Kỹ thuật lập trình Python - IE221.Q11

LAB04

Họ và tên: Phan Anh Kiệt

MSSV: 22520723

DEMO:

Bài 1:

[12]

```python
# DEMO 1
class Book:
    def __init__(self, title, author, isbn, price):
        self.book_title = title
        self.book_author = author
        self.book_isbn = isbn
        self.__book_price = price
        self.is_available = True

    def display_info(self):
        status = "Còn sẵn" if self.is_available else "Đã mượn"
        print(f"{self.book_title} - {self.book_author} | "
              f"ISBN: {self.book_isbn} | "
              f"Giá: {self.__book_price} | "
              f"Tình trạng: {status}")

    def borrow(self):
        if self.is_available:
            self.is_available = False
            print(f"Bạn đã mượn sách: {self.book_title}")
        else:
            print(f"Sách '{self.book_title}' hiện đang được mượn.")

    def return_book(self):
        if not self.is_available:
            self.is_available = True
            print(f"Bạn đã trả sách: {self.book_title}")
        else:
            print(f"Sách '{self.book_title}' chưa được mượn, không thể trả.")


class Library:
    def __init__(self, library_name):
        self.library_name = library_name
        self._book_list = []

    def add_book(self, book):
        self._book_list.append(book)
        print(f"Đã thêm sách vào thư viện: {book.book_title}")

    def remove_book(self, isbn):
        for book in self._book_list:
            if book.book_isbn == isbn:
                self._book_list.remove(book)
                print(f"Đã xóa sách: {book.book_title}")
                return
        print("Không tìm thấy sách có mã ISBN tương ứng.")

    def find_book(self, keyword):
        print(f"\nKết quả tìm kiếm cho từ khóa: '{keyword}'")
        found = False
        for book in self._book_list:
            if (keyword.lower() in book.book_title.lower() or
                    keyword.lower() in book.book_author.lower()):
                book.display_info()
                found = True
```

```
        def find_book(self, keyword):
            print(f"\nKết quả tìm kiếm cho từ khóa: '{keyword}'")
            found = False
            for book in self._book_list:
                if (keyword.lower() in book.book_title.lower() or
                        keyword.lower() in book.book_author.lower()):
                    book.display_info()
                    found = True
            if not found:
                print("Không tìm thấy sách phù hợp với từ khóa.")

        def display_all_books(self):
            print(f"\nDanh sách sách trong thư viện '{self.library_name}':")
            if not self._book_list:
                print("Hiện tại thư viện chưa có sách.")
            for book in self._book_list:
                book.display_info()


    if __name__ == "__main__":
        uit_library = Library("UIT Library")

        python_book = Book("Python 101: Lập trình cơ bản", "Nguyễn Minh Tuấn", "ISBN001", 180000)
        ml_book = Book("Machine Learning Cơ Bản", "Phạm Thị Lan", "ISBN002", 220000)
        ai_book = Book("Trí tuệ Nhân tạo Ứng dụng", "Lê Văn An", "ISBN003", 270000)

        uit_library.add_book(python_book)
        uit_library.add_book(ml_book)
        uit_library.add_book(ai_book)

        uit_library.display_all_books()

        python_book.borrow()
        python_book.borrow()
        python_book.return_book()

        uit_library.find_book("Python")

        uit_library.remove_book("ISBN002")

        uit_library.display_all_books()
```

```
Đã thêm sách vào thư viện: Python 101: Lập trình cơ bản
Đã thêm sách vào thư viện: Machine Learning Cơ Bản
Đã thêm sách vào thư viện: Trí tuệ Nhân tạo Ứng dụng

Danh sách sách trong thư viện 'UIT Library':
Python 101: Lập trình cơ bản - Nguyễn Minh Tuấn | ISBN: ISBN001 | Giá: 180000 | Tình trạng: Còn sẵn
Machine Learning Cơ Bản - Phạm Thị Lan | ISBN: ISBN002 | Giá: 220000 | Tình trạng: Còn sẵn
Trí tuệ Nhân tạo Ứng dụng - Lê Văn An | ISBN: ISBN003 | Giá: 270000 | Tình trạng: Còn sẵn
Bạn đã mượn sách: Python 101: Lập trình cơ bản
Sách 'Python 101: Lập trình cơ bản' hiện đang được mượn.
Bạn đã trả sách: Python 101: Lập trình cơ bản

Kết quả tìm kiếm cho từ khóa: 'Python'
Python 101: Lập trình cơ bản - Nguyễn Minh Tuấn | ISBN: ISBN001 | Giá: 180000 | Tình trạng: Còn sẵn
Đã xóa sách: Machine Learning Cơ Bản

Danh sách sách trong thư viện 'UIT Library':
Python 101: Lập trình cơ bản - Nguyễn Minh Tuấn | ISBN: ISBN001 | Giá: 180000 | Tình trạng: Còn sẵn
Trí tuệ Nhân tạo Ứng dụng - Lê Văn An | ISBN: ISBN003 | Giá: 270000 | Tình trạng: Còn sẵn
```

