```python
import queue
class BinaryTreeNode:
    def __init__(self, data):
        self.data = data
        self.left = None
        self.right = None

    # Time complexity: O(H)
    # Space complexity: O(H)
    # where H is the height of the input BST
def findPathBST(root,data):
    if root is None:
        return None
    #if data is found at the current node
    if data == root.data:
        output = []
        output.append(root.data)
        return output
    #if data is smaller, check in left sub-tree
    elif data < root.data:
        output = findPathBST(root.left,data)
        if output is not None:
            output.append(root.data)
        return output
    #if data is bigger, check in right sub-tree
    else:
        output = findPathBST(root.right,data)
        if output is not None:
            output.append(root.data)
        return output



def buildLevelTree(levelorder):
    index = 0
    length = len(levelorder)
    if length<=0 or levelorder[0]==-1:
        return None
    root = BinaryTreeNode(levelorder[index])
    index += 1
    q = queue.Queue()
    q.put(root)
    while not q.empty():
        currentNode = q.get()
        leftChild = levelorder[index]
        index += 1
        if leftChild != -1:
            leftNode = BinaryTreeNode(leftChild)
            currentNode.left =leftNode
            q.put(leftNode)
        rightChild = levelorder[index]
        index += 1
        if rightChild != -1:
            rightNode = BinaryTreeNode(rightChild)
            currentNode.right =rightNode
            q.put(rightNode)
    return root

# Main
levelOrder = [int(i) for i in input().strip().split()]
root = buildLevelTree(levelOrder)
data = int(input())
path = findPathBST(root,data)
if path is not None:
    for ele in path:
        print(ele,end=' ')
```