UCS1302:
DATA STRUCTURES

AVL implementation

Session Meta Data

Author	**Dr. B. Bharathi**
Reviewer	
Version Number	1.2
Release Date	07 August 2019

Revision History

Revision Date	Details	Version no.
22 September 2017	1. New SSN template applied	1.2

Session Objectives

- To learn about AVL tree implementation

Session Outcomes

- At the end of this session, participants will be able to
  - Understand the concepts of AVL tree implementation

Agenda

- Implementation of AVL tree operations

AVL tree implementation

Dr. B. Bharathi
SSNCE

August 07, 2019

Node Declaration for AVL Trees

```
typedef struct avlnode *position;
typedef struct avlnode *avltree;
struct avlnode
{
    elementtype element;
    avltree left;
    avltree right;
    int height;
};
```

Function to compute height of an AVL node

```
static int height (position p)
{
    if(p==NULL)
        return -1;
    else
        return p->height;
}
```

Insertion into an AVL tree

```
avltree insert(elementtype x, avltree t)
{
if(t==NULL)
{
    t=malloc(sizeof(struct avlnode));
    if(t==NULL)
                FatalError("Out of Space");
    else
    {
                t->element=x;
                t->height=0;
                t->left=t->right=NULL;
    }
}
```


contd.

```
else if(x<t->element)
{
    t->left=insert(x,t->left);
    if(height(t->left) – height(t->right) == 2)
        if(x<t->left->element)
                t=singlerotatewithleft(t);
        else
                t=doublerotatewithleft(t);
}
```

contd.

```
else if(x>t->element)
{
    t->right=insert(x,t->right);
    if(height(t->right) – height(t->left) == 2))
        if(x>t->right->element)
                t=singlerotatewithright(t);
        else
                t=doublerotatewithright(t);
}
t->height=max(height(t->left),height(t->right))+1;
return t;
}
```

Routine to perform single rotation

```
static position singlerotatewithleft(position k2)
{
    position 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),k2-> height)+1;
    return k1;
}
```

Routine to perform double rotation

```
static position doublerotatewithleft(position k3)
{ //rotate between k1 and k2
    k3->left=singlerotatewithright(k3->left);
    //rotate between k3 and k2
    return singlerotatewithleft(k3);
}
```

Summary

- AVL tree Implementation
- Operations on AVL tree