## Lowest Common Ancestor in Binary Tree

**Algorithm:**

- Create a recursive function that takes a node and the two values n1 and n2.
- If the value of the current node is less than both n1 and n2, then LCA lies in the right subtree. Call the recursive function for the right subtree.
- If the value of the current node is greater than both n1 and n2, then LCA lies in the left subtree. Call the recursive function for the left subtree.
- If both the above cases are false, then return the current node as LCA.

```
#include <stdio.h>
#include <stdlib.h>
struct node
{
  int data;
  struct node *left, *right;
};

//Write your code here
struct node *newNode (int data)
{
  struct node *node = (struct node *) malloc (sizeof (struct node));
  node->data = data;
  node->left = node->right = NULL;
  return (node);
}
int main ()
{
  struct node *root = newNode (20);
  root->left = newNode (8);
  root->right = newNode (22);
  root->left->left = newNode (4);
  root->left->right = newNode (12);
  root->left->right->left = newNode (10);
  root->left->right->right = newNode (14);
  int n1 = 10, n2 = 14;
  struct node *t = lca (root, n1, n2);
  printf ("LCA of %d and %d is %d \n", n1, n2, t->data);
  n1 = 14, n2 = 8;
  t = lca (root, n1, n2);
  printf ("LCA of %d and %d is %d \n", n1, n2, t->data);
  n1 = 10, n2 = 22;
  t = lca (root, n1, n2);
  printf ("LCA of %d and %d is %d \n", n1, n2, t->data);
  getchar ();
  return 0;
}
```