BÀI 14: CẤU TRÚC DỮ LIỆU CÂY NHỊ PHÂN

Mục tiêu:

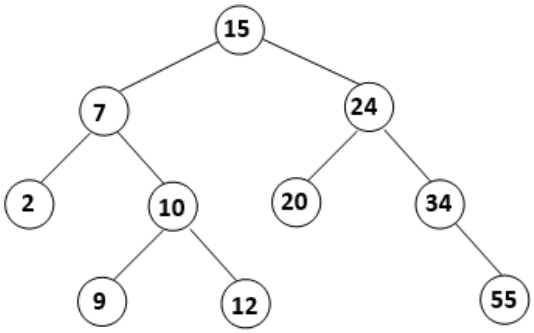
Hoàn tất bài thực hành, sinh viên có khả năng:

- Tạo được 1 cây nhị phân tìm kiếm.
- Thực hiện được các thao tác cơ bản trên cây nhị phân tìm kiếm: Duyệt cây, tìm kiếm một nút trên cây, thêm một nút vào cây, xóa một nút v.v…
- Sử dụng được cấu trúc cây nhị phân tìm kiếm để tổ chức lưu trữ dữ liệu cho một bài toán đơn giản và giải quyết bài toán đó dựa trên các thao tác cơ bản.

Tóm tắt nội dung bài học

Cây nhị phân tìm kiếm (CNPTK) là cây nhị phân hoặc rỗng hoặc không rỗng thì phải thoả mãn đồng thời các điều kiện sau:

- Khoá của các nút thuộc cây con trái nhỏ hơn khoá nút gốc
- Khoá của nút gốc nhỏ hơn khoá của các nút thuộc cây con phải của nút gốc
- Cây con trái và cây con phải của gốc cũng là cây nhị phân tìm kiếm



Hình 1. Minh họa Cây nhị phân tìm kiếm

Các thao tác cơ bản trên cây nhị phân tìm kiếm:

Thêm nút: Việc thêm một nút mới vào cây sẽ tạo ra cây mới và cây mới phải đảm bảo điều kiện ràng buộc của CNPTK.

Giải thuật:

1. Bắt đầu đi từ nút gốc. Nếu nút gốc là NULL (cây rỗng) thì vị trí cần chèn nút mới chính là gốc cây và xuống bước 3. Ngược lại ta xuống bước 2.
2. Việc đi tìm vị trí thêm nút là cách chúng ta đi xét từ nút gốc rồi xuống cây trái hay cây phải của nó. Quá trình đi tìm dừng lại khi ta tìm được nút trái hay phải của 1 nút nào đó là NULL. Khi dừng lại thì nút mới sẽ được thêm vào chính nút trái (phải) của nút đó và xuống bước 3.
3. Kết thúc quá trình thêm nút.

Lưu ý:

1. Nút mới sau khi chèn thường nằm ở nút lá.

2. Quá trình tạo cây chính là quá trình thêm lần lượt các nút mới vào 1 cây rỗng ban đầu.

Xóa nút:

Chúng ta cần phải loại bỏ khỏi CNPTK một nút có khoá X (ta gọi tắt là nút X) cho trước, sao cho việc huỷ một nút ra khỏi cây thì cây mới tạo được cũng phải bảo đảm điều kiện ràng buộc của CNPTK.

Có ba trường hợp khi huỷ một nút X có thể xảy ra:

1. X là nút lá

2. X là nút nửa lá (chỉ có một con trái hoặc con phải)

3. X có đủ hai con (trường hợp tổng quát)

Duyệt cây:

Là việc ta đi tới tất cả các nút trên CNPTK. Có 3 cách thực hiện: duyệt theo thứ tự trước (NLR), duyệt theo thứ tự giữa (LNR), duyệt theo thứ tự sau (LRN). Lưu ý, do tính chất của CNPTK, phép duyệt giữa cho phép duyệt các khóa của nút trên cây theo thứ tự tăng dần.

NỘI DUNG THỰC HÀNH

Hãy viết chương trình thực hiện các yêu cầu sau:

1. Tạo 1 cây nhị phân tìm kiếm.

2. Thêm các nút chứa khóa là các giá trị nguyên vào cây cho tới khi khóa nhập vào là -1 (chú ý: -1 không phải là khóa của nút trên cây).

3. Xóa 1 nút có khóa bằng giá trị X cho trước khỏi cây vừa tạo.

4. In các nút trên cây ra màn hình bằng cách duyệt theo thứ tự trước (NLR).

Gợi ý khai báo cấu trúc dữ liệuh và các hàm thực hiện các yêu cầu của bài toán:

//Khai báo cấu trúc dữ liệu cây nhị phân tìm kiếm

```
struct NODE{
    int infor;
    NODE *Left, *Right;
};
NODE *ROOT;
```

//Khởi tạo 1 cây rỗng

```
void Khoitao(NODE *&Root)
{
    Root = NULL;
}
```

//Tạo 1 cây với dữ liệu nhập từ bàn phím

```
void TaoCay(NODE *&Root) {
    int Data;
    do{
       cout<<"Nhap so nguyen, ket thuc nhap -1: ";
       cin>>Data;
       if (Data == -1)
           break;
       Chen(Root, Data);
    } while(1);
}
```

