Inheritance

- Child class inherits the methods and attributes of the parent class
- The child class can extend the init method of the parent class with super() function:

In the example below the Developer class EXTENDS the parent class Employee.
init method of the Developer class calls the init method in the parent class to instantiate the common variables between Employee and Developer class which are :
->name
->id
->pay

The variables which belong to ONLY class developer which are:
->dept
->prog_lang
are initialized later.

```
class Employee:
  raise_percent = 1.5
  def __init__(self,name,id,pay):
    self.name = name
    self.id = id
    self.pay = pay

  def apply_raise(self):
    self.raise_amt = (self.raise_percent* self.pay)/100
    self.pay =  int(self.pay + self.raise_amt)
    return self.pay


class Developer(Employee):  #inherits Employee class
  raise_percent = 1.2       #class variable overloaded
  def __init__(self,name,id,pay,dept,prog_lang):
    super().__init__(name,id,pay)  #calling init from parent class
    self.dept = dept
    self.prog_lang = prog_lang
```

Another class added that inherits the Employee class with its own set of methods:

```python
class Employee:
  raise_percent = 1.5
  def __init__(self,name,id,pay):
    self.name = name
    self.id = id
    self.pay = pay

  def apply_raise(self):
    self.raise_amt = (self.raise_percent* self.pay)/100
    self.pay =  int(self.pay + self.raise_amt)
    return self.pay


class Developer(Employee):  #inherits Employee class
  raise_percent = 1.2       #class variable overloaded
  def __init__(self,name,id,pay,dept,prog_lang):
    super().__init__(name,id,pay)  #calling init from parent class
    self.dept = dept
    self.prog_lang = prog_lang


class Manager(Employee):
  def __init__(self,name,id,pay,dept,reportees=None):
    super().__init__(name,id,pay)  #calling init from parent class
    self.dept = dept
    self.reportees = reportees

  def add_new_reportee(self,reporteelist):
    if reporteelist is None:
      return "Nothing to add"
    else:
      for reportee in reporteelist:
        self.reportees.append(reportee)

  def get_reportee_list(self):
    return self.reportees

#create instance of Manager class
mgr1 = Manager('Ben',100,100,'Retail',['a','b','c'])
mgr1.add_new_reportee(['lisa','kyle'])
print("Reportee list for manager: ", mgr1.get_reportee_list())

print("\n\n")
#create instance of Developer class
dev1 = Developer('Mohima',100,100,'Retail','shell')
print("Developer's pay: ", dev1.pay)
print("Developer's pay after raise", dev1.apply_raise())
```

- Testing object class relationship-> is objectA an instance of Class B? -> isinstance()
- Testing parent child relationship between classes -> is class A child of subclass B? -> issubclass

```
class Employee:
  raise_percent = 1.5
  def __init__(self,name,id,pay):
    self.name = name
    self.id = id
    self.pay = pay


class Developer(Employee):  #inherits Employee class
  raise_percent = 1.2       #class variable overloaded
  def __init__(self,name,id,pay,dept,prog_lang):
    super().__init__(name,id,pay)  #calling init from parent class
    self.dept = dept
    self.prog_lang = prog_lang


class Manager(Employee):
  def __init__(self,name,id,pay,dept,reportees=None):
    super().__init__(name,id,pay)  #calling init from parent class
    self.dept = dept
    self.reportees = reportees


#create instance of Manager class
mgr1 = Manager('Ben',100,100,'Retail',['a','b','c'])

#create instance of Developer class
dev1 = Developer('Mohima',100,100,'Retail','shell')

#isinstance
print(isinstance(mgr1,Manager))
print(isinstance(mgr1,Employee))
print(isinstance(mgr1,Developer))

print("*"*20,"\n")

print(issubclass(Manager,Employee))
print(issubclass(Manager,Developer))
print(issubclass(Developer,Manager))

OUTPUT
True
True
False
********************

True
False
False
```