**Câu 1:**

Cây nhị phân tìm kiếm là cây nhị phân mà giá trị của phần tử bên trái của một node có giá trị nhỏ hơn giá trị của node, phần tử bên phải có giá trị lớn hơn giá trị của node.

Các thao tác thực hiện trong cây nhị phân tìm kiếm là: tìm node, thêm node, xóa node trên cây, duyệt cây nhị phân tìm kiếm theo NLR, LNR, LRN.

Hạn chế: Việc thêm và xóa node trong cây khá phức tạp, thực hiện nhiều phép so sánh, gây tốn bộ nhớ vì phải lưu trữ địa chỉ con trỏ của các nút con bên trái, phải.

**Câu 2: So sánh Cây nhị phân tìm kiếm với các cấu trúc dữ liệu cơ bản như Danh sách đặc, Danh sách liên kết, Danh sách hạn chế.**

* **Giống nhau:**
* Đều thực hiện được các thao tác cơ bản như thêm, chèn, xóa, tìm kiếm, các phép duyệt tương tự các kiểu dữ liệu khác.
* Tương tự như kiểu cấu trúc dữ liệu Danh sách liên kết, Cây nhị phân tìm kiếm cũng có khả năng quản lý một tập các phần tử có số lượng lớn, được cấp phát rời rạc trong bộ nhớ.
* Cây nhị phân tìm kiếm còn có điểm giống với cấu trúc dữ liệu Danh sách liên kết là đều tốn bộ nhớ để lưu trữ các biến con trỏ
* **Khác nhau:**
* Mỗi nút (phần tử) trong Cây nhị phân tìm kiếm ngoài thành phần lưu trữ giá trị của nút (info), còn có hai thành phần kiểu địa chỉ bộ nhớ dùng để lưu trữ địa chỉ của nút con bên trái (left) và lưu địa chỉ nút con bên phải (right). Trong đó, *cây con bên trái* và *cây con bên phải* luôn tuân theo điều kiện **left < info < right.**
* Xét về tốc độ thực hiện thao tác, thì Cây nhị phân tìm kiếm có *khả năng tìm kiếm nhanh* hơn các kiểu dữ liệu khác. Bởi vì tính chất, giá trị của một nút sẽ lớn hơn các giá trị của nhánh con bên phải và nhỏ hơn các giá trị của nhánh con bên phải, nên việc tìm kiếm trên Cây nhị phân tìm kiếm sẽ tiết kiểm được nhiều thời gian dễ dàng hơn. Bên cạnh đó, thì việc thực hiện thao tác *thêm/xóa nút trong cây* lại tốn khá nhiều thời gian và khá phức tạp vì phải thực hiện nhiều phép so sánh.