Câu 2:

[11]

```python
# DEMO 2
from abc import ABC, abstractmethod


class PaymentMethod(ABC):
    def __init__(self, amount, description):
        self.payment_amount = amount
        self.payment_description = description

    @abstractmethod
    def validate(self):
        pass

    @abstractmethod
    def process_payment(self):
        pass


class CreditCard(PaymentMethod):
    def __init__(self, amount, description, card_number, cvv):
        super().__init__(amount, description)
        self.card_number = card_number
        self.cvv = cvv

    def validate(self):
        return len(self.card_number) == 16 and len(self.cvv) == 3

    def process_payment(self):
        if self.validate():
            print(f"Giao dịch {self.payment_amount} bằng Thẻ tín dụng đã được thực hiện thành công.")
        else:
            print("Thanh toán thất bại: Số thẻ hoặc mã CVV không hợp lệ.")


class PayPal(PaymentMethod):
    def __init__(self, amount, description, email):
        super().__init__(amount, description)
        self.email = email

    def validate(self):
        return "@" in self.email and "." in self.email

    def process_payment(self):
        if self.validate():
            print(f"Giao dịch {self.payment_amount} thông qua PayPal ({self.email}) đã hoàn tất.")
        else:
            print("Thanh toán thất bại: Địa chỉ email PayPal không hợp lệ.")


class BankTransfer(PaymentMethod):
    def __init__(self, amount, description, account_number):
        super().__init__(amount, description)
```

