BINARY SEARCH TREE

Binary search tree (BST) is a popular data structure for storing items when fast search is a desired functionality. Its central property is that the nodes in the tree are completely sorted. In order to achieve this, BSTs impose the following property: The key in each node must be greater than any key stored in the left sub-tree, and less than any key stored in the right sub-tree.

The node in the BST can be represented as follows:

```
struct NODE {
        int key;
        struct NODE *left;
        struct NODE *right;
        struct NODE *parent;
}
```

Your task is to implement a binary search tree using pointer, and finally print out the node values in ascending order together with its parent using the following piece of code

```
void printBST(struct Node *root)
{
    if (root != NULL)
    {
        printBST(root->left);
        printf("Node: %d, ", root->key);
        if (root->parent == NULL)
          printf("Parent: NULL\n");
        else
          printf("Parent: %d\n", root->parent->key);
        printBST(root->right);
    }
}
```

The input contains one line of integer values that is going to be inserted to the tree. The values are separated by a single whitespace and there will be no duplicate value.

The output contains several lines. Each line will follow the format: "Node: <value of the node>, Parent: <value of the node's parent>"

SAMPLE INPUT
12 50 5 10 9 20 7 6

SAMPLE OUTPUT
Node: 5, Parent: 12
Node: 6, Parent: 7
Node: 7, Parent: 9

```
Node: 9, Parent: 10
Node: 10, Parent: 5
Node: 12, Parent: NULL
Node: 20, Parent: 50
Node: 50, Parent: 12
```