Assignment 4

|  |  |  |
| --- | --- | --- |
| **4.1 Class** | | |
| **Aim:** Create a student class with attributes for name, age, and grades, and methods to calculate the average grade and display student information.  Create a BankAccount class with attributes for account number, balance, and account type, and methods to deposit, withdraw, and display account information.  Create a Person superclass with attributes for name and age, and a Student subclass that inherits from Person and adds an attribute for student ID and a method to display student information. | | |
| **Code:**  class BasicStudent:  def \_\_init\_\_(self, name, age, grades):  self.name = name  self.age = age  self.grades = grades *# list of numeric grades*  def average\_grade(self):  if not self.grades:  return 0  return sum(self.grades) / len(self.grades)  def display\_info(self):  print(f"Student Name: {self.name}")  print(f"Age: {self.age}")  print(f"Grades: {self.grades}")  print(f"Average Grade: {self.average\_grade():.2f}")  class BankAccount:  def \_\_init\_\_(self, account\_number, balance, account\_type):  self.account\_number = account\_number  self.balance = balance  self.account\_type = account\_type  def deposit(self, amount):  if amount > 0:  self.balance += amount  print(f"Deposited ${amount:.2f}. New balance: ${self.balance:.2f}")  else:  print("Deposit amount must be positive.")  def withdraw(self, amount):  if amount <= 0:  print("Withdrawal amount must be positive.")  elif amount > self.balance:  print("Insufficient funds.")  else:  self.balance -= amount  print(f"Withdrew ${amount:.2f}. New balance: ${self.balance:.2f}")  def display\_info(self):  print(f"Account Number: {self.account\_number}")  print(f"Account Type: {self.account\_type}")  print(f"Balance: ${self.balance:.2f}")  class Person:  def \_\_init\_\_(self, name, age):  self.name = name  self.age = age  class Student(Person):  def \_\_init\_\_(self, name, age, student\_id):  super().\_\_init\_\_(name, age)  self.student\_id = student\_id  def display\_info(self):  print(f"Student ID: {self.student\_id}")  print(f"Name: {self.name}")  print(f"Age: {self.age}")  if \_\_name\_\_ == "\_\_main\_\_":  print("---- Testing BasicStudent ----")  student1 = BasicStudent("Alice", 20, [88, 92, 79])  student1.display\_info()  print("\n---- Testing BankAccount ----")  account1 = BankAccount("123456789", 500.0, "Checking")  account1.display\_info()  account1.deposit(100)  account1.withdraw(50)  print("\n---- Testing Student (inherits from Person) ----")  student2 = Student("Bob", 21, "S12345")  student2.display\_info()  **Output Screenshot:** | | |
| **Conclusion/Summary:**  This assignment demonstrates the implementation and usage of object-oriented programming in Python by creating multiple classes with distinct responsibilities. The code includes:  A BasicStudent class that calculates and displays a student's average grade.  A BankAccount class that manages account transactions such as deposits and withdrawals, displaying the account details.  A Person superclass and a Student subclass to illustrate inheritance, where the Student class extends functionality by including a student ID attribute.  Together, these classes showcase principles such as encapsulation and inheritance, laying a solid foundation for more complex class interactions in future projects. | | |
| **Student Signature & Date** | **Marks:** | **Evaluator Signature & Date** |