**Red Black Tree**

#include<stdio.h>

#include<stdlib.h>

#define RED 'R'

#define BLACK 'B'

struct Node

{

int data;

char color;

struct Node \*left,\*right,\*parent;

};

struct Node \*root=NULL;

void leftRotate(struct Node \*x);

void rightRotate(struct Node \*y);

void color(struct Node \*z);

void insert(int val);

void printTree(struct Node \*root,int level,int space);

void leftRotate(struct Node \*x)

{

struct Node \*y=x->right;

x->right=y->left;

if(y->left!=NULL)

y->left->parent=x;

y->parent=x->parent;

if(x->parent==NULL)

root=y;

else if(x==x->parent->left)

x->parent->left=y;

else

x->parent->right=y;

y->left=x;

x->parent=y;

}

void rightRotate(struct Node \*y)

{

struct Node \*x=y->left;

y->left=x->right;

if(x->right!=NULL)

x->right->parent=y;

x->parent=y->parent;

if(y->parent==NULL)

root=x;

else if(y==y->parent->left)

y->parent->left=x;

else

y->parent->right=x;

x->right=y;

y->parent=x;

}

void color(struct Node \*z)

{

while(z->parent!=NULL&&z->parent->color==RED)

{

if(z->parent==z->parent->parent->left)

{

struct Node \*y=z->parent->parent->right;

if(y!=NULL&&y->color==RED)

{

z->parent->color=BLACK;

y->color=BLACK;

z->parent->parent->color=RED;

z=z->parent->parent;

}

else

{

if(z==z->parent->right)

{

z=z->parent;

leftRotate(z);

}

z->parent->color=BLACK;

z->parent->parent->color=RED;

rightRotate(z->parent->parent);

}

}

else

{

struct Node \*y=z->parent->parent->left;

if(y!=NULL&&y->color==RED)

{

z->parent->color=BLACK;

y->color=BLACK;

z->parent->parent->color=RED;

z=z->parent->parent;

}

else

{

if(z==z->parent->left)

{

z=z->parent;

rightRotate(z);

}

z->parent->color=BLACK;

z->parent->parent->color=RED;

leftRotate(z->parent->parent);

}

}

}

root->color=BLACK;

}

void insert(int val)

{

struct Node \*z=(struct Node \*)malloc(sizeof(struct Node));

struct Node \*y=NULL;

struct Node \*x=root;

z->data=val;

z->left=z->right=z->parent=NULL;

z->color=RED;

while(x!=NULL)

{

y=x;

if(z->data<x->data)

x=x->left;

else

x=x->right;

}

z->parent=y;

if(y==NULL)

root=z;

else if(z->data<y->data)

y->left=z;

else

y->right=z;

color(z);

}

void printTree(struct Node \*root,int level,int space)

{

int i;

if(root==NULL)

return;

space+=5;

printTree(root->right,level+1,space);

printf("\n");

for(i=5;i<space;i++)

printf(" ");

printf("%d%c\n",root->data,root->color);

printTree(root->left,level+1,space);

}

void main()

{

int ch ,val;

while(1)

{

printf("\n1.Insert\n2.Display\n3.Exit\nEnter Choice:");

scanf("%d",&ch);

switch(ch)

{

case 1:

printf("Enter element:");

scanf("%d",&val);

insert(val);

break;

case 2:

printf("Red Black Tree Structure:\n");

printf("---------------------------");

printTree(root,1,0);

printf("---------------------------");

break;

case 3:

exit(0);

}

}

}