TRƯỜNG ĐẠI HỌC CÔNG NGHỆ
KHOA CÔNG NGHỆ THÔNG TIN

BÀI TẬP MÔN HỌC

Kiểm thử và đảm bảo chất lượng phần mềm

BÀI TẬP: KIỂM THỬ DÒNG ĐIỀU KHIỂN

Sinh viên: Đỗ Hữu Hoàng Tùng

Mã sinh viên: 22026551

1. Đặc tả bài toán

Chương trình mô phỏng đơn giản quá trình giao dịch trong ngân hàng, nhận 3 thành phần chính là đầu vào:

1. balance: Số tiền hiện tại có trong tài khoản
2. transaction_type: Loại giao dịch, có thể nhận 1 trong 2 giá trị:
    a. deposit: Giao dịch bỏ tiền vào tài khoản
    b. withdrawal: Giao dịch rút tiền khỏi tài khoản
3. amount: Số tiền sử dụng vào giao dịch

Tùy vào loại giao dịch, số balance trong tài khoản sẽ được thêm vào hoặc bớt đi 1 lượng tương ứng với amount. Chương trình có 1 số ràng buộc nhất định như sau:

1. amount và balance đều phải là 1 số tự nhiên
2. transaction_type chỉ có thể nhận 1 trong 2 giá trị là deposit và withdrawal
3. Do vấn đề an ninh, chỉ cho phép số lượng amount tối đa trong mỗi giao dịch là 1 triệu
4. Nếu transaction_type là withdrawal thì amount không được lớn hơn balance

2. Cài đặt chương trình

Sử dụng ngôn ngữ Python để viết hàm cho chương trình

```
def bank_transaction(balance: int, transaction_type:str, amount:int) ->
int:
    """
    Perform a bank transaction (deposit or withdrawal) on the current
amount.

    Parameters:
    balance (int): The current amount in the bank account.
    transaction_type (str): The type of transaction ('deposit' or
'withdrawal').
    amount (int): The amount to deposit or withdraw.

    Returns:
    int: The new amount in the bank account after the transaction.
```

