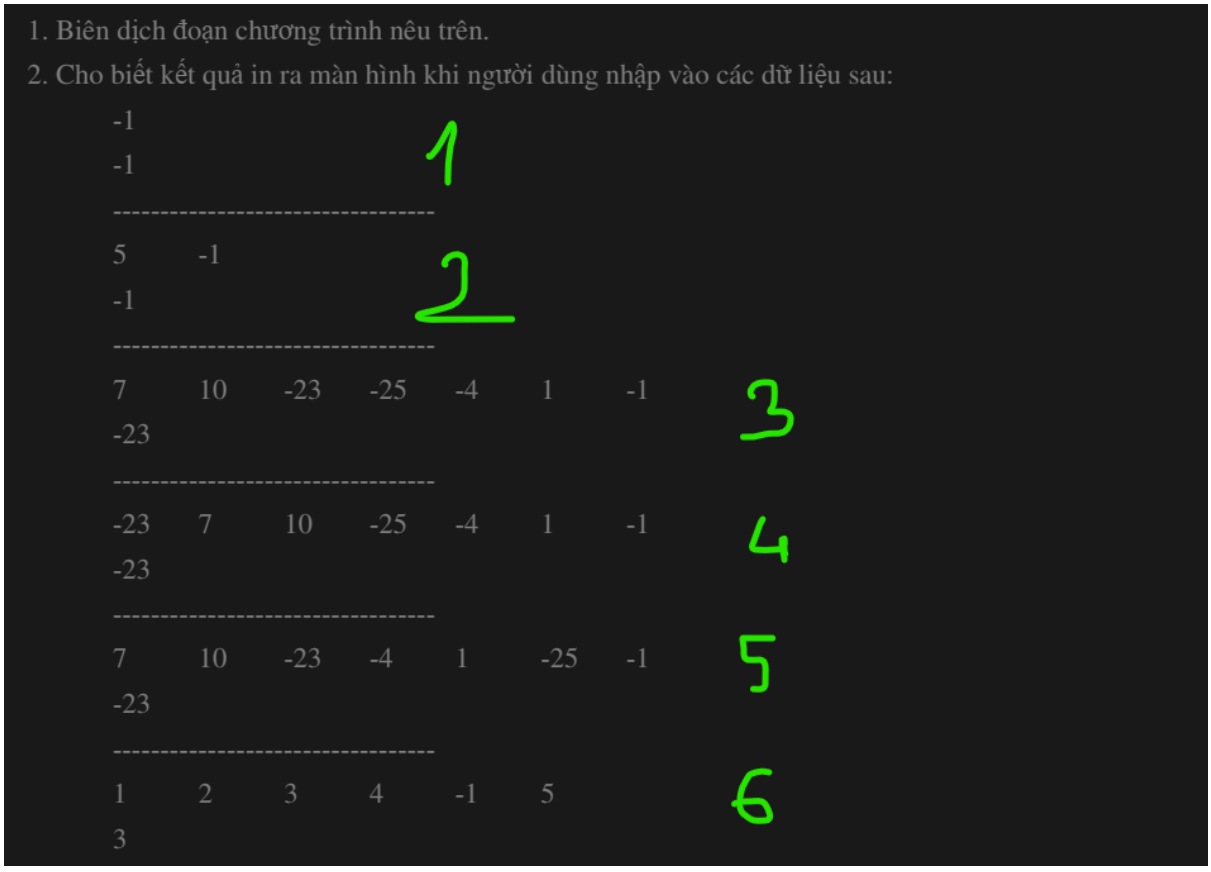
Cây Nhị Phân Tìm Kiếm

MSSV	20280083
Tên	Lại Toàn Thắng



Người dùng nhập vào các dữ liệu trên, kết quả được in ra:

1. -1 không có trong cây
2. 5
   -1 không có trong cây
3. 7 10 -23 -25 -4 1 10
   23 co xuat hien trong cay.

