Câu 1: Hãy trình bày các vấn đề sau: Định nghĩa và đặc điểm của cây nhị phân tìm kiếm; các thao tác thực hiện tốt trong kiểu này; Hạn chế của CTDL này?

Định nghĩa: cây nhị phân tìm kiếm là cây nhị phân mà giá trị (khóa) của các phần tử bên trái của một node có giá trị nhỏ hơn giá trị (khóa) của node, giá trị (khóa) của các phần tử bên phải của một node có giá trị lớn hơn giá trị (khóa) của node đó.

Đặc điểm: mỗi node trong cây nhị phân tìm kiếm ngoài thành phần lưu trữ giá trị node (info) còn có 2 thành phần kiểu địa chỉ bộ nhớ dung để lưu trữ địa chỉ của node con bên trái (left) và node con bên phải (right). Cấu trúc cây nhị phân tìm kiếm quản lý một tập các phần tử có số lượng khá lớn, được cấp phát rời rạc trong bộ nhớ.

Các thao tác trong cây nhị phân tìm kiếm:

* Tìm một node trên cây nhị phân tìm kiếm
* Thêm một node mới vào cây
* Duyệt cây nhị phân tìm kiếm (LRN, LNR, NLR)
* Xóa một node trên cây

Hạn chế: Cấu trúc cây nhị phân tìm kiếm quản lý một tập các phần tử có số lượng khá lớn; việc thêm/xóa node trong cây khá phức tạp, do phải thực hiện nhiều pháp so sánh.

Câu 2: Hãy so sánh cây nhị phân tìm kiếm và các CTDL cơ bản: danh sách đặc, danh sách liên kết, danh sách hạn chế.

Cây nhị phân tìm kiếm :

Độ phức tạp tính toán của thuật toán tìm kiếm nhị phân trong trường hợp tốt nhất là O(1), trong trường hợp xấu nhất là O(log2n) và trong trường hợp trung bình cũng là O(log2n). Tuy nhiên, ta không nên quên rằng trước khi sử dụng tìm kiếm nhị phân, dãy khoá phải được sắp xếp rồi, tức là thời gian chi phí cho việc sắp xếp cũng phải tính đến. Nếu dãy khoá luôn luôn biến động bởi phép bổ sung hay loại bớt đi thì lúc đó chi phí cho sắp xếp lại nổi lên rất rõ làm bộc lộ nhược điểm của phương pháp này.

Danh sách đặc và danh sách liên kết :

Không thể kết luận phương pháp cài đặt nào hiệu quả hơn, mà nó hoàn toàn tuỳ thuộc vào từng ứng dụng hay tuỳ thuộc vào các phép toán trên danh sách. Tuy nhiên ta có thể tổng kết một số ưu nhược điểm của từng phương pháp làm cơ sở để lựa chọn phương pháp cài đặt thích hợp cho từng ứng dụng:

Cài đặt bằng mảng đòi hỏi phải xác định số phần tử của mảng, do đó nếu không thể ước lượng được số phần tử trong danh sách thì khó áp dụng cách cài đặt này một cách hiệu quả vì nếu khai báo thiếu chỗ thì mảng thường xuyên bị đầy, không thể làm việc được còn nếu khai báo quá thừa thì lãng phí bộ nhớ.

Cài đặt bằng con trỏ thích hợp cho sự biến động của danh sách, danh sách có thể rỗng hoặc lớn tuỳ ý chỉ phụ thuộc vào bộ nhớ tối đa của máy. Tuy nhiên ta phải tốn thêm vùng nhớ cho các con trỏ (trường next).

Cài đặt bằng mảng thì thời gian xen hoặc xoá một phần tử tỉ lệ với số phần tử đi sau vị trí xen/ xóa. Trong khi cài đặt bằng con trỏ các phép toán này mất chỉ một hằng thời gian.

Phép truy nhập vào một phần tử trong danh sách, chẳng hạn như PREVIOUS, chỉ tốn một hằng thời gian đối với cài đặt bằng mảng, trong khi đối với danh sách cài đặt bằng con trỏ ta phải tìm từ đầu danh sách cho đến vị trí trước vị trí của phần tử hiện hành.Nói chung danh sách liên kết thích hợp với danh sách có nhiều biến động, tức là ta thường xuyên thêm, xoá các phần tử.

Danh sách hạn chế

Hàng đợi, hay ngắn gọn là hàng (queue) cũng là một danh sách đặc biệt mà phép thêm và bớt được thực hiện ở 2 đầu khác nhau. Thêm ở cuối danh sách (REAR), còn bớt thì ở phía đầu của danh sách (FRONT).

Ngăn xếp (Stack) là một danh sách mà ta giới hạn việc thêm vào hoặc loại bỏ một phần tử chỉ thực hiện tại một đầu của danh sách, đầu này gọi là đỉnh (TOP) của ngăn xếp. Một chồng đĩa đặt trên bàn cho ta hình ảnh trực quan của ngăn xếp, muốn thêm vào chồng đó 1 đĩa ta để đĩa mới trên đỉnh chồng, muốn lấy các đĩa ra khỏi chồng ta cũng phải lấy đĩa trên trước. Như vậy ngăn xếp là một cấu trúc có tính chất "vào sau - ra trước" hay LIFO (last in - first out ).