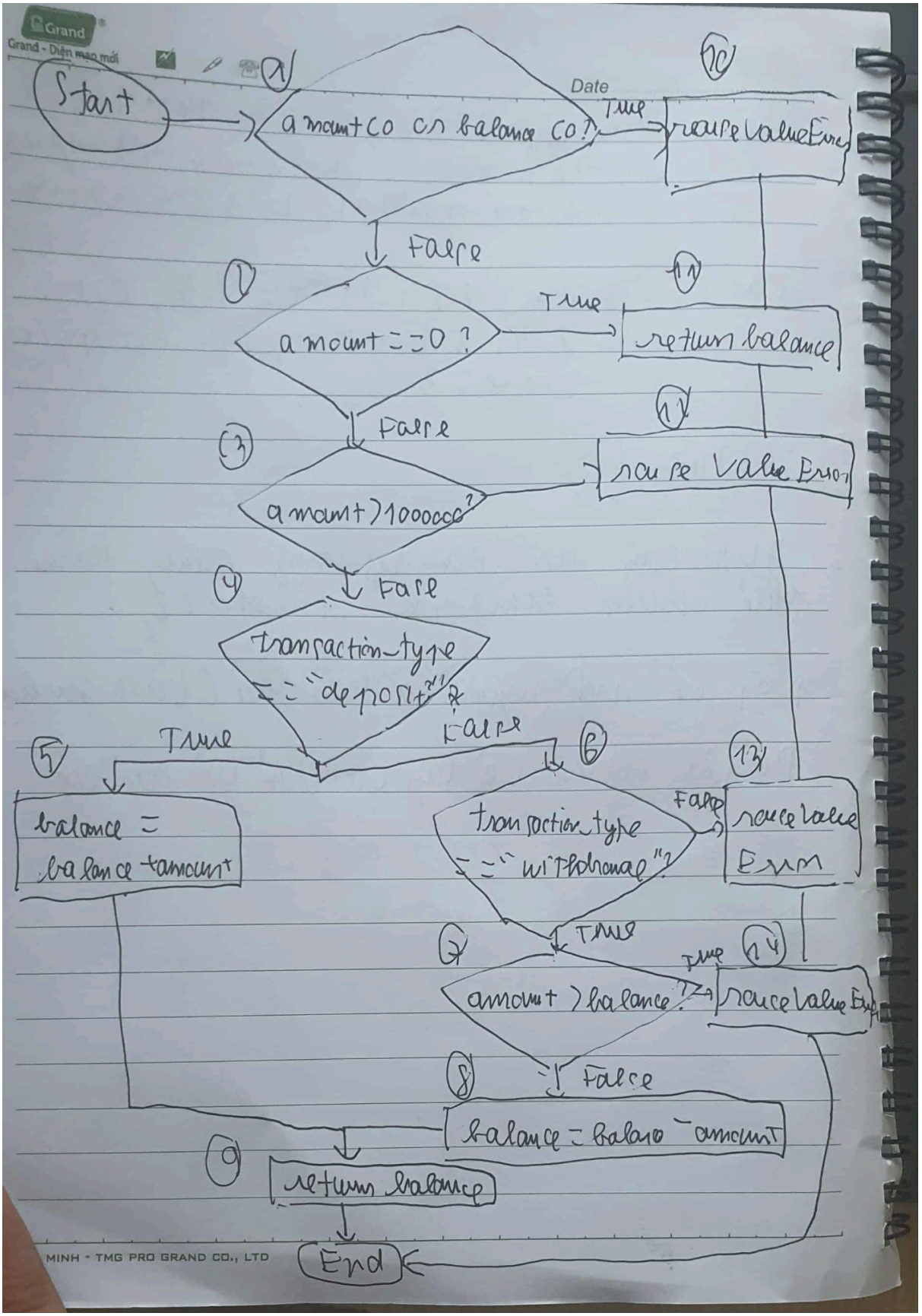
```
    """
    if amount < 0 or balance < 0:
        raise ValueError("Input not valid.")
    if amount == 0:
        return balance
    if amount > 1000000:
        raise ValueError("Input not valid.")
    if transaction_type == 'deposit':
        balance = balance + amount
    elif transaction_type == 'withdrawal':
        if amount > balance:
            raise ValueError("Input not valid.")
        balance = balance - amount
    else:
        raise ValueError("Input not valid.")
    return balance
```

1. Đoạn code chương trình giao dịch ngân hàng

3. Kiểm thử dòng điều khiển

3.1 Đồ thị dòng điều khiển

Dựa vào mã nguồn trên, ta vẽ được đồ thị dòng điều khiển như sau:



Hình 1. Đồ thị dòng điều khiển của chương trình

Trong đợt kiểm thử này, ta sử dụng độ phủ nhánh (C2), từ đồ thị ở trên nhận thấy muốn đạt 100% độ phủ thì cần phải đi qua những con đường sau:

- 1 -> 10
- 1 -> 2 -> 3 -> 4 -> 5 -> 9
- 1 -> 2 -> 11
- 1 -> 2 -> 3 -> 12
- 1 -> 2 -> 3 -> 4 -> 6 -> 7 -> 8 ->9
- 1 -> 2 -> 3 -> 4 -> 6 -> 13
- 1 -> 2 -> 3 -> 4 -> 6 -> 7 -> 14

Do có 7 con đường nên tương ứng sẽ có 7 ca kiểm thử để thỏa mãn độ phủ đạt 100%

STT	Input	Expected Output
1	balance: -10 transaction_type: deposit amount: 500	Raise Value Error
2	balance: 1000 transaction_type: deposit amount: 5000	6000
3	balance: 1000 transaction_type: deposit amount: 0	1000
4	balance: 1000 transaction_type: deposit amount: 1000001	Raise Value Error
5	balance: 1000 transaction_type: withdrawal amount: 500	500
6	balance: 1000 transaction_type: transfer	Raise Value Error

	amount: 500	
7	balance: 1000 transaction_type: withdrawal amount: 1500	Raise Value Error