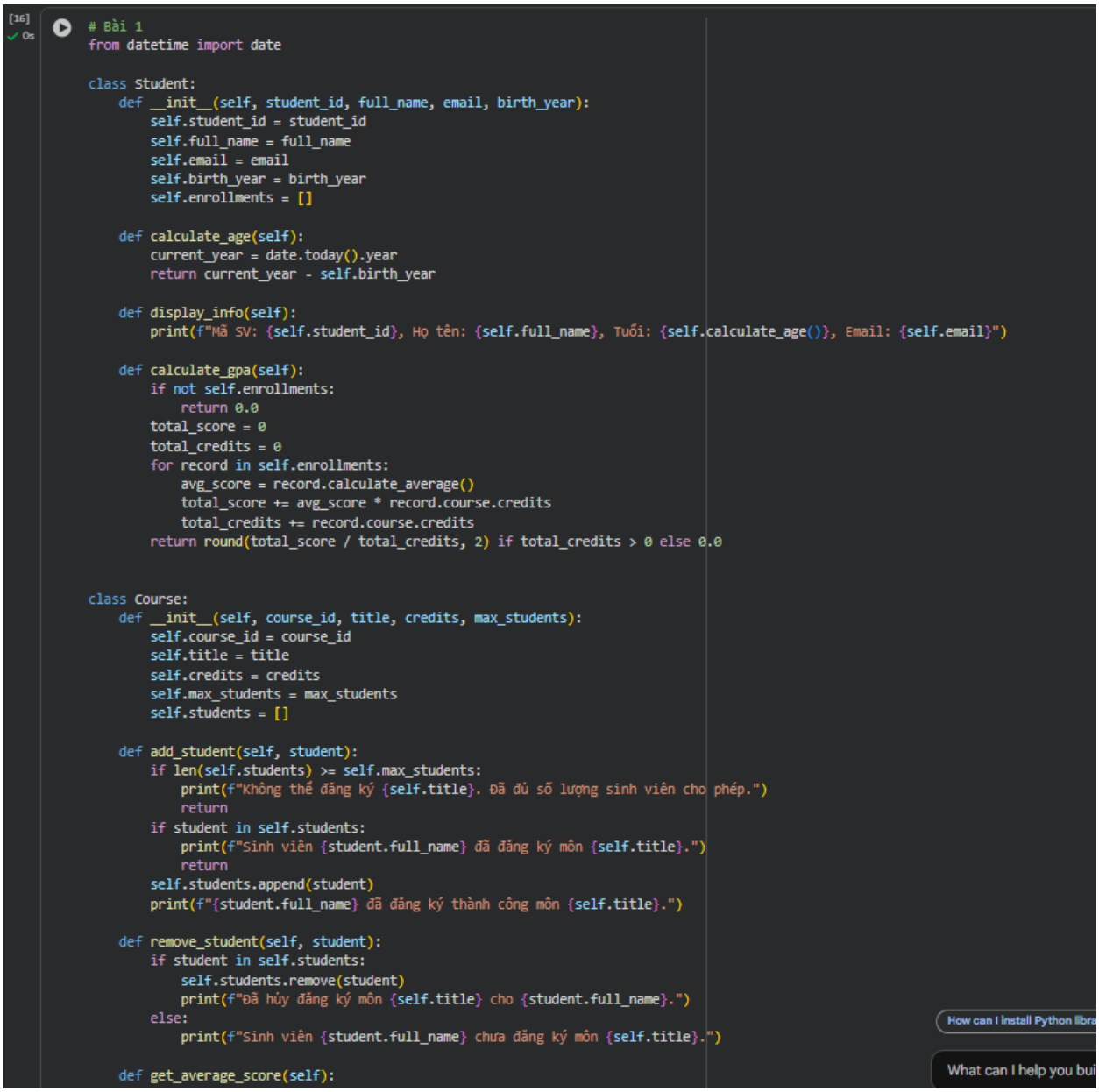
```python
class BankTransfer(PaymentMethod):
    def __init__(self, amount, description, account_number):
        super().__init__(amount, description)
        self.account_number = account_number

    def validate(self):
        return len(self.account_number) >= 8 and self.account_number.isdigit()

    def process_payment(self):
        if self.validate():
            print(f"Chuyển khoản {self.payment_amount} từ tài khoản {self.account_number} thành công.")
        else:
            print("Thanh toán thất bại: Số tài khoản không hợp lệ.")


class DigitalWallet(PaymentMethod):
    def __init__(self, amount, description, phone_number, pin):
        super().__init__(amount, description)
        self.phone_number = phone_number
        self.pin = pin

    def validate(self):
        return len(self.phone_number) == 10 and len(self.pin) == 4 and self.phone_number.isdigit()

    def process_payment(self):
        if self.validate():
            print(f"Giao dịch {self.payment_amount} bằng Ví điện tử ({self.phone_number}) thành công.")
        else:
            print("Thanh toán thất bại: Số điện thoại hoặc mã PIN không hợp lệ.")


class PaymentProcessor:
    def __init__(self):
        self.transaction_history = []

    def process(self, payment_method: PaymentMethod):
        print(f"\nĐang xử lý giao dịch: {payment_method.payment_description}")
        payment_method.process_payment()
        self.transaction_history.append(payment_method)

    def show_history(self):
        print("\nLịch sử giao dịch:")
        if not self.transaction_history:
            print("Chưa có giao dịch nào được thực hiện.")
        else:
            for transaction in self.transaction_history:
                print(f"- {transaction.payment_description}: "
                      f"{transaction.payment_amount} ({transaction.__class__.__name__})")


if __name__ == "__main__":
    processor = PaymentProcessor()
```

```python
if __name__ == "__main__":
    processor = PaymentProcessor()

    credit_payment = CreditCard(250000, "Mua laptop tại cửa hàng ABC", "4111222233334444", "321")
    paypal_payment = PayPal(80000, "Thanh toán khóa học trực tuyến", "hocvien@edu.vn")
    bank_payment = BankTransfer(1200000, "Đóng học phí kỳ 2", "99887766")
    wallet_payment = DigitalWallet(35000, "Mua cà phê sáng", "0912345678", "9876")

    processor.process(credit_payment)
    processor.process(paypal_payment)
    processor.process(bank_payment)
    processor.process(wallet_payment)

    processor.show_history()
```

```
...
Đang xử lý giao dịch: Mua laptop tại cửa hàng ABC
Giao dịch 250000 bằng Thẻ tín dụng đã được thực hiện thành công.

Đang xử lý giao dịch: Thanh toán khóa học trực tuyến
Giao dịch 80000 thông qua PayPal (hocvien@edu.vn) đã hoàn tất.

Đang xử lý giao dịch: Đóng học phí kỳ 2
Chuyển khoản 1200000 từ tài khoản 99887766 thành công.

Đang xử lý giao dịch: Mua cà phê sáng
Giao dịch 35000 bằng Ví điện tử (0912345678) thành công.

Lịch sử giao dịch:
- Mua laptop tại cửa hàng ABC: 250000 (CreditCard)
- Thanh toán khóa học trực tuyến: 80000 (PayPal)
- Đóng học phí kỳ 2: 1200000 (BankTransfer)
- Mua cà phê sáng: 35000 (DigitalWallet)
```

