**Lab Exercise 2- Inheritance with Python**

Here's a more complex lab exercise that involves implementing inheritance in Python with an example related to employee departments and roles:

**Step 1: Create a Parent Class**

class Employee:

def \_\_init\_\_(self, name, employee\_id):

self.name = name

self.employee\_id = employee\_id

def display\_info(self):

print(f"Name: {self.name}, Employee ID: {self.employee\_id}")

**Step 2: Create a Child Class that Inherits from the Parent Class**

class Department(Employee):

def \_\_init\_\_(self, name, employee\_id, department\_name):

super().\_\_init\_\_(name, employee\_id)

self.department\_name = department\_name

def display\_department\_info(self):

self.display\_info()

print(f"Department: {self.department\_name}")

**Step 3: Create Another Child Class**

class Manager(Department):

def \_\_init\_\_(self, name, employee\_id, department\_name, role):

super().\_\_init\_\_(name, employee\_id, department\_name)

self.role = role

def display\_manager\_info(self):

self.display\_department\_info()

print(f"Role: {self.role}")

**Step 4: Create Objects and Test the Inheritance**

# Create objects and test inheritance

employee1 = Employee("John Doe", 1001)

employee1.display\_info()

employee2 = Department("Jane Smith", 1002, "Human Resources")

employee2.display\_department\_info()

manager1 = Manager("Bob Johnson", 1003, "Finance", "Senior Manager")

manager1.display\_manager\_info()

**Conclusion**

This exercise demonstrated a more complex example of inheritance in Python related to employee departments and roles. The Employee class serves as the parent class, the Department class serves as the child class inheriting from the Employee class, and the Manager class serves as the child class inheriting from the Department class. Experiment further with inheritance and explore more complex class structures to deepen your understanding of inheritance in Python.