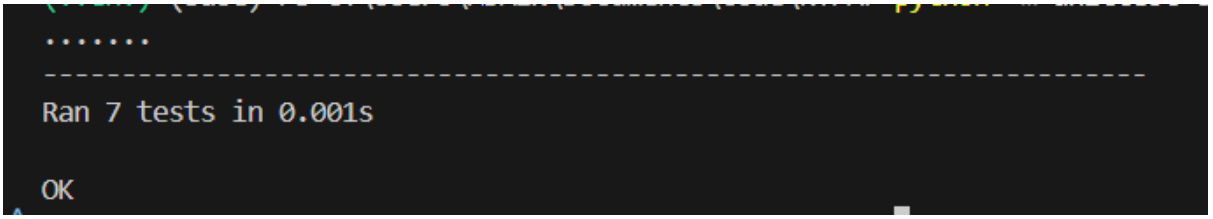
Bảng 1. Bảng kiểm thử theo bảng quyết định

```
class TestBankTransactionControlFlowC2Method(unittest.TestCase):
    def test_1(self):
        with self.assertRaises(ValueError):
            bank_transaction(-10, 'deposit', 500)

    def test_2(self):
        self.assertEqual(bank_transaction(1000, 'deposit', 5000), 6000)

    def test_3(self):
        self.assertEqual(bank_transaction(1000, 'deposit', 0), 1000)

    def test_4(self):
        with self.assertRaises(ValueError):
            bank_transaction(1000, 'deposit', 1000001)

    def test_5(self):
        self.assertEqual(bank_transaction(1000, 'withdrawal', 500),
500)

    def test_6(self):
        with self.assertRaises(ValueError):
            bank_transaction(1000, 'transfer', 500)

    def test_7(self):
        with self.assertRaises(ValueError):
            bank_transaction(1000, 'withdrawal', 1500)
```

2. Đoạn code xây dựng các ca kiểm thử theo dòng điều khiển

3.3 Kết quả kiểm thử

Sử dụng công cụ kiểm thử được tích hợp sẵn Pyunit của Python để tạo và chạy các ca kiểm thử vừa được sinh ra



Hình 1. Kết quả kiểm thử

STT	Input	Expected Output	Actual Output	Status
1	balance: -10 transaction_type: deposit amount: 500	Raise Value Error	Raise Value Error	Passed
2	balance: 1000 transaction_type: deposit amount: 5000	6000	6000	Passed
3	balance: 1000 transaction_type: deposit amount: 0	1000	1000	Passed
4	balance: 1000 transaction_type: deposit amount: 1000001	Raise Value Error	Raise Value Error	Passed
5	balance: 1000 transaction_type: withdrawal amount: 500	500	500	Passed
6	balance: 1000 transaction_type: transfer amount: 500	Raise Value Error	Raise Value Error	Passed
7	balance: 1000 transaction_type: withdrawal amount: 1500	Raise Value Error	Raise Value Error	Passed

Bảng 2. Bảng báo cáo kiểm thử theo dòng điều khiển

4. Nhận xét và kết luận

Từ kết quả kiểm thử ở trên, ta thấy với phương pháp kiểm thử theo dòng điều khiển độ phủ nhánh C2, chương trình vượt qua được toàn bộ các ca kiểm thử. Tuy nhiên cũng không thể dựa vào đó để kết luận là chương trình không còn lỗi.Nhược điểm của độ phủ nhánh C2 là gây thiếu ca kiểm thử với một số ngôn ngữ cho phép xử lý tắt (short circuit) các phép toán logic, điều này cũng được thể hiện tại khối (1) trong đồ thị (amount < 0 or balance <0), ca kiểm thử amount < 0 đã bị bỏ qua, cho thấy sự hạn chế của phương pháp này và nhấn mạnh lại sự quan trọng của việc kết hợp nhiều phương pháp kiểm thử với nhau