------------------------------------------------------------

BÁO CÁO CÀI ĐẶT CÂY AVL

Chứa các số nguyên (Thêm, Xóa và các thao tác cây BST)

------------------------------------------------------------

1. MỞ ĐẦU

Cây AVL là một dạng cây tìm kiếm nhị phân (BST) tự cân bằng. Mục tiêu chính của cây AVL là đảm bảo chiều cao của cây luôn được tối ưu sau mỗi thao tác chèn hoặc xóa, giúp giảm độ phức tạp của các thao tác như tìm kiếm, chèn và xóa xuống mức O(log n).

2. CẤU TRÚC CÂY AVL

Mỗi nút của cây AVL bao gồm các thành phần chính sau:

- Key (giá trị): Số nguyên lưu giá trị của nút.

- Left: Con trỏ đến cây con bên trái.

- Right: Con trỏ đến cây con bên phải.

- Height: Chiều cao của nút, được dùng để tính hệ số cân bằng của cây.

3. CÁC THAO TÁC CHÍNH

Các thao tác cơ bản trong cây AVL gồm:

- Chèn phần tử (Insert): Thêm một giá trị mới vào cây và cân bằng lại cây nếu cần.

- Xóa phần tử (Remove): Loại bỏ một nút khỏi cây, sau đó cân bằng lại cây.

- Duyệt cây:

- Pre-order (NLR): Duyệt theo thứ tự gốc – trái – phải.

- In-order (LNR): Duyệt theo thứ tự trái – gốc – phải.

- Post-order (LRN): Duyệt theo thứ tự trái – phải – gốc.

- Tìm kiếm phần tử: Kiểm tra sự tồn tại của một giá trị trong cây.

- Tìm giá trị nhỏ nhất và lớn nhất: Duyệt cây theo thứ tự để tìm giá trị nhỏ nhất và lớn nhất.

4. CÂN BẰNG CÂY

Sau mỗi thao tác chèn hoặc xóa, cây cần được cân bằng lại bằng cách tính hệ số cân bằng của mỗi nút:

balance = height(left) - height(right)

Các trường hợp cần xoay:

- Xoay phải (Right Rotation - RR)

- Xoay trái (Left Rotation - LL)

- Xoay trái-phải (Left-Right - LR)

- Xoay phải-trái (Right-Left - RL)

5. MÃ NGUỒN C++ CÀI ĐẶT CÂY AVL

Dưới đây là đoạn mã nguồn C++ được sử dụng để cài đặt cây AVL, bao gồm các thao tác chèn, xóa, và duyệt cây.

------------------------------------------------------------