BÀI TẬP VỀ NHÀ:

Bài 1:



```
[16]
✓ 0s
# Bài 1
from datetime import date

class Student:
    def __init__(self, student_id, full_name, email, birth_year):
        self.student_id = student_id
        self.full_name = full_name
        self.email = email
        self.birth_year = birth_year
        self.enrollments = []

    def calculate_age(self):
        current_year = date.today().year
        return current_year - self.birth_year

    def display_info(self):
        print(f"Mã SV: {self.student_id}, Họ tên: {self.full_name}, Tuổi: {self.calculate_age()}, Email: {self.email}")

    def calculate_gpa(self):
        if not self.enrollments:
            return 0.0
        total_score = 0
        total_credits = 0
        for record in self.enrollments:
            avg_score = record.calculate_average()
            total_score += avg_score * record.course.credits
            total_credits += record.course.credits
        return round(total_score / total_credits, 2) if total_credits > 0 else 0.0


class Course:
    def __init__(self, course_id, title, credits, max_students):
        self.course_id = course_id
        self.title = title
        self.credits = credits
        self.max_students = max_students
        self.students = []

    def add_student(self, student):
        if len(self.students) >= self.max_students:
            print(f"Không thể đăng ký {self.title}. Đã đủ số lượng sinh viên cho phép.")
            return
        if student in self.students:
            print(f"Sinh viên {student.full_name} đã đăng ký môn {self.title}.")
            return
        self.students.append(student)
        print(f"{student.full_name} đã đăng ký thành công môn {self.title}.")

    def remove_student(self, student):
        if student in self.students:
            self.students.remove(student)
            print(f"Đã hủy đăng ký môn {self.title} cho {student.full_name}.")
        else:
            print(f"Sinh viên {student.full_name} chưa đăng ký môn {self.title}.")

    def get_average_score(self):
```

How can I install Python libra

What can I help you bui

[16] ✓ 0s

```
    def get_average_score(self):
        total = 0
        count = 0
        for record in Enrollment.all_enrollments:
            if record.course == self:
                total += record.calculate_average()
                count += 1
        return round(total / count, 2) if count > 0 else 0.0


class Enrollment:
    all_enrollments = []

    def __init__(self, student, course):
        self.student = student
        self.course = course
        self.midterm = 0
        self.final = 0
        self.assignment = 0
        Enrollment.all_enrollments.append(self)
        student.enrollments.append(self)

    def update_scores(self, midterm, final, assignment):
        self.midterm = midterm
        self.final = final
        self.assignment = assignment
        print(f"Đã cập nhật điểm cho {self.student.full_name} - {self.course.title}.")

    def calculate_average(self):
        return round(self.midterm * 0.3 + self.final * 0.5 + self.assignment * 0.2, 2)

    def get_grade(self):
        avg = self.calculate_average()
        if avg >= 8.5:
            return "Giỏi"
        elif avg >= 7.0:
            return "Khá"
        elif avg >= 5.0:
            return "Trung bình"
        else:
            return "Yếu"
```

