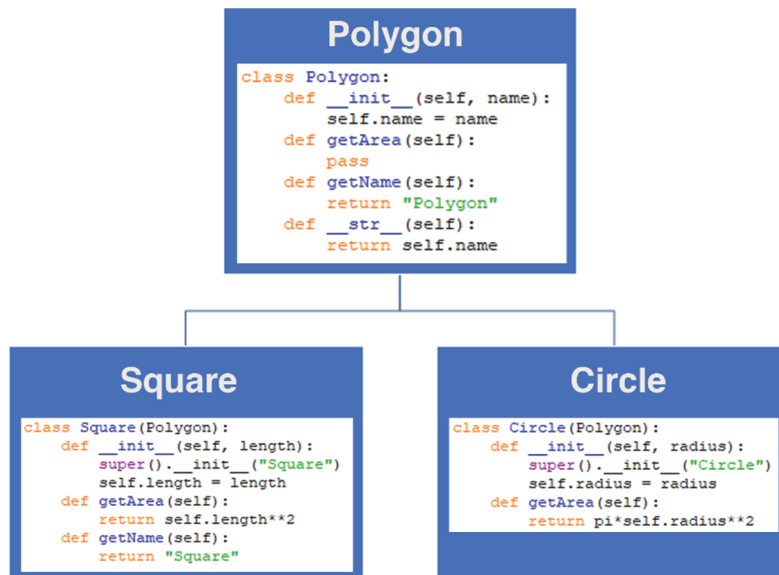


# Polymorphism

The child classes in Python inherits methods and attributes from the parent class. We can redefine certain methods and attributes specifically for the child class using a feature called **Method Overriding**.



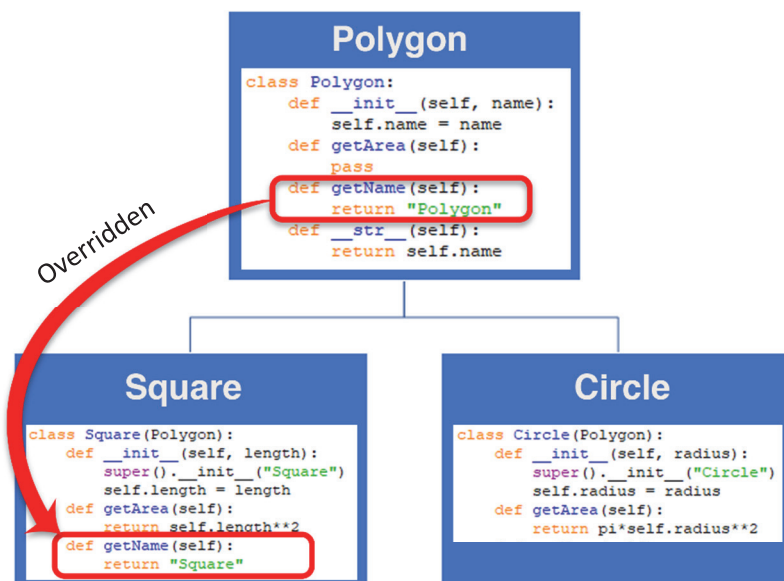
Polymorphism allows us to access these overridden methods and attributes that have the same name as defined in the parent class. For example, if we create an object called `squareObj` from the Square class:

```
squareObj = Square(7)
```

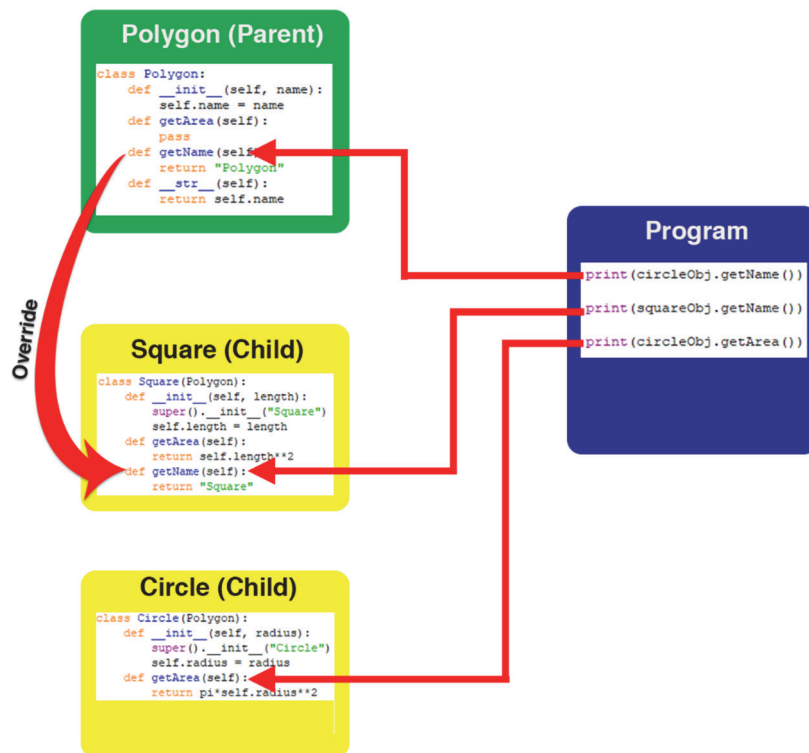
Then call the `getName()` method.

```
print(squareObj.getName())
```

The interpreter automatically recognizes that the `getName()` method for `squareObj` is overridden.



So, it uses the method defined in the Square class overriding the method defined in the parent. Similarly with `getArea()`.



You'll also notice that we have called the `getName()` method from the circle class. In this case, there is no method defined in the circle class, so the interpreter will call the `getName()` method from the parent (you'll need the `super()` function to enable this). Have a look at `polyclass2.py`