Full name: Mai Anh Phúc Minh

Student ID: 24120094

Report for week 7.

No.	Percentage understood	Content understood	Percentage Referenced	Content Referenced	Source Referenced
1. binary_tree.cpp	100%				
2. bst.cpp	90%				
3. avl.cpp	x				

Em xin báo cáo về phần bài tập tuần 7, với yêu cầu xử lí cấu trúc dữ liệu cây nhị phân, nhị phân tìm kiếm và avl. Em đã hoàn thành được 2 phần là cây nhị phân và nhị phân tìm kiếm còn về cây avl do chưa hoàn toàn nắm rõ được nên chưa thể hoàn thành.

1. binary_tree.cpp

- createNode(): được xây dựng để tạo một node và gán giá trị là tham số truyền vào. Hai con trỏ trái và phải được gán trị null vì mới được khởi tạo.
- NLR(): được xây dựng để duyệt cây theo pre-order, từ nút sang cây con trái rồi đến cây con phải. Để thực hiện được điều này, em dùng một stack để lưu node mình xét đến, sau đó pop ra và đưa giá trị vào vector kết quả, kế tiếp là truyền vào con phải và con trải vì theo stack là vào sau ra trước. Lặp lại cho đến khi trong stack rỗng rồi trả về vector đã được duyệt.
- LNR(): được xây dựng để duyệt cây theo in-order, từ cây con trái sang nút rồi đến cây con phải. Để thực hiện được điều này, em dùng 1 stack để lưu trữ các node tiến về phía cây con bên trái, tới khi gặp null thì pop ra, cho giá trị vào trong vector kết quả, rồi xét bên phải của node đó. Lặp lại quá trình này cho đến khi stack rỗng thì trả về vector.
- LRN(): được xây dựng để duyệt cây theo post-order, từ cây con trái sang cây con phải rồi mới đến nút. Để thực hiện được điều này, em sử dụng 2 stack đối lập nhau, 1 stack duyệt theo NRL và stack còn tại để reverse cho đúng chiều và đưa vào trong vector kết quả. Em push vào stack 1 giá trị nút trước, sau đó đưa vào vòng lặp để pop nó ra đưa vào stack 2, rồi đưa lần lượt bên trái và bên phải vào. Lặp lại cho đến khi stack 1 rỗng thì em pop các giá trị của stack 2 vào trong vector kết quả.
- levelOrder(): được xây dựng để duyệt theo chiều ngang, theo từng độ cao. Để thực hiện điều này thì em sử dụng queue theo vào trước sẽ ra trước, em push

vào từ trái sang phải của từng tầng, đếm xem phải lặp bao nhiêu lần rồi xét từng nút bên trong đó, pop ra đưa vào vector kết quả rồi lại push vào con trái và con phải. Lặp lại cho đến khi queue trống và trả về vector kết quả.

- countNode(): được xây dựng để đếm số nút trong cây. Em chuyển hóa cây thành 1 vector rồi trả về size của vector đó.
- sumNode(): được xây dựng để tính tổng các giá trị trong cây. Em chuyển hóa cây thành 1 vector, sau đó lặp qua các phần tử và cộng đồn vào một biến, cuối cùng là trả về biến tổng đó.
- heightNode(): được xây dựng để xác định độ cao của nút đó, nếu không tìm thấy hoặc không có cây sẽ trả về -1. Để thực hiện được điều đó, em chuyển hóa cây thành vector 2 chiều duyệt theo chiều ngang, rồi duyệt các tầng xem có tìm thấy được phần tử đó không rồi trả về tầng tương ứng của nó.
- Level(): được xây dựng để xác định cấp bậc của nút đó trong cây, nếu không có cây hoặc không tìm thấy nút sẽ trả về -1. Để thực hiện được điều đó, em chuyển hóa cây thành 1 vector 2 chiều rồi duyệt theo chiều ngược lại so với độ cao và trả về cấp bậc tương ứng.
- countLeaf(): được xây dựng để đếm số lá trong cây. Để thực hiện được điều đó, em duyệt tương tự như pre-order, sau đó xét xem nếu cây con trái và phải của node đó đều là null thì sẽ tính là node lá và cộng vào biến đếm. Sau khi duyệt qua mọi phần tử thì trả về biến đếm.

2. bst.cpp

- search(): được xây dựng để tìm giá trị trong cây. Để thực hiện được điều này, sử dụng tính chất của cây nhị phân tìm kiếm thì ta so sánh node hiện tại, nếu nhỏ hơn thì dịch qua cây con trái, ngược lại dịch sang cây con phải. Khi tìm được thì trả về node đó còn không thì trả về null.
- insert(): được xây dựng để thêm node vào cây. Để thực hiện được điều đó thì ta so sánh giá trị của node, nếu lớn thì dịch phải còn nhỏ thì dịch trái, nếu tới nút lá thì tạo node rồi nối vào cây.
- remove(): được xây dựng dể loại bỏ một nút ra khỏi cây.
- createTree(): được xây dựng để chuyển array thành cây. Để thực hiện được điều đó, ta duyệt qua các phần tử của array và insert vào cây.
- removeTree(): được xây dựng để loại bỏ cả cây. Để thực hiện được điều đó thì em duyệt theo LRN để có được stack sau khi duyệt rồi pop ra để delete.
- height(): được xây dựng để cho biết chiều cao của cây. Em chuyển hóa cây thành vector 2 chiều duyệt theo chiều ngang rồi trả về kích thước của vector đó.

- countLess(): được xây dựng để đếm số phần tử nhỏ hơn giá trị cần xét. Em chuyển cây thành vector duyệt theo in-order, sau đó duyệt qua các phần tử của vector đồng thời đếm xem những giá trị nào thỏa điều kiện. Sau khi duyệt xong thì trả về biến đếm.
- countGreater(): được xây dựng để đếm số phần tử lớn hơn giá trị cần xét. Em làm tương tự với hàm trên nhưng đổi ngược lại điều kiện.
- isBST(): được xây dựng để kiểm tra xem cây có phải cây nhị phân tìm kiếm không. Để thực hiện điều đó em chuyển thành vector duyệt theo in-order rồi kiểm tra xem phần tử trước có nhỏ hơn phần sau không.
- isFullBST(): được xây dựng để kiểm tra cây có đầy chưa. Để thực hiện được điều đó, em duyệt từng nút rồi xét xem nếu chỉ có 1 cây con thì trả về false.

3. Minh chứng Github