[16]
✓ 0s

```python
if __name__ == "__main__":
    stu1 = Student("SV101", "Phạm Minh Quân", "quan.pham@uit.edu.vn", 2003)
    stu2 = Student("SV102", "Trần Diệu Linh", "linh.tran@uit.edu.vn", 2004)
    stu3 = Student("SV103", "Ngô Hữu Đạt", "dat.ngo@uit.edu.vn", 2002)

    cs1 = Course("CS201", "Lập trình hướng đối tượng", 3, 2)
    cs2 = Course("CS202", "Phân tích dữ liệu", 4, 3)

    cs1.add_student(stu1)
    cs1.add_student(stu2)
    cs1.add_student(stu3)
    cs2.add_student(stu1)
    cs2.add_student(stu3)

    en1 = Enrollment(stu1, cs1)
    en2 = Enrollment(stu2, cs1)
    en3 = Enrollment(stu1, cs2)
    en4 = Enrollment(stu3, cs2)

    en1.update_scores(9, 8, 9)
    en2.update_scores(6, 7, 5)
    en3.update_scores(8, 8, 7)
    en4.update_scores(9, 9, 9)

    print("\nDANH SÁCH SINH VIÊN:")
    for s in [stu1, stu2, stu3]:
        s.display_info()
        print(f"GPA: {s.calculate_gpa()}\n")

    print("ĐIỂM TRUNG BÌNH CÁC MÔN:")
    print(f"- {cs1.title}: {cs1.get_average_score()}")
    print(f"- {cs2.title}: {cs2.get_average_score()}")

    print("\nKẾT QUẢ TÌM KIẾM TÊN 'Linh':")
    keyword = "Linh"
    found = False
    for s in [stu1, stu2, stu3]:
        if keyword.lower() in s.full_name.lower():
            s.display_info()
            found = True
    if not found:
        print("Không tìm thấy sinh viên phù hợp.")
```

```
Phạm Minh Quân đã đăng ký thành công môn Lập trình hướng đối tượng.
Trần Diệu Linh đã đăng ký thành công môn Lập trình hướng đối tượng.
Không thể đăng ký Lập trình hướng đối tượng. Đã đủ số lượng sinh viên cho phép.
Phạm Minh Quân đã đăng ký thành công môn Phân tích dữ liệu.
Ngô Hữu Đạt đã đăng ký thành công môn Phân tích dữ liệu.
Đã cập nhật điểm cho Phạm Minh Quân - Lập trình hướng đối tượng.
Đã cập nhật điểm cho Trần Diệu Linh - Lập trình hướng đối tượng.
Đã cập nhật điểm cho Phạm Minh Quân - Phân tích dữ liệu.
Đã cập nhật điểm cho Ngô Hữu Đạt - Phân tích dữ liệu.

DANH SÁCH SINH VIÊN:
Mã SV: SV101, Họ tên: Phạm Minh Quân, Tuổi: 22, Email: quan.pham@uit.edu.vn
GPA: 8.1

Mã SV: SV102, Họ tên: Trần Diệu Linh, Tuổi: 21, Email: linh.tran@uit.edu.vn
GPA: 6.3

Mã SV: SV103, Họ tên: Ngô Hữu Đạt, Tuổi: 23, Email: dat.ngo@uit.edu.vn
GPA: 9.0

ĐIỂM TRUNG BÌNH CÁC MÔN:
- Lập trình hướng đối tượng: 7.4
- Phân tích dữ liệu: 8.4

KẾT QUẢ TÌM KIẾM TÊN 'Linh':
Mã SV: SV102, Họ tên: Trần Diệu Linh, Tuổi: 21, Email: linh.tran@uit.edu.vn
```

Bài 2:

