Sri Sivasubramania College of Engineering, Kalavakkam

Department of Computer Science and Engineering

Tutorial on AVL Tree

**Date: 10.11.2022 [CO1, K3]**

1. Write routines for single rotation with right child and single rotation with right child.
2. Construct AVL tree for the following input:

**15,20,24,10,13,7,30,36,25**

1. Construct AVL tree for the following input

**H, I, J, B, A, E, C, F, D, G, K, L**

1. While inserting the elements **H, I, J, B, A, E, C, F, D, G, K, L** into AVL tree, trace the insertion algorithm for every element insertion.

Following is the avladt.h. Look at the and complete the code. And write avlApp.c for the above-mentioned input.

**avladt.h**

struct avltree

{

int data;

struct avltree \*left;

struct avltree \*left;

int height;

};

**static int height (struct avltree \*t)**

{

if(t==NULL)

return -1;

else

return t->height;

}

**struct avltree\* insert (struct avltree \*t, int x)**

{

if(t==NULL)

{

t=struct avltree\* malloc(sizeof(struct avltree));

t->data=x;

t->height=0;

t->left=t->right=NULL;

}

}

else if(x<t->data)

{

t->left=insert(t->left,x);

if(height(t->left)-height(t->right) ==2)

if(x<t->left->data)

t=singlerotatewithleft(t);

else

t=doublerotatewithleft(t);

}

else if(x>t->element)

{

t->right=insert(t->right,x);

if(height(t->left)-height(t->right) ==2)

if(x>t->right->data)

t=singlerotatewithright(t);

else

t=doublerotatewithright(t);

}

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

return t;

}

**struct avltree\* singlerotatewithleft(struct avltree \*k2)**

{

struct avltree \*k1;

k1=k2->left;

k2->left=k1->right;

k1->right=k2;

k2->height=max(height(k2->left),height(k2->right))+1;

k1->height=max(height(k1->left),height(k1-> height))+1;

return k1;

}

**struct avltree\* doublerotatewithleft(struct avltree \*k3)**

{

//rotate between k1 and k2

k3->left=singlerotatewithright(k3->left);

//rotate between k3 and k2

return singlerotatewithleft(k3);

}

**struct avltree\* singlerotatewithright(struct avltree \*?)**

**{**

**------------**

**}**

**struct avltree\* doublerotatewithleft(struct avltree \*?)**

**{**

**------------**

**}**