//Chèn 1 nút có giá trị x lên cây

```
void ThemNut(NODE *&Root, int x)
{
    if (Root == NULL)
    {
        NODE *q;
        q = new NODE;
        q->infor = x;
        q->Left = q->Right = NULL;
        Root = q;
    }
    else
    {
        if (x < Root->infor)
            Chen (Root->Left, x);
        else if (x > Root->infor)
            Chen (Root->Right, x);
    }
}
```

//Xóa 1 nút có giá trị là x trên cây:

```
void TimNut (NODE* &Root, NODE* &q) {
    if (Root->Right!=NULL)
        DuyetCay(Root->Right,q);
    else {
        q->infor = Root->infor;
        q = Root;
        Root = Root->Left;
    }
}
```

```
void XoaNut(NODE* &Root, int x) {
    NODE* p;
    if (Root == NULL)
        cout<<x<<" khong co trong cay";
    else {
        if (x < Root->infor)
            XoaNut(Root->Left,x);
        else
            if (x > Root->infor)
                XoaNut(Root->Right,x);
            else {
```
//ghi chú: đây là trường hợp nút bị xóa có đủ cả 2 con trái và phải
```
                p = Root;
                if (p->Right == NULL)
                    Root = p->Left;
                else
                    if (p->Left == NULL)
                        Root = p->Right;
                    else {
                        TimNut(Root->Left,p);
                    }
                delete p;
            }
    }
}
```
// In các nút trên cây ra màn hình theo thứ tự trước (NLR)
```
void NLR(NODE*Root) {
    if (Root != NULL)
    {
        printf("%4d", Root->infor);
        NLR(Root->Left);
        NLR(Root->Right);
    }
}
```

Bài tập cơ bản

1. Hãy giải thích cụ thể ý nghĩa của các hàm trên.

2. Hãy viết một chương trình hoàn chỉnh dựa trên các hàm mẫu để giải quyết bài tập đã cho.

3. Hãy để ý tới hàm XoaNut(), với trường hợp nút bị xóa có 2 con thì có 2 cách để thay thế nút bị xóa này: Hoặc là được thay thế bằng nút trái cùng của cây con phải hoặc là nút phải cùng của cây con trái. Hãy cho biết trong hàm Xoanut() trên thì cách thay thế nào được sử dụng? Vì sao?

4. Tiếp theo câu hỏi 3: Hãy viết lại hàm XoaNut() trên, trong đó sử dụng cách thay thế còn lại?

5. Kết quả in ra sẽ thế nào khi ta viết lại hàm NLR() bên trên thành hàm dưới đây?

```
void NLR(NODE*Root) {
    if (Root != NULL) {
        NLR(Root->Left);
        NLR(Root->Right);
        printf("%4d", Root->infor);
    }
}
```

6. Hãy viết lại chương trình trên bằng cách khử đệ quy, rồi chạy lại và so sánh kết quả với phương pháp đệ quy.

7. Hãy cho biết CNPTK có độ cao bao nhiêu?

8. Hãy cho biết số lượng các nút trên CNPTK.

9. Hãy cho biết số lượng các nút lá trên CNPTK.

10. Hãy cho biết có bao nhiêu nút trên CNPTK chứa giá trị là số nguyên tố? hãy tính tổng tất cả các số nguyên tố này.

11. Hãy cho biết có bao nhiêu nút trên CNPTK chứa giá trị là số chính phương? hãy tính tổng tất cả các số chính phương này.

12. Hãy viết hàm tìm giá trị nhỏ nhất và lớn nhất trên CNPTK.

BÀI TẬP VỀ NHÀ

1. Cho 1 đoạn văn bản không dấu. Viết chương trình thực hiện các yêu cầu sau:

a) Hãy tổ chức để lưu trữ các từ trong văn bản đó bằng cách sử dụng cấu trúc CNPTK.

b) Dựa vào CNPTK hãy cho biết đoạn văn bản có bao nhiêu từ khác nhau và mỗi từ xuất hiện bao nhiêu lần trong văn bản.

c) Nhập vào 1 từ. Hãy kiểm tra xem từ đó có trong văn bản hay không dựa vào CNPTK.

2. Bài toán: Cho dãy khóa x = {50, 40, 60, 10, 20, 30, 10, 15, 65, 55, 45, 58}

- Dựng cây nhị phân tìm kiếm với các nút trên cây được lấy từ dãy khóa x theo thứ tự trái sang phải. Vẽ hình ảnh cây sau khi dựng. Thiết kế giải thuật dựng cây.
- Cho biết thứ tự các khóa được thăm khi duyệt cây theo thứ tự trước.
- Cho biết thứ tự các khóa được thăm khi duyệt cây theo thứ tự giữa.
- Cho biết thứ tự các khóa được thăm khi duyệt cây theo thứ tự sau.
- Vẽ hình mô tả thao tác chèn thêm nút có khóa 35 vào cây.
- Mô tả thao tác tìm nút có khóa 48 trên cây.
- Mô tả thao tác tìm cha của nút có khóa 58 trên cây.
- Mô tả thao tác xóa nút có khóa 55 trên cây.

Yêu cầu:

- Khai báo cấu trúc dữ liệu của cây nhị phân tìm kiếm.
- Cài đặt chương trình ứng dụng với các chức năng biểu diễn các thao tác được cho trong bài toán.
- Lưu ý, sau mỗi thao tác hãy duyệt và hiển thị lại cây.