[22]

```python
# Bài 2
import uuid
from datetime import datetime

class Transaction:
    def __init__(self, tran_type, amount, balance):
        self.tran_type = tran_type
        self.amount = amount
        self.balance = balance
        self.time = datetime.now()

    def __str__(self):
        return f"{self.time.strftime('%Y-%m-%d %H:%M:%S')} | {self.tran_type:<15} | {self.amount:>10,.2f} | Số dư: {self.balance:>10,.2f}"


class BankAccount:
    def __init__(self, owner, balance=0.0):
        self.account_number = str(uuid.uuid4())[:8]
        self.owner = owner
        self.balance = balance
        self.transactions = []

    def deposit(self, amount):
        if amount <= 0:
            print("Số tiền gửi phải lớn hơn 0.")
            return
        self.balance += amount
        self.transactions.append(Transaction("Gửi tiền", amount, self.balance))
        print(f"Gửi {amount:.2f} vào tài khoản {self.account_number}. Số dư hiện tại: {self.balance:.2f}")

    def withdraw(self, amount):
        if amount <= 0:
            print("Số tiền rút phải lớn hơn 0.")
            return False
        if self.balance < amount:
            print("Số dư không đủ để rút.")
            return False
        self.balance -= amount
        self.transactions.append(Transaction("Rút tiền", -amount, self.balance))
        print(f"Rút {amount:.2f} từ tài khoản {self.account_number}. Số dư còn lại: {self.balance:.2f}")
        return True

    def get_balance(self):
        return self.balance

    def show_history(self):
        print(f"\nLịch sử giao dịch của tài khoản {self.account_number} ({self.owner}):")
        if not self.transactions:
            print("Không có giao dịch nào.")
        else:
            for t in self.transactions:
                print(t)
        print("-" * 60)
```

```python
class SavingsAccount(BankAccount):
    def __init__(self, owner, balance=0.0, interest_rate=0.03, withdraw_limit=3):
        super().__init__(owner, balance)
        self.interest_rate = interest_rate
        self.withdraw_count = 0
        self.withdraw_limit = withdraw_limit

    def withdraw(self, amount):
        if self.withdraw_count >= self.withdraw_limit:
            print("Đã vượt quá số lần rút tiền cho phép trong kỳ.")
            return False
        if super().withdraw(amount):
            self.withdraw_count += 1
            return True
        return False

    def calculate_interest(self):
        return self.balance * self.interest_rate

    def apply_interest(self):
        interest = self.calculate_interest()
        self.balance += interest
        self.transactions.append(Transaction("Cộng lãi", interest, self.balance))
        print(f"Đã cộng lãi {interest:.2f}. Số dư mới: {self.balance:.2f}")


class CheckingAccount(BankAccount):
    def __init__(self, owner, balance=0.0, overdraft_limit=500.0, fee=5.0):
        super().__init__(owner, balance)
        self.overdraft_limit = overdraft_limit
        self.fee = fee

    def withdraw(self, amount):
        if amount <= 0:
            print("Số tiền rút phải lớn hơn 0.")
            return False
        if self.balance + self.overdraft_limit < amount:
            print("Vượt quá hạn mức thấu chi.")
            return False
        self.balance -= (amount + self.fee)
        self.transactions.append(Transaction("Rút tiền + phí", -(amount + self.fee), self.balance))
        print(f"Rút {amount:.2f} (phí {self.fee:.2f}). Số dư hiện tại: {self.balance:.2f}")
        return True


def transfer(from_account, to_account, amount):
    print(f"\nThực hiện chuyển {amount:.2f} từ {from_account.account_number} → {to_account.account_number}")
    if from_account.withdraw(amount):
        to_account.deposit(amount)
        from_account.transactions.append(Transaction("Chuyển tiền đi", -amount, from_account.balance))
        to_account.transactions.append(Transaction("Nhận tiền đến", amount, to_account.balance))
        print("Chuyển khoản thành công!")
    else:
```

```python
def transfer(from_account, to_account, amount):
    print(f"\nThực hiện chuyển {amount:.2f} từ {from_account.account_number} → {to_account.account_number}")
    if from_account.withdraw(amount):
        to_account.deposit(amount)
        from_account.transactions.append(Transaction("Chuyển tiền đi", -amount, from_account.balance))
        to_account.transactions.append(Transaction("Nhận tiền đến", amount, to_account.balance))
        print("Chuyển khoản thành công!")
    else:
        print("Chuyển khoản thất bại.")


if __name__ == "__main__":
    acc1 = SavingsAccount("Nguyễn Văn A", 1000000)
    acc2 = CheckingAccount("Trần Thị B", 500000)

    acc1.deposit(500000)
    acc1.withdraw(200000)
    acc1.withdraw(100000)
    acc1.apply_interest()

    acc2.withdraw(600000)
    acc2.deposit(300000)

    transfer(acc1, acc2, 400000)

    acc1.show_history()
    acc2.show_history()
```

```
Gửi 500000.00 vào tài khoản b655f9cc. Số dư hiện tại: 1500000.00
Rút 200000.00 từ tài khoản b655f9cc. Số dư còn lại: 1300000.00
Rút 100000.00 từ tài khoản b655f9cc. Số dư còn lại: 1200000.00
Đã cộng lãi 36000.00. Số dư mới: 1236000.00
Vượt quá hạn mức thấu chi.
Gửi 300000.00 vào tài khoản 3dd91da7. Số dư hiện tại: 800000.00

Thực hiện chuyển 400000.00 từ b655f9cc → 3dd91da7
Rút 400000.00 từ tài khoản b655f9cc. Số dư còn lại: 836000.00
Gửi 400000.00 vào tài khoản 3dd91da7. Số dư hiện tại: 1200000.00
Chuyển khoản thành công!

Lịch sử giao dịch của tài khoản b655f9cc (Nguyễn Văn A):
2025-11-06 15:24:48 | Gửi tiền        | 500,000.00 | Số dư: 1,500,000.00
2025-11-06 15:24:48 | Rút tiền        | -200,000.00 | Số dư: 1,300,000.00
2025-11-06 15:24:48 | Rút tiền        | -100,000.00 | Số dư: 1,200,000.00
2025-11-06 15:24:48 | Cộng lãi        |  36,000.00 | Số dư: 1,236,000.00
2025-11-06 15:24:48 | Rút tiền        | -400,000.00 | Số dư: 836,000.00
2025-11-06 15:24:48 | Chuyển tiền đi  | -400,000.00 | Số dư: 836,000.00
------------------------------------------------------------

Lịch sử giao dịch của tài khoản 3dd91da7 (Trần Thị B):
2025-11-06 15:24:48 | Gửi tiền        | 300,000.00 | Số dư: 800,000.00
2025-11-06 15:24:48 | Gửi tiền        | 400,000.00 | Số dư: 1,200,000.00
2025-11-06 15:24:48 | Nhận tiền đến   | 400,000.00 | Số dư: 1,200,000.00
------------------------------------------------------------
```

