#include <stdio.h>

#include <stdlib.h>

struct Node

{

int key;

struct Node \*left;

struct Node \*right;

int height;

};

int getHeight(struct Node \*n){

if(n==NULL)

return 0;

return n->height;

}

struct Node \*createNode(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;

}

int max (int a, int b){

return (a>b)?a:b;

}

int getBalanceFactor(struct Node \* n){

if(n==NULL){

return 0;

}

return getHeight(n->left) - getHeight(n->right);

}

struct Node\* rightRotate(struct Node\* y){

struct Node\* x = y->left;

struct Node\* T2 = x->right;

x->right = y;

y->left = T2;

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

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

return x;

}

struct Node\* leftRotate(struct Node\* x){

struct Node\* y = x->right;

struct Node\* T2 = y->left;

y->left = x;

x->right = T2;

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

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

return y;

}

struct Node \*insert(struct Node\* node, int key){

if (node == NULL)

return createNode(key);

if (key < node->key)

node->left = insert(node->left, key);

else if (key > node->key)

node->right = insert(node->right, key);

node->height = 1 + max(getHeight(node->left), getHeight(node->right));

int bf = getBalanceFactor(node);

// Left Left Case

if(bf>1 && key < node->left->key){

return rightRotate(node);

}

// Right Right Case

if(bf<-1 && key > node->right->key){

return leftRotate(node);

}

// Left Right Case

if(bf>1 && key > node->left->key){

node->left = leftRotate(node->left);

return rightRotate(node);

}

// Right Left Case

if(bf<-1 && key < node->right->key){

node->right = rightRotate(node->right);

return leftRotate(node);

}

return node;

}

void preOrder(struct Node \*root)

{

if(root != NULL)

{

printf("%d ", root->key);

preOrder(root->left);

preOrder(root->right);

}

}

int main(){

struct Node \* root = NULL;

root = insert(root, 1);

root = insert(root, 2);

root = insert(root, 4);

root = insert(root, 5);

root = insert(root, 6);

root = insert(root, 3);

preOrder(root);

return 0;

}

4 2 1 3 5 6

--------------------------------

Process exited after 0.03014 seconds with return value 0

Press any key to continue . . .