2124UCEM1041\_Gaurav\_Bodkhe

Unit 4 - Lab Work

**Problem Statement=** Implement a class “BankAccount” with methods to deposit money, withdraw money, and check balance. Implement inheritance to create classes “SavingsAccount ” and “CurrentAccount”. Use polymorphism to override methods for different account types.

**CODE=**

# Base class for BankAccount

class BankAccount:

def \_\_init\_\_(self, account\_holder, balance=0):

self.account\_holder = account\_holder

self.balance = balance

def deposit(self, amount):

"""Deposit money into the account"""

if amount > 0:

self.balance += amount

print(f"Deposited ${amount}. Current balance: ${self.balance}")

else:

print("Deposit amount must be positive.")

def withdraw(self, amount):

"""Withdraw money from the account"""

if amount > 0 and amount <= self.balance:

self.balance -= amount

print(f"Withdrew ${amount}. Current balance: ${self.balance}")

elif amount <= 0:

print("Withdrawal amount must be positive.")

else:

print("Insufficient funds.")

def check\_balance(self):

"""Check the current balance"""

print(f"Current balance: ${self.balance}")

# Subclass for SavingsAccount (inherits from BankAccount)

class SavingsAccount(BankAccount):

def \_\_init\_\_(self, account\_holder, balance=0, interest\_rate=0.02):

super().\_\_init\_\_(account\_holder, balance)

self.interest\_rate = interest\_rate

def deposit(self, amount):

"""Override deposit method for savings account (apply interest on deposit)"""

if amount > 0:

self.balance += amount + (amount \* self.interest\_rate)

print(f"Deposited ${amount}. Interest applied. Current balance: ${self.balance}")

else:

print("Deposit amount must be positive.")

def withdraw(self, amount):

"""Override withdraw method for savings account"""

if amount > 0 and amount <= self.balance:

self.balance -= amount

print(f"Withdrew ${amount}. Current balance: ${self.balance}")

elif amount <= 0:

print("Withdrawal amount must be positive.")

else:

print("Insufficient funds.")

# Subclass for CurrentAccount (inherits from BankAccount)

class CurrentAccount(BankAccount):

def \_\_init\_\_(self, account\_holder, balance=0, overdraft\_limit=500):

super().\_\_init\_\_(account\_holder, balance)

self.overdraft\_limit = overdraft\_limit

def deposit(self, amount):

"""Override deposit method for current account"""

if amount > 0:

self.balance += amount

print(f"Deposited ${amount}. Current balance: ${self.balance}")

else:

print("Deposit amount must be positive.")

def withdraw(self, amount):

"""Override withdraw method for current account (allowing overdraft)"""

if amount > 0 and (self.balance - amount >= -self.overdraft\_limit):

self.balance -= amount

print(f"Withdrew ${amount}. Current balance: ${self.balance}")

elif amount <= 0:

print("Withdrawal amount must be positive.")

else:

print("Exceeded overdraft limit.")

# Example usage

savings = SavingsAccount("Alice", 1000, 0.03)

current = CurrentAccount("Bob", 500, 200)

# Performing operations on SavingsAccount

print("\nSavings Account Operations:")

savings.deposit(500)

savings.withdraw(200)

savings.check\_balance()

# Performing operations on CurrentAccount

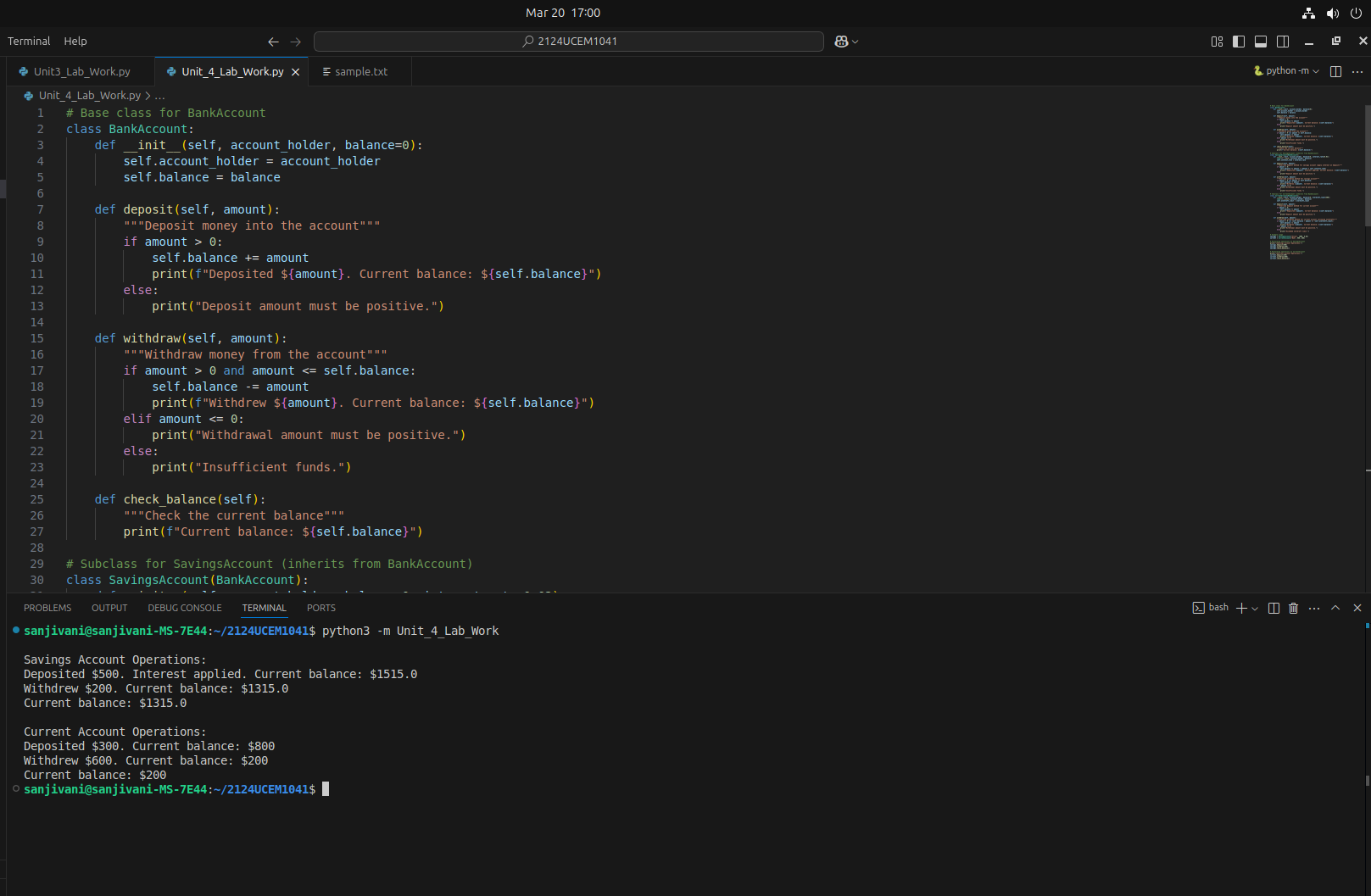
print("\nCurrent Account Operations:")

current.deposit(300)

current.withdraw(600)

current.check\_balance()

**O/P=**

****