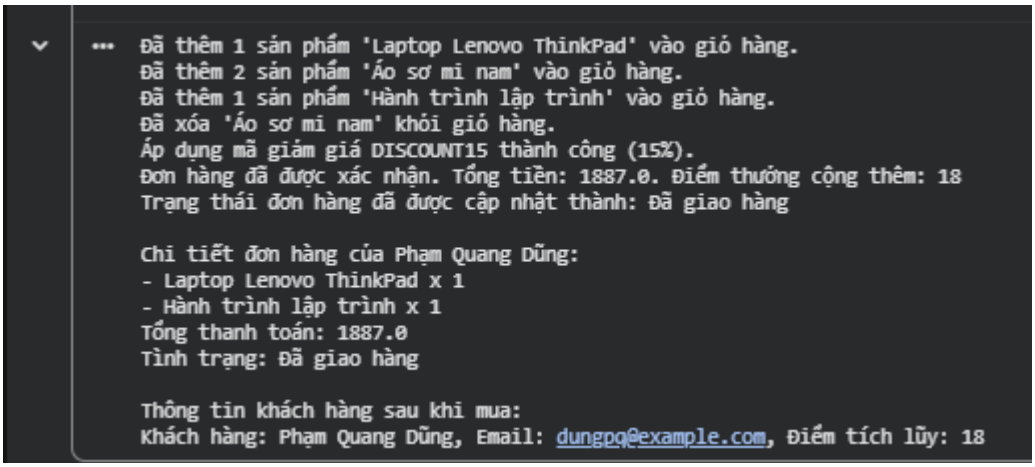
Bài 3:

```
[24]
✓ 0s
```

```python
# Bài 3
from datetime import date

class Product:
    def __init__(self, code, title, price, stock):
        self.code = code
        self.title = title
        self.price = price
        self.stock = stock

    def display_info(self):
        print(f"Mã: {self.code} | Tên: {self.title} | Giá: {self.price} | Tồn kho: {self.stock}")

    def update_stock(self, quantity):
        self.stock += quantity


class Electronics(Product):
    def __init__(self, code, title, price, stock, warranty):
        super().__init__(code, title, price, stock)
        self.warranty = warranty

    def display_info(self):
        print(f"[Điện tử] {self.title} - Giá: {self.price}, Bảo hành: {self.warranty} năm, Tồn kho: {self.stock}")


class Clothing(Product):
    def __init__(self, code, title, price, stock, size):
        super().__init__(code, title, price, stock)
        self.size = size

    def display_info(self):
        print(f"[Quần áo] {self.title} - Giá: {self.price}, Size: {self.size}, Tồn kho: {self.stock}")


class Book(Product):
    def __init__(self, code, title, price, stock, author):
        super().__init__(code, title, price, stock)
        self.author = author

    def display_info(self):
        print(f"[Sách] {self.title} - Tác giả: {self.author}, Giá: {self.price}, Tồn kho: {self.stock}")


class Customer:
    def __init__(self, fullname, email, points=0):
        self.fullname = fullname
        self.email = email
        self.points = points

    def display_info(self):
        print(f"Khách hàng: {self.fullname}, Email: {self.email}, Điểm tích lũy: {self.points}")
```

```python
class Coupon:
    def __init__(self, code, discount, expiry):
        self.code = code
        self.discount = discount
        self.expiry = expiry

    def is_valid(self):
        return date.today() <= self.expiry


class ShoppingCart:
    def __init__(self):
        self.items = []
        self.coupon = None

    def add_item(self, product, quantity):
        if product.stock < quantity:
            print(f"Sản phẩm {product.title} không đủ số lượng tồn kho.")
            return
        self.items.append((product, quantity))
        product.update_stock(-quantity)
        print(f"Đã thêm {quantity} sản phẩm '{product.title}' vào giỏ hàng.")

    def remove_item(self, code):
        for item in self.items:
            if item[0].code == code:
                item[0].update_stock(item[1])
                self.items.remove(item)
                print(f"Đã xóa '{item[0].title}' khỏi giỏ hàng.")
                return
        print("Không tìm thấy sản phẩm cần xóa trong giỏ hàng.")

