**Case 1:**  **Additional Builtin Functions - Example**

class Person:

def \_\_init\_\_(self, name, age):

self.name = name

self.age = age

def myfunc(self):

print("Hello my name is " + self.name)

p1 = Person("John", 36)

p1.myfunc()

setattr(p1,'name','Felix')

print(hasattr(p1, 'name'))

print(getattr(p1,'age'))

p1.myfunc()

delattr(p1,'age')

print(hasattr(p1,'name'))

print(hasattr(p1,'age'))

………………………..

**Case 2: Deletion of the object- Example**

class Person:

def \_\_init\_\_(mysillyobject, name, age):

mysillyobject.name = name

mysillyobject.age = age

def myfunc(abc):

print("Hello my name is " + abc.name, "My age is: ", + abc.age)

p1 = Person("John", 36)

p1.age= 40

p1.myfunc()

del p1

p1.myfunc()

**case 3: Destructor in Python**

class student():

def \_\_init\_\_(self,fn,ln):

self.fname=fn

self.lname=ln

def display(self):

print(self.fname,self.lname)

def \_\_del\_\_(self):

print("destructor called")

obj = student("abc","xyz")

obj.display()

……………………………………

**Case 4 Checking Class Relationships**

class Person:

def \_\_init\_\_(self, fname, lname):

self.firstname = fname

self.lastname = lname

def printname(self):

print(self.firstname, self.lastname)

class Student(Person):

def \_\_init\_\_(self, fname, lname, year):

super().\_\_init\_\_(fname, lname)

self.graduationyear = year

x = Student("john", "Grisham", 2023)

print(issubclass(Student,Person))

print(isinstance(x,Student))

print(isinstance(x,Person))

**case 5: Method overriding Example 1:**

class Parent():

def \_\_init\_\_(self):

self.value = "Inside Parent"

def show(self):

print(self.value)

class Child(Parent):

def \_\_init\_\_(self):

self.value = "Inside Child"

def show(self):

print(self.value)

obj1 = Parent()

obj2 = Child()

obj1.show()

obj2.show()

**case 6: Multiple Inheritance Overriding Example 2**

class Parent1():

def show(self):

print("Inside Parent1")

class Parent2():

def display(self):

print("Inside Parent2")

class Child(Parent1, Parent2):

def show(self):

print("Inside Child")

obj = Child()

obj.show()

obj.display()

**case 7: Multiple Inheritance Overriding Example 3**

class Parent1():

def show(self):

print("Inside Parent1")

class Parent2():

def show(self):

print("Inside Parent2")

class Child(Parent2, Parent1):

pass

obj = Child()

obj.show()

**case 8: Multilevel Inheritance Overriding - Example**

class Parent():

def display(self):

print("Inside Parent")

class Child(Parent):

def show(self):

print("Inside Child")

class GrandChild(Child):

def show(self):

print("Inside GrandChild")

g = GrandChild()

g.show()

g.display()

**case 9: Calling Parent’s method within overridden method**

class Parent():

def show(self):

print("Inside Parent")

class Child(Parent):

def show(self):

Parent.show(self)

print("Inside Child")

obj = Child()

obj.show()

**case 10: Calling Parent’s method within overridden method using super()**

class Parent():

def show(self):

print("Inside Parent")

class Child(Parent):

def show(self):

super().show()

print("Inside Child")

obj = Child()

obj.show()