.
  Note: Due to the complexity of calculating areas of general polygons, the `calcArea()` method can return a default value of -1.

`Triangle`

- A `Triangle` "is-a" `Polygon`.
- Overrides the `Polygon` constructor in order to verify the correct number of points.
- Overrides the `calcArea()` method. You can use Heron's Formula below:
  Given a triangle with side lengths a, b, and c.

$$s = \frac{a+b+c}{2}$$
$$A = \sqrt{s(s-a)(s-b)(s-c)}$$

`Rectangle`

- A `Rectangle` "is-a" `Polygon`.
- Overrides the `Polygon` constructor in order to verify the correct number of points.
- Overrides the `calcArea()` method.

(b) Implement a `Square` class.
Note: The constructor of your `Square` class should verify the points are valid for a square.

(c) Draw a UML diagram of your entire system of classes.