Complete the below AVL tree program accordingly.

```
// C program to Insert a node in Avl tree
#include <stdio.h>
#include <stdlib.h>

// An AVL tree node
struct Node
{
        int key;
        struct Node *left;
        struct Node *right;
        int height;
};

// A function to get maximum of two integers

int max(int a, int b) {

  return (a > b) ? a : b;

}

// A function to get the height of the tree

int height(struct Node *p) {

  if (p == NULL) return 0;
  return p->height;

}

/* Helper function that allocates a new node with the
given key and NULL left and right pointers. */

struct Node *newNode(int key) {

  struct Node *node = (struct Node *)
  malloc(sizeof(struct Node));

  node->key = key;
  node->left = NULL;
  node->right = NULL;
  node->height = 1;

  return (node);

}
```

```
//A utility function to right rotate subtree rooted with y

struct Node *rightRotate(struct Node *y)
{

  struct Node *x = y->left;
  struct Node *T2 = x->right;

//Perform rotation

  x->right = y;
  y->left = T2;

//Update heights

  y->height = max(height(y->left), height(y->right)) + 1;
  x->height = max(height(x->left), height(x->right)) + 1;

//Return new root

  return x;

}

//A utility function to left rotate subtree rooted with x

struct Node *leftRotate(struct Node *x)
{
  struct Node *y = x->right;
  struct Node *T2 = y->left;

//Perform rotation

  y->left = x;
  x->right = T2;

//Update heights

  x->height = max(height(x->left), height(x->right)) + 1;
  y->height = max(height(y->left), height(y->right)) + 1;

//Return new root

  return y;
}
```