Phân Tích Hướng Đối Tượng (OOA)

Hệ thống quản lý tài khoản ngân hàng được xây dựng dựa trên các đối tượng chính sau:

- **Khách Hàng (Customer):** Một thực thể duy nhất, có thể sở hữu nhiều tài khoản ngân hàng. Thuộc tính bao gồm mã khách hàng và họ tên.
- **Tài Khoản (Account):** Đại diện cho một tài khoản ngân hàng cơ bản. Đây là đối tượng cốt lõi, có các thuộc tính như số tài khoản, số dư, và lịch sử giao dịch. Các hành động chính bao gồm nạp tiền, rút tiền và hiển thị thông tin.
- **Giao Dịch (Transaction):** Lưu trữ chi tiết của một giao dịch cụ thể, bao gồm số tiền, loại giao dịch (nạp/rút/chuyển khoản), và thời gian thực hiện.
- **Ngân Hàng (Bank):** Đối tượng quản lý toàn bộ hệ thống, bao gồm danh sách các khách hàng và tài khoản. Các hành động chính là tạo khách hàng, tạo tài khoản và chuyển tiền.

Thiết Kế Lớp và Nguyên Lý Hướng Đối Tượng

Thiết kế của chương trình tận dụng các nguyên lý cốt lõi của Lập trình Hướng Đối Tượng (OOP) để tạo ra một hệ thống linh hoạt, dễ bảo trì và mở rộng.

- **Kế Thừa (Inheritance):**
    - Lớp TaiKhoanTietKiem kế thừa từ lớp cơ sở TaiKhoan.
    - **Lợi ích:** Kế thừa cho phép lớp TaiKhoanTietKiem tự động có các thuộc tính và phương thức của TaiKhoan (như soTaiKhoan, soDu, napTien). Điều này giúp tái sử dụng mã hiệu quả và giảm sự trùng lặp.
    - **Tính đa hình (Polymorphism):** Bằng cách ghi đè (override) phương thức rutTien() trong lớp TaiKhoanTietKiem, chúng tôi có thể áp dụng các quy tắc rút tiền riêng biệt (ví dụ: giới hạn số lần rút) mà không ảnh hưởng đến logic của lớp TaiKhoan cơ bản.
- **Nạp Chồng Toán Tử (Operator Overloading):**
    - **Toán tử +=:** Được nạp chồng trong lớp TaiKhoan để cho phép thêm một đối tượng GiaoDich vào tài khoản một cách trực quan, cập nhật số dư tương ứng.
    - **Toán tử == và <:** Được nạp chồng để so sánh số dư giữa hai đối tượng TaiKhoan, giúp việc so sánh trở nên tự nhiên và dễ đọc hơn.

- **Lợi ích:** Nạp chồng toán tử giúp mã nguồn trở nên trực quan và gần gũi hơn với cách chúng ta suy nghĩ. Ví dụ, thay vì viết taiKhoan1.themGiaoDich(gd), chúng ta có thể đơn giản viết *tk1 += gd;.

Giải Thích Code

Chương trình được xây dựng trên ngôn ngữ C++ và bao gồm các lớp chính đã được phân tích ở trên.

- **GiaoDich:** Lớp này sử dụng hàm strftime() để định dạng thời gian của giao dịch. strftime() là một phương thức an toàn và linh hoạt hơn so với ctime() khi làm việc với các định dạng ngày giờ tùy chỉnh.
- **TaiKhoan:** Lớp cơ sở chứa các phương thức napTien() và rutTien() cơ bản, cùng với logic để quản lý lichSuGiaoDich.
- **TaiKhoanTietKiem:** Kế thừa từ TaiKhoan, lớp này có thêm thuộc tính laiSuat và gioiHanRut. Phương thức rutTien() được ghi đè để kiểm tra xem số lần rút tiền đã vượt quá giới hạn chưa.
- **KhachHang:** Lớp này sử dụng **con trỏ thông minh** unique_ptr<TaiKhoan> để quản lý các tài khoản của khách hàng. Điều này giúp tự động hóa việc giải phóng bộ nhớ, tránh rò rỉ bộ nhớ (memory leak).
- **NganHang:** Lớp quản lý chính, điều phối các hoạt động giữa các đối tượng.

Kết Quả Kiểm Thử (Test Results)

Kết quả đầu ra dưới đây minh họa các chức năng cốt lõi, đặc biệt là kế thừa và nạp chồng toán tử.

Đầu ra mẫu:

Plaintext

=== HE THONG QUAN LY TAI KHOAN NGAN HANG ===

Khoi tao he thong...

