**Assignment 2- Inheritance in Python - Bank System**

**Task 1: Create a Parent Class**

The **Account class** will serve as the parent class, representing a generic bank account. It will have the following attributes and methods:

**Attributes:**

* account\_number: representing the account number
* balance: representing the current balance in the account

**Methods:**

* display\_info(): to display the account number and balance

**Task 2: Create a Child Class - SavingsAccount**

The SavingsAccount class will inherit from the Account class. It will represent a savings account with an additional attribute for the interest rate.

**Attributes:**

* Inherits attributes from the Account class
* interest\_rate: representing the interest rate for the savings account

**Methods:**

* add\_interest(): to add interest to the account based on the interest rate
* Override the display\_info() method to include the interest rate in the displayed information

**Task 3: Create Another Child Class - CheckingAccount**

The CheckingAccount class will also inherit from the Account class. It will represent a checking account with an additional attribute for the transaction limit.

**Attributes:**

* Inherits attributes from the Account class
* transaction\_limit: representing the maximum number of transactions allowed

**Methods:**

* process\_transaction(amount): to process a transaction (withdrawal) from the account and update the transaction limit
* Override the display\_info() method to include the transaction limit in the displayed information

**Task 4: Test the Inheritance**

Test the inheritance by creating instances of the SavingsAccount and CheckingAccount classes and calling the relevant methods to demonstrate the functionality of each class.

**Conclusion**

This assignment focuses on implementing inheritance in Python to create a simple bank system with savings and checking accounts. The inheritance structure helps to efficiently manage common attributes and methods shared between different types of accounts. Experiment further with the code and explore more advanced functionalities to deepen your understanding of inheritance in Python.

**Solution**:

class Account:  
 def \_\_init\_\_(self, account\_number, balance):  
 self.account\_number = account\_number  
 self.balance = balance  
  
 def display\_info(self):  
 print(f"Account Number: {self.account\_number}, Balance: {self.balance}")  
  
  
class SavingsAccount(Account):  
 def \_\_init\_\_(self, account\_number, balance, interest\_rate):  
 super().\_\_init\_\_(account\_number, balance)  
 self.interest\_rate = interest\_rate  
  
 def add\_interest(self):  
 self.balance += ((self.interest\_rate/100) \* self.balance)  
 print(f"New Balance: {self.balance}")  
  
 def display\_interest\_rate\_info(self):  
 self.display\_info()  
 print(f"Interest Rate: {self.interest\_rate}")  
  
  
class CheckingAccount(Account):  
 def \_\_init\_\_(self, account\_number, balance, transaction\_limit):  
 super().\_\_init\_\_(account\_number, balance)  
 self.transaction\_limit = transaction\_limit  
  
 def process\_transaction(self, amount):  
 self.transaction\_limit -= amount  
  
 def display\_tx\_limit\_info(self):  
 self.display\_info()  
 print(f"Transaction Limit: {self.transaction\_limit}")  
  
# Create objects and test inheritance  
pcls = Account(123456, 10000)  
pcls.display\_info()  
  
ccls1 = SavingsAccount(123456, 10000, 5)  
ccls1.add\_interest()  
ccls1.display\_interest\_rate\_info()  
  
ccls2 = CheckingAccount(123456, 10000, 50000)  
ccls2.process\_transaction(2000)  
ccls2.display\_tx\_limit\_info()