1. Cấu trúc dữ liệu được sử dụng

Chương trình sử dụng vector (mảng động) để lưu trữ:

* vector<KhachHang> danhSachKH - danh sách khách hàng
* vector<GiaoDich> danhSachGD - danh sách giao dịch

2. Phân tích độ phức tạp từng chức năng

Thêm khách hàng - **themKhachHang()**

* Tạo số tài khoản unique: O(n) cho mỗi lần kiểm tra + worst case có thể lặp nhiều lần
* Thêm vào vector: O(1) amortized
* Tổng: O(n) đến O(n²) trong trường hợp xấu

Tìm kiếm khách hàng - **timKhachHang()**

Ảnh có chứa văn bản, ảnh chụp màn hình, Phông chữ

Nội dung do AI tạo ra có thể không chính xác.

* Độ phức tạp: O(n) - tìm kiếm tuyến tính
* Trường hợp tốt nhất: O(1) - tìm thấy ở vị trí đầu
* Trường hợp xấu nhất: O(n) - tìm thấy ở cuối hoặc không tìm thấy

Xóa khách hàng - **xoaKhachHang()**

Ảnh có chứa văn bản, ảnh chụp màn hình, Phông chữ

Nội dung do AI tạo ra có thể không chính xác.

* Tìm kiếm: O(n)
* Xóa phần tử: O(n) - cần shift các phần tử phía sau
* Tổng: O(n)

Cập nhật thông tin - **capNhatThongTin()**

* Tìm kiếm: O(n)
* Cập nhật: O(1)
* Tổng: O(n)

Giao dịch (rút tiền, gửi tiền, chuyển khoản)

Ảnh có chứa ảnh chụp màn hình, văn bản, Phông chữ

Nội dung do AI tạo ra có thể không chính xác.

* Tìm tài khoản: O(n)
* Thực hiện giao dịch: O(1)
* Lưu log giao dịch: O(1)
* Tổng: O(n)

Hiển thị danh sách - hienThiDanhSach()

Ảnh có chứa văn bản, ảnh chụp màn hình, Phông chữ, màu đen

Nội dung do AI tạo ra có thể không chính xác.

* Độ phức tạp: O(n)

File I/O Operations

* Đọc file: O(n)
* Ghi file: O(n)

3. Đánh giá tổng quan

|  |  |
| --- | --- |
| CHỨC NĂNG | ĐỘ PHỨC TẠP |
| Thêm khác hàng | O(n) – O() |
| Tìm khác hàng | O(n) |
| Xóa khách hàng | O(n) |
| Cập nhật thông tin | O(n) |
| Giao dịch | O(n) |
| Hiển thị | O(n) |

4. Kết luận

Hiệu năng hiện tại: Phù hợp cho hệ thống nhỏ (< 1000 khách hàng)

* Với 10,000 khách hàng: mỗi thao tác có thể mất 10,000 so sánh
* Không phù hợp cho hệ thống lớn