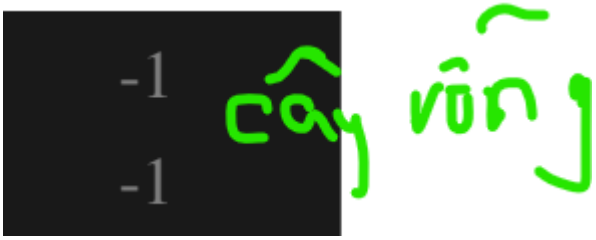
Chieu cao cua nut -23 la 3
7 -25 -4 1 10

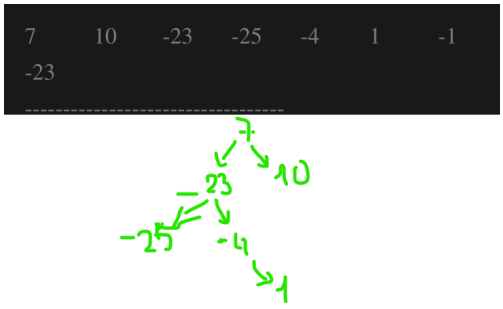
4. -23 -25 7 -4 1 10
   -23 co xuat hien trong cay.
   Chieu cao cua nut -23 la 4
   -25 7 -4 1 10
5. 7 -23 -25 -4 1 10
   -23 co xuat hien trong cay.
   Chieu cao cua nut -23 la 3
   7 -25 -4 1 10
6. 1 2 3 4
   5 khong co trong cay.

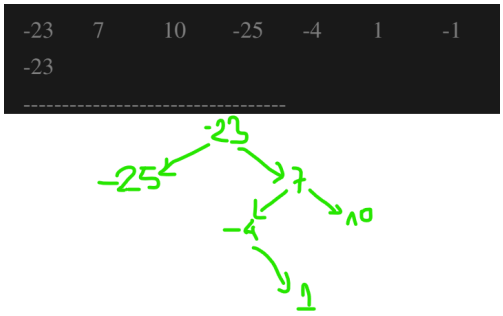
3. Nêu nhận xét ngắn gọn mối liên hệ giữa thứ tự nhập dữ liệu vào (với cùng tập dữ liệu – dữ liệu thứ 3, 4, và 5) với thứ tự in dữ liệu ra màn hình.

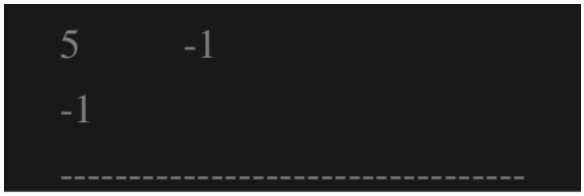
Nhận xét: thứ tự in dữ liệu ra màn hình theo phương thức duyệt cây theo mức (LNR), tức là nó sẽ in node root, xong đến node left và sang right. (Dữ liệu đã được insert vào vị trí của nó theo cấu trúc của cây nhị phân - nhỏ hơn thì qua trái và lớn hơn thì qua phải)

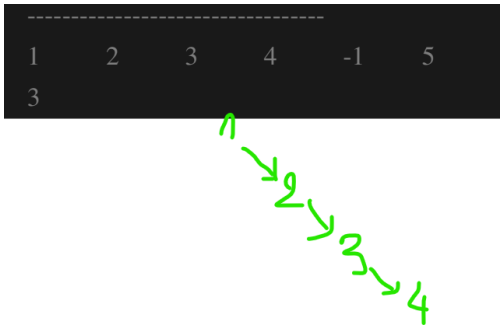
4. Vẽ hình cây nhị phân tìm kiếm theo dữ liệu được nhập ở câu 2.

