I. Độ phức tạp cho các thao tác trên KhachHangList (danh sách khách hàng)

Cấu trúc KhachHangList là một danh sách liên kết đôi.

II. Độ phức tạp cho các thao tác trên GiaoDichList (danh sách giao dịch)

Tương tự như KhachHangList, GiaoDichList cũng là một danh sách liên kết đôi.

III. Độ phức tạp của các hàm trong lớp QuanLyNganHang

Lớp QuanLyNganHang sử dụng cả KhachHangList và GiaoDichList.

IV. Tổng kết và Đánh giá hiệu năng

Chương trình này sử dụng danh sách liên kết đôi thay vì mảng tuyến tính. Điều này có những ưu và nhược điểm riêng về hiệu năng:

* Ưu điểm của Danh sách liên kết đôi so với Mảng tuyến tính:
  + Thêm/Xóa phần tử: Các thao tác thêm và xóa ở đầu hoặc cuối danh sách (hoặc khi đã có con trỏ đến vị trí cụ thể) là O(1). Trong mảng tuyến tính, thêm/xóa ở đầu hoặc giữa yêu cầu di chuyển các phần tử, dẫn đến O(N).
  + Kích thước động: Danh sách liên kết có thể mở rộng hoặc thu hẹp linh hoạt theo nhu cầu, không cần cấp phát lại bộ nhớ lớn như mảng khi đầy (như std::vector khi cần resize).
* Nhược điểm của Danh sách liên kết đôi soán với Mảng tuyến tính:
  + Truy cập ngẫu nhiên: Truy cập phần tử tại một vị trí bất kỳ (ví dụ: phần tử thứ k) là O(N) vì phải duyệt từ đầu hoặc cuối. Trong mảng tuyến tính, truy cập là O(1).
  + Chi phí bộ nhớ: Mỗi Node trong danh sách liên kết cần thêm không gian để lưu trữ các con trỏ next và prev, làm tăng chi phí bộ nhớ cho mỗi phần tử so với mảng tuyến tính chỉ lưu trữ dữ liệu.
  + Độ phức tạp tìm kiếm: Việc tìm kiếm một khách hàng hoặc giao dịch dựa trên một tiêu chí (ví dụ: số tài khoản) vẫn là O(N) vì phải duyệt toàn bộ danh sách.

Đánh giá tổng thể:

Chương trình hiện tại có hiệu năng chấp nhận được cho một hệ thống quản lý ngân hàng quy mô nhỏ với lượng dữ liệu không quá lớn. Các thao tác tìm kiếm, cập nhật, rút/gửi tiền và chuyển khoản đều có độ phức tạp O(N) trong trường hợp xấu nhất (khi cần duyệt danh sách khách hàng hoặc giao dịch). Đối với các hệ thống lớn hơn với hàng triệu khách hàng hoặc giao dịch, độ phức tạp O(N) cho các thao tác thường xuyên như tìm kiếm sẽ trở thành một nút thắt cổ chai về hiệu năng.

Để cải thiện hiệu năng cho các hệ thống lớn hơn, nên cân nhắc các cấu trúc dữ liệu khác:

* Đối với tìm kiếm khách hàng theo số tài khoản: Sử dụng Bảng băm (Hash Table) hoặc Cây tìm kiếm nhị phân cân bằng (Balanced Binary Search Tree) có thể giảm độ phức tạp tìm kiếm trung bình xuống O(1) hoặc O(logN).
* Đối với quản lý giao dịch: Duyệt qua tất cả giao dịch là cần thiết cho lịch sử, nhưng việc tìm kiếm giao dịch theo các tiêu chí cụ thể (ví dụ: theo ngày, theo số tài khoản) cũng có thể được tối ưu hóa bằng cách sử dụng các cấu trúc dữ liệu phù hợp.

Tuy nhiên, với việc hiện tại chương trình được xây dựng trên danh sách liên kết, các hoạt động chính như tìm kiếm, thêm, xóa, cập nhật, và thực hiện giao dịch đều phải duyệt qua danh sách, dẫn đến độ phức tạp O(N). Các hoạt động đọc/ghi file cũng có độ phức tạp O(N) do phải duyệt toàn bộ danh sách để ghi hoặc đọc từng mục.