Bài tập phần cây tìm kiếm nhị phân

Bài 1. Cho cây tìm kiếm nhị phân như trong hình 1

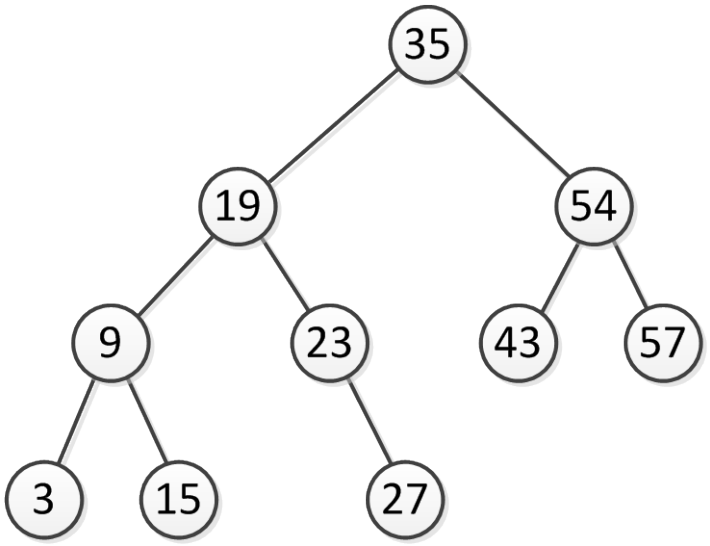

Figure 1. A Binary search tree

Các khóa nào sẽ được so sánh với khóa tìm kiếm khi tìm kiếm khóa 35, 23, 27, 15, 58, 1, 25

Bài 2. Chèn lần lượt các khóa sau vào cây nhị phân tìm kiếm trong hình 1, hãy vẽ cây khết quả thu được sau mỗi lần chèn

1. 12
2. 29
3. 87
4. 5

Bài 3. Vẽ cây nhị phân thu được sau khi xóa khóa sau khỏi cây khung trong hình 1(các nút được xóa riêng rẽ)

1. 15
2. 9
3. 19
4. 35

Bài 4. Vẽ cây nhị phân tìm kiếm thu được sau khi thêm lần lượt các khóa sau vào cây nhị phân ban đầu rỗng

a) Jan Guy Jon Ann Jim Eva Amy Tim Ron Kim Tom Roy Kay Dot
b) Amy Tom Tim Ann Roy Dot Eva Ron Kim Kay Guy Jon Jan Jim
c) 5 23 6 16 78 45 23 22 10 2 34 27
d) 78 34 56 23 98 2 13 5 19 42 11 9

Bài 5. Cây nhị phân tìm kiếm bị suy biến (degenarate) thành cây lệch trái, lệch phải và zig-zag trong trường hợp nào

Bài 6. Trong trường hợp xây dựng cây nhị phân tìm kiếm bằng cách thêm các khóa lần lượt vào cây nhị phân ban đầu rỗng, thì trường hợp nào xây dựng dễ nhất (thực hiện nhanh nhất), trường hợp nào là khó nhất

Bài 7. Với các khóa 22, 41, 25, 34, 65, 27, 9, 36, 45, hãy đưa ra 4 hoán vị của chúng mà đều tạo ra 1 cây tìm kiếm nhị phân giống nhau nếu thực hiện theo cách thêm lần lượt vào cây nhị phân tìm kiếm ban đầu rỗng

Bài 8. Dãy khóa thêm vào ban đầu phải có đặc điểm gì để cây nhị phân tìm kiếm thu được có chiều cao thấp nhất? Cho ví dụ.

Bài 9. Hoàn thiện chương trình minh họa các thao tác trên cây nhị phân tìm kiếm từ các hàm đã có trong slide. Chương trình hoàn thiện cần có các tính năng sau:

- Thêm các khóa mới vào cây từ bàn phím hoặc từ file
- In ra các nút trên cây khi duyệt theo các thứ tự: trước, giữa và sau
- Xóa nút trên cây

Bài 10. Xây dựng 1 hàm để kiểm tra xem 1 cây nhị phân có phải là ây tìm kiếm nhị phân không

Bài 11. Việc sử dụng cây tìm kiếm nhị phân có ưu điểm gì hơn so với sử dụng mảng và thuật toán tìm kiếm nhị phân?

Bài 12. Cho danh sách các khóa đã được sắp xếp theo thứ tự tăng dần, hãy thiết kế thuật toán để xây dựng xây nhị phân tìm kiếm có chiều cao là nhỏ nhất có thể.

Bài 13. Cho một cây nhị phân tìm kiếm, hãy xây dựng hàm để đưa ra tất cả các nút ở mức thứ k trên cây.

Bài 14. Viết hàm để tìm nút kế tiếp của một nút trên cây nhị phân tìm kiếm. Nút kế tiếp là nút tiếp theo nút hiện tại trong tìm duyệt theo thứ tự giữa.

Bài 15. Xây dựng hàm để tìm và trả về nút lá có độ sâu lớn nhất trên cây.

Bài 16. Xây dựng hàm để kiểm tra xem hai cây nhị phân có trùng nhau hay không. Hai cây nhị phân là trùng nhau nếu giá trị của các nút tại các vị trí tương ứng là bằng nhau.

Bài 17. Xây dựng hàm để chuyển đổi một cây nhị phân tìm kiếm thành danh sách liên kết.

Bài 18. Xây dựng hàm để kết hợp 2 cây nhị phân tìm kiếm thành 1 cây nhị phân tìm kiếm?

Bài 19. Giả sử bạn có 1 dãy n các số $x_1, x_2, .., x_n$, và bạn phải trả lời câu hỏi sau một cách nhanh nhất: "cho hai giá trị chỉ số i, j ($1 \leq i \leq j \leq n$) hãy tìm giá trị nhỏ nhất trong khoảng x_i, x_j" .

a) Thiết kế cấu trúc dữ liệu để lưu trữ dãy số để có thể trả lời truy vấn với thời gian cỡ $O(1)$

b) Thiết kế cấu trúc dữ liệu để lưu trữ dãy số để có thể trả lời truy vấn với thời gian cỡ $O(\log n)$

Bài 20. So sánh thuật toán sắp xếp treeSort với thuật toán quicksort

TreeSort giống với QuickSort trong trường hợp chốt chọn là phần tử đầu tiên

Ưu điểm của TreeSort là trong quá trính sắp xếp không cần phải biết thông tin của toán bộ các phần tử của dãy. Do đó treeSort có thể dùng trong trường hợp sắp xếp các dãy mà giá trị của các phần tử trong dãy được cung cấp một cách lần lượt. Ưu điểm nữa của treeSort là kết quả của TreeSort tạo ra một cây nhị phân tìm kiếm, do đó các thao tác thêm, xóa về sau thực hiện dễ dàng hơn.