--- Tao khach hang ---

Da tao khach hang moi: Nguyen Van An (Ma: KH1)

Da tao khach hang moi: Tran Thi Bich (Ma: KH2)

--- Tao tai khoan ---

Da tao tai khoan thuong: TK1000

Da them tai khoan TK1000 cho khach hang Nguyen Van An

Da tao tai khoan tiet kiem: TK1001

Da them tai khoan TK1001 cho khach hang Nguyen Van An

...

--- Su dung toan tu ---

Da them giao dich qua toan tu +=. So du moi: $800.00

...

--- Kiem tra tai khoan tiet kiem ---

Da reset so lan rut tien cho thang moi.

Da rut $10.00 tu tai khoan TK1001

Da rut $10.00 tu tai khoan TK1001

Da vuot qua so lan rut tien cho phep trong thang. Khong the thuc hien giao dich!

...

- **Demonstration:**
    - **Kế thừa:** Khi gọi tktk->rutTien(10, "Rut thu nghiem") lần thứ ba, chương trình đã in ra thông báo lỗi Da vuot qua so lan rut tien..., cho thấy phương thức rutTien() đã được ghi đè thành công với logic riêng của tài khoản tiết kiệm.
    - **Nạp chồng toán tử:** Câu lệnh *tk1 += gd; đã thực hiện thành công việc thêm giao dịch và cập nhật số dư, sau đó in ra số dư mới, chứng minh toán tử += hoạt động như mong đợi.

Sơ Đồ UML

Sơ Đồ Lớp (Class Diagram)

Giải thích sơ đồ:

- **Kế thừa:** Đường mũi tên rỗng chỉ từ TaiKhoanTietKiem đến TaiKhoan thể hiện mối quan hệ kế thừa (tính tổng quát hóa).
- **Mối quan hệ Tổng hợp (Composition):** Mũi tên kim cương đặc từ KhachHang đến TaiKhoan thể hiện rằng một khách hàng sở hữu các tài khoản.
- **Mối quan hệ Phụ thuộc (Dependency):** Đường mũi tên nét đứt từ NganHang đến KhachHang và TaiKhoan chỉ ra rằng lớp NganHang sử dụng các lớp này.
- **Nạp chồng toán tử:** Các toán tử +=, ==, < được liệt kê trong phần phương thức của lớp TaiKhoan để chỉ rõ chúng đã được nạp chồng.

Sơ Đồ Trình Tự (Sequence Diagram)

Giải thích sơ đồ:

- Sơ đồ này mô tả trình tự các bước khi một khách hàng muốn rút tiền.
- Khách hàng (Client) gọi phương thức rutTien() trên đối tượng TaiKhoan của mình.
- Đối tượng TaiKhoan kiểm tra điều kiện (số dư có đủ không).
- Nếu đủ, nó cập nhật số dư và tạo một đối tượng GiaoDich mới, sau đó thêm vào lịch sử giao dịch của nó.
- Sơ đồ cho thấy luồng hoạt động rõ ràng, từ việc gọi hàm đến việc cập nhật dữ liệu.

Sử Dụng LLM

Em đã sử dụng mô hình ngôn ngữ lớn (LLM) để hỗ trợ quá trình phân tích và thiết kế, cụ thể là:

- **Phân tích hướng đối tượng:** Em đã sử dụng LLM để xác định các thực thể và mối quan hệ chính trong một hệ thống ngân hàng cơ bản, với prompt: *"Phân tích hướng đối tượng cho một hệ thống quản lý ngân hàng đơn giản. Xác định các lớp chính, thuộc tính và mối quan hệ của chúng."*
- **Nạp chồng toán tử:** Em đã hỏi LLM để tìm ý tưởng cho việc nạp chồng toán tử trong lớp TaiKhoan với prompt: *"Gợi ý cách nạp chồng các toán tử trong một lớp tài khoản ngân hàng để làm cho mã nguồn trực quan hơn."* Phản hồi đã đưa ra ý tưởng về việc sử dụng += để thêm giao dịch và < để so sánh số dư.

- **Cải tiến code:** Em cũng tham khảo LLM về cách xử lý lỗi font tiếng Việt trong console C++, dẫn đến việc sử dụng SetConsoleOutputCP(CP_UTF8) và SetConsoleCP(CP_UTF8) trên Windows.

Lưu ý: Tất cả mã nguồn đều tự viết. LLM chỉ được dùng như một công cụ hỗ trợ tư duy và tìm kiếm giải pháp.