    def apply_coupon(self, coupon):
        if coupon.is_valid():
            self.coupon = coupon
            print(f"Áp dụng mã giảm giá {coupon.code} thành công ({coupon.discount}%).")
        else:
            print(f"Mã giảm giá {coupon.code} đã hết hạn.")

    def calculate_total(self):
        total = sum(p.price * q for p, q in self.items)
        if self.coupon:
            total *= (1 - self.coupon.discount / 100)
        return round(total, 2)
```

```python
class Order:
    def __init__(self, customer, cart):
        self.customer = customer
        self.items = cart.items.copy()
        self.total = cart.calculate_total()
        self.status = "Chờ xác nhận"

    def confirm(self):
        self.status = "Đã xác nhận"
        earned_points = int(self.total * 0.01)
        self.customer.points += earned_points
        print(f"Đơn hàng đã được xác nhận. Tổng tiền: {self.total}. Điểm thưởng cộng thêm: {earned_points}")

    def update_status(self, new_status):
        self.status = new_status
        print(f"Trạng thái đơn hàng đã được cập nhật thành: {self.status}")

    def display_order(self):
        print(f"\nChi tiết đơn hàng của {self.customer.fullname}:")
        for p, q in self.items:
            print(f"- {p.title} x {q}")
        print(f"Tổng thanh toán: {self.total}")
        print(f"Tình trạng: {self.status}")


if __name__ == "__main__":
    laptop = Electronics("E101", "Laptop Lenovo ThinkPad", 1800, 4, 3)
    tshirt = Clothing("C201", "Áo sơ mi nam", 350, 6, "M")
    novel = Book("B301", "Hành trình lập trình", 420, 10, "Lê Minh")

    customer = Customer("Phạm Quang Dũng", "dungpq@example.com")

    cart = ShoppingCart()
    cart.add_item(laptop, 1)
    cart.add_item(tshirt, 2)
    cart.add_item(novel, 1)
    cart.remove_item("C201")

    coupon = Coupon("DISCOUNT15", 15, date(2025, 12, 31))
    cart.apply_coupon(coupon)

    order = Order(customer, cart)
    order.confirm()
    order.update_status("Đã giao hàng")
    order.display_order()

    print("\nThông tin khách hàng sau khi mua:")
    customer.display_info()
```


Đã thêm 1 sản phẩm 'Laptop Lenovo ThinkPad' vào giỏ hàng.
Đã thêm 2 sản phẩm 'Áo sơ mi nam' vào giỏ hàng.
Đã thêm 1 sản phẩm 'Hành trình lập trình' vào giỏ hàng.
Đã xóa 'Áo sơ mi nam' khỏi giỏ hàng.
Áp dụng mã giảm giá DISCOUNT15 thành công (15%).
Đơn hàng đã được xác nhận. Tổng tiền: 1887.0. Điểm thưởng cộng thêm: 18
Trạng thái đơn hàng đã được cập nhật thành: Đã giao hàng
Chi tiết đơn hàng của Phạm Quang Dũng:
- Laptop Lenovo ThinkPad x 1
- Hành trình lập trình x 1
Tổng thanh toán: 1887.0
Tình trạng: Đã giao hàng
Thông tin khách hàng sau khi mua:
Khách hàng: Phạm Quang Dũng, Email: dungpq@example.com, Điểm tích lũy: 18