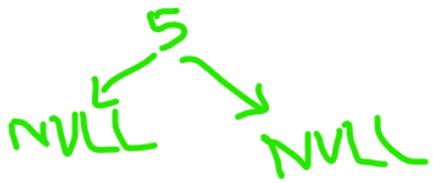


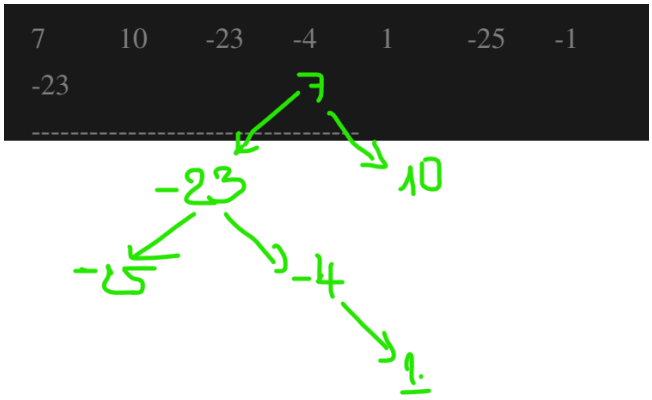












5. Nếu bỏ `Init(pTree)` trong hàm `main` kết quả có thay đổi hay không? Giải thích lý do?

Nếu bỏ hàm Init (pTree) trong hàm main thì kết quả có bị thay đổi. Cụ thể hơn là chương trình sẽ gặp lỗi. Lý do là bởi vì chưa cho cây bằng NULL → Cây vẫn chưa rỗng. → Cây nhị phân không hợp lệ

Khi vào hàm Insert, mình phải xét xem tree pointer (root) có phải là nullptr, sau đó mới xem đến left child và right child.

6. Nếu trong hàm `CreateTree` vòng lặp `do...while` được thay đổi như dưới đây thì kết quả kết xuất ra màn hình sẽ như thế nào đối với dữ liệu câu 2? Giải thích lý do?

```
do
{
	printf("Nhap vao du lieu, -1 de ket thuc: ");
	scanf("%d", &Data);
	Insert(pRoot, Data);
	if (Data == -1)
		break;
}while (1);
```

Nếu thay đổi code như trên thì kết quả xuất ra màn hình sẽ có thêm -1. Vì ở dòng Insert xuất hiện trước dòng điều kiện If. Khi nhập vào -1 thì sẽ Insert trước rồi mới break ra khỏi vòng lặp

7. Trong hàm NLR nếu ta đổi trật tự như bên dưới thì kết quả thế nào?

```
void NLR(NODE* pTree)
{
	if(pTree != NULL)
	{
		NLR(pTree->pLeft);
		printf("%4d", pTree->Key);
		NLR(pTree->pRight);
```

Ta được kết quả là một mảng tăng dần các giá trị.

Xét trong dòng điều kiện if, pTree sẽ là node left tận cùng của cây. (khác null)

Sau đó in node left đấy ra màn hình.

Dòng kế tiếp là di chuyển pTree sang pRight.

Vì NLR là hàm đệ quy nên nó sẽ ngược lại. pTree sẽ di chuyển đến pLeft ở vòng lặp trước (đệ quy ngược)... Tiếp tục làm như vậy ta được một mảng tăng dần các giá trị

8. Hãy ghi chú các thông tin bằng cách trả lời các câu hỏi ứng với các dòng lệnh có yêu cầu ghi chú (//Ghi chú) trong các hàm Search, RemoveNode.

9. Nếu trong hàm RemoveNode thay đổi như sau, kết quả có thay đổi không? Nếu có, chỉ ra cách để kết quả không thay đổi. Nếu không, giải thích lý do

```
else {
	//Ghi chú: Hàm bên dưới dùng để làm gì?
	SearchStandFor(Tree->pRight,p);
}
```

Kết quả có bị thay đổi.

```
else {
    // Neu Tree->pRight thi dung cach 2: THAY THE BANG PHAN TU
    // NHO NHAT TRONG CAY CON PHAI
    SearchStandFor(Tree->pRight,p);
}
```

Để kết quả không thay đôi thì cần chỉnh sửa xíu về code.

```
void SearchStandFor(NODE* &Tree, NODE* &q)
{
    if (Tree->pLeft)
        SearchStandFor(Tree->pLeft,q);
    else
    {
        q->Key = Tree->Key;
        q = Tree;
        Tree = Tree->pRight;
    }
}
```

10. Trong hàm `RemoveNode` nếu không có dòng `delete p;` thì kết quả có gì khác? Dòng đó dùng để làm gì.

Nếu không có dòng delete p: thì kết quả không có gì khác. Nhưng vẫn nên có dòng đó vì trong hàm mình đã cấp phát động cho p thì cuối hàm phải delete nó để tránh các trường hợp trong bộ nhớ bị trùng địa chỉ.