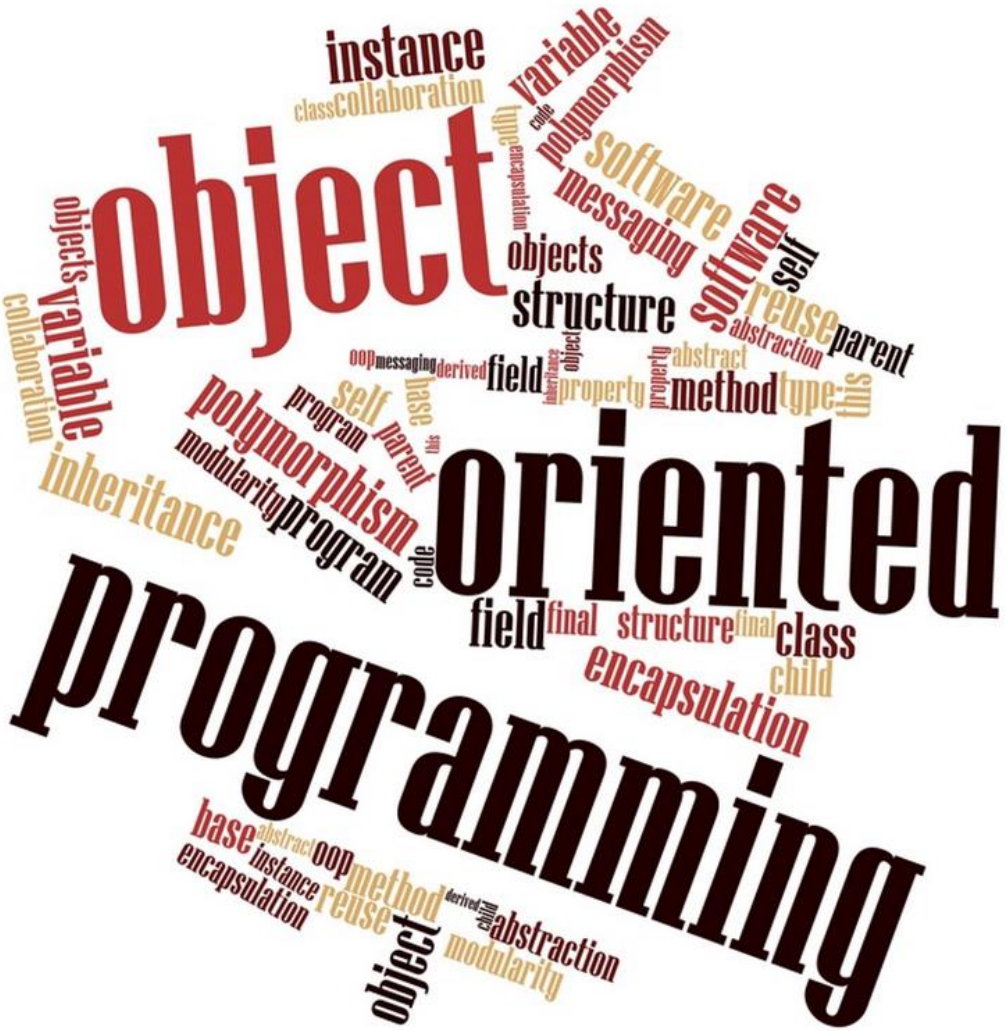
instance
variable
polymorphism
class
collaboration
code
encapsulation
object
software
messaging
objects
structure
software
self
reuse
abstraction
parent
objects
variable
collaboration
oop
messaging
derived
field
inheritance
object
property
abstract
method
type
this
self
base
program
parent
this
polymorphism
modularity
program
inheritance
code
oriented
field
final
structure
final
class
child
encapsulation
programming
base
abstract
oop
instance
encapsulation
method
reuse
derived
child
abstraction
object
modularity

METHOD

```
class Classy:
      def method(self):
            print("method")

obj = Classy()
obj.method()
```

```
class Classy:
      def method(self, par):
            print("method:", par)

obj = Classy()
obj.method(1)
```

METHOD

```
class Classy:
      Variable = 2
      def method(self):
            print(self.Variable, self.var)


obj = Classy()
obj.var = 3
obj.method()
```

```
class Classy:
      def other(self):
            print("other")
      def method(self):
            print("method")
            self.other()


obj = Classy()
obj.method()
```

METHODS IN DETAIL

```
class Classy:
        def __init__(self,value):
                self.var = value

obj1 = Classy("object")
print(obj1.var)
```

```
class Classy:
        def __init__(self,value = None):
                self.var = value

obj1 = Classy("object")
obj2 = Classy()
print(obj1.var)
print(obj2.var)
```

METHODS IN DETAIL

```
class Classy:
        def visible(self):
                print("visible")
        def __hidden(self):
                print("hidden")

obj = Classy()
obj.visible()
try:
        obj.__hidden()
except:
        print("failed")
obj._Classy__hidden()
```

INHERITANCE

Ejercicio 7 – Método str

```
class Star:
        def __init__(self,name,galaxy):
                self.name = name
                self.galaxy = galaxy

sun = Star("Sun","Milky Way")
print(sun)
```

<__main__.Star object at 0x7f1074cc7c50>

```
class Star:
        def __init__(self,name,galaxy):
                self.name = name
                self.galaxy = galaxy

        def __str__(self):
                return self.name + ' in ' + self.galaxy

sun = Star("Sun","Milky Way")
print(sun)
```

INHERITANCE

```
class Vehicle:
      pass

class LandVehicle(Vehicle):
      pass

class TrackedVehicle(LandVehicle):
      pass
```

```
issubclass(class1,class2)
```

INHERITANCE

```
isinstance(object,class)
```

```
class Vehicle:
      pass

class LandVehicle(Vehicle):
      pass

class TrackedVehicle(LandVehicle):
      pass

vehicle = Vehicle()
landvehicle = LandVehicle()
trackedvehicle = TrackedVehicle()
```

INHERITANCE

Ejercicio 8 – Método str

```
class Super:
        def __init__(self,name):
                self.name = name
        def __str__(self):
                return "My name is " + self.name + "."

class Sub(Super):
        def __init__(self,name):
                Super.__init__(self,name)

object = Sub("Andy")
print(object)
```

INHERITANCE

Ejercicio 9 – Función super

```
class Super:
        def __init__(self,name):
                self.name = name
        def __str__(self):
                return "My name is " + self.name + "."

class Sub(Super):
         def __init__(self,name):
                super().__init__(name)

object = Sub("Andy")
print(object)
```

INHERITANCE

```
class SuperA:
   VarA = 10
   def funa(self):
        return 11

class SuperB:
   VarB = 20
   def funb(self):
        return 21

class Sub(SuperA,SuperB):
   pass

object = Sub()

print(object.VarA, object.funa())
print(object.VarB, object.funb())
```