**Trường Đại Học Sài Gòn**

**Khoa Công Nghệ Thông Tin**

**---o0o---**

**BÁO CÁO**

**TH 8 – Cài Đặt Cây AVL**

**Họ và Tên: Lương Văn Hay**

**MSSV: 3124411082**

**Giảng viên hướng dẫn: Đỗ Như Tài**

**1. Giới thiệu** Cây AVL là một loại cây tìm kiếm nhị phân tự cân bằng, đảm bảo rằng hiệu số chiều cao giữa hai cây con của bất kỳ nút nào không lớn hơn 1. Điều này giúp cho các thao tác tìm kiếm, thêm, và xóa phần tử có thời gian thực thi trung bình là O(log n), giúp tối ưu hiệu suất so với cây tìm kiếm nhị phân thông thường.

**2. Ý tưởng thực hiện** Chương trình cài đặt cây AVL sử dụng cấu trúc dữ liệu động với các thao tác chính:

**Thêm một nút**: Chèn một giá trị vào cây và thực hiện cân bằng lại nếu cần.

**Xóa một nút**: Xóa một giá trị khỏi cây, sau đó cân bằng lại nếu mất cân bằng.

**Duyệt cây**: Duyệt cây theo thứ tự trung tố (LNR) để hiển thị các phần tử theo thứ tự tăng dần.

Cây AVL duy trì tính cân bằng bằng cách sử dụng các phép quay:

**Quay trái**: Khi cây bị lệch về phía phải.

**Quay phải**: Khi cây bị lệch về phía trái.

**Quay trái-phải**: Khi cây bị lệch về bên trái nhưng cây con trái lại lệch về bên phải.

**Quay phải-trái**: Khi cây bị lệch về bên phải nhưng cây con phải lại lệch về bên trái.

**3. Cách cài đặt** Chương trình được viết bằng ngôn ngữ C++, bao gồm các thành phần chính sau:

**Cấu trúc dữ liệu của một nút trong cây AVL**:

Gồm khóa (giá trị), con trỏ trái/phải và chiều cao của nút.

**Các hàm chính**:

* + layChieuCao(nut): Lấy chiều cao của một nút.
  + layCanBang(nut): Tính hệ số cân bằng của một nút.
  + xoayPhai(nut): Thực hiện phép quay phải.
  + xoayTrai(nut): Thực hiện phép quay trái.
  + chen(nut, khoa): Chèn một giá trị vào cây và thực hiện các phép quay nếu cần.
  + xoa(nut, khoa): Xóa một giá trị khỏi cây và thực hiện cân bằng lại.
  + duyetLNR(nut): Duyệt cây theo thứ tự LNR để in ra các phần tử theo thứ tự tăng dần.

**Chương trình chính (main)**:

Chèn một số phần tử vào cây.

In ra cây theo thứ tự LNR.

Xóa một phần tử và in lại cây.

**4. Kết quả thực thi:** Chương trình được kiểm tra với dữ liệu đầu vào cụ thể, cho kết quả đúng với mong đợi. Cây AVL được xây dựng và duy trì cân bằng sau các thao tác chèn và xóa, đảm bảo hiệu suất O(log n).

**5. Kết luận:** Cây AVL là một trong những phương pháp hiệu quả để quản lý dữ liệu có tính động. Việc cài đặt các phép quay giúp giữ cân bằng cây, đảm bảo hiệu suất tối ưu trong các thao tác tìm kiếm, thêm và xóa. Chương trình đã thể hiện đầy đủ cách thức hoạt động của cây AVL và có thể mở rộng để ứng dụng trong nhiều bài toán thực tế như cơ sở dữ liệu, hệ thống tìm kiếm và quản lý bộ nhớ.