```
#include<iostream>
#include<string.h>
using namespace std;

class avl
{       char word[30],mean[50];
        avl *left,*right;
        int ht;
public:
        avl* create(avl *root);
        avl* insert(avl *root,char word[],char mean[]);
        void display(avl*);
        int  height(avl*);
        avl* rotateright(avl*);
        avl* rotateleft(avl*);
        int  BF(avl*);
        avl* Delete(avl*,char*);
        avl* RR(avl*);
        avl* LL(avl*);
        avl* LR(avl*);
        avl* RL(avl*);
        avl* mirror(avl*);
        avl* update(avl*);
};

avl* avl::create(avl *root)
{
   int n,i;
        char w[20],m[50];
        cout<<"\n Enter the number of words:\t";
        cin>>n;
        for(i=0;i<n;i++)
        {       cout<<"\n Enter the "<<i+1<<" word:";
                cin>>w;
```

```
                cout<<"\n Enter the meaning:";
                cin>>m;
                root=insert(root,w,m);
        }
        return root;
}

avl* avl::insert(avl *root,char w[],char m[])
{       if(root==NULL)
        {       root=new avl;
                strcpy(root->word,w);
                strcpy(root->mean,m);
                root->left=NULL;
                root->right=NULL;
                return root;
        }
  else
  {
         if(strcmp(w,root->word)>0)
         {
                root->right=insert(root->right,w,m);
                if(BF(root)==2)
                {
                        if(strcmp(w,root->word)>=0)
                                root=RR(root);
                        else
                                root=RL(root);
                }

         }
         else
         {
                 if(strcmp(w,root->word)<0)
                 {
```

```
                                        root->left=insert(root->left,w,m);
                                        if(BF(root)==-2)
                                        {
                                                if(strcmp(w,root->word)<=0)
                                                        root=LL(root);
                                                else
                                                        root=LR(root);
                                        }
                        }
                }
        }
        root->ht=height(root);
        return root;
}

void avl::display(avl* root)
{       if (root!=NULL)
        {
           display(root->left);
                cout<<"\n"<<root->word<<" :: "<<root->mean<<"(Bf="<<BF(root)<<")";
                display(root->right);
        }
}

int avl::height(avl *root)
{
        int lh,rh;
        if(root==NULL)
                return 0;
        if(root->left==NULL)
                lh=0;
        else
                lh=1+root->left->ht;
```

```
        if(root->right==NULL)
                rh=0;
        else
                rh=1+root->right->ht;

        if(lh>rh)
                return(lh);
        else
                return(rh);
}

avl* avl::rotateright(avl *x)
{
        avl *y;
        y=x->left;
        x->left=y->right;
        y->right=x;
        x->ht=height(x);
        y->ht=height(y);
        return(y);
}

avl* avl::rotateleft(avl *x)
{
        avl *y;
        y=x->right;
        x->right=y->left;
        y->left=x;
        x->ht=height(x);
        y->ht=height(y);
        return(y);
}

int avl::BF(avl *root)
```

```
{
        int lh,rh;
        if(root==NULL)
                return 0;

        if(root->left==NULL)
                lh=0;
        else
                lh=1+root->left->ht;

        if(root->right==NULL)
                        rh=0;
                else
                rh=1+root->right->ht;
        int z=lh-rh;
        return(z);
}
avl* avl::Delete(avl* T,char* w)
{
        avl *p;
        if(T==NULL)
        {
                cout<<"\n Word not found";
                return T;
        }
        if(strcmp(w,T->word)>0)
        {
                T->right=Delete(T->right,w);
                if(BF(T)==2)
                {
                        if(BF(T->left)>=0)
                                T=LL(T);
                        else
                                T=LR(T);
```

```
		}


}
else
		if(strcmp(w,T->word)<0)
		{
			T->left=Delete(T->left,w);
			if(BF(T)==-2)
			{
				if(BF(T->right)<=0)
					T=RR(T);
				else
					T=RL(T);
			}
		}
		else
		{
			if(T->right!=NULL)
			{
				p=T->right;
				while(p->left!=NULL)
					p=p->left;
				strcpy(T->word,p->word);
				strcpy(T->mean,p->mean);
				T->right=Delete(T->right,p->word);
					if(BF(T)==2)
					{
						if(BF(T->left)>=0)
							T=LL(T);
						else
							T=LR(T);
					}
			}
			else
```

```
                                        return(T->left);
                        }
        T->ht=height(T);
        return(T);
}
avl* avl::RR(avl*T)
{
        T=rotateleft(T);
        return(T);
}
avl* avl::LL(avl*T)
{
        T=rotateright(T);
        return(T);
}
avl* avl::LR(avl*T)
{
        T->left=rotateleft(T->left);
        T=rotateright(T);
        return(T);
}
avl* avl::RL(avl*T)
{
        T->right=rotateright(T->right);
        T=rotateleft(T);
        return(T);
}
avl* avl::mirror(avl* temp)
{
        avl*p;
        if(temp==NULL)
                return NULL;
        p=new avl;
        strcpy(p->word,temp->word);
```

```
        strcpy(p->mean,temp->mean);
        p->left=mirror(temp->right);
        p->right=mirror(temp->left);
        return p;
}
avl *avl::update(avl *root)
{
        avl *temp;
        char w[20],m[50];
        temp=root;
        cout<<"\n Enter a word which you want to update : ";
        cin>>w;
        cout<<"\n Enter the meaning(updated meaning): ";
        cin>>m;
        while(temp!=NULL)
        {
                if(strcmp(w,temp->word)==0)
                {
                        strcpy(temp->word,w);
                        strcpy(temp->mean,m);
                        break;
                }
                if(strcmp(w,temp->word)<0)
                {
                        temp=temp->left;
                }
                else
                {
                        temp=temp->right;
                }
        }
        return root;
        cout<<root;
}
```

```
int main()
{
	int ch;
	char z;
	avl d,*root,*root1;
	root=NULL;
	char w[20],m[50];
	cout<<"******************CREATION OF AVL*****************";
	do
	{
		cout<<"\n 1.Create \n 2.Insert \n 3.Delete \n 4.Display\n 5.Update\n 6.Descending order\n
7.Exit";
		cout<<"\n Enter your choice : ";
		cin>>ch;
		switch(ch)
		{
		case 1:root=d.create(root);
		break;
		case 2:cout<<"\n Enter the word :";
			cin>>w;
			cout<<"\n Enter the meaning :";
			cin>>m;
			root=d.insert(root,w,m);
			break;
		case 3:cout<<"\n Enter word you want to delete : ";
			cin>>w;
			root=d.Delete(root,w);
			break;
		case 4:cout<<"\n Word in ascending order : ";
			d.display(root);
			break;
		case 5:d.update(root);
			break;
```

```
                    case 6:cout<<"\n Word in decending order : ";
                        root1=d.mirror(root);
                        d.display(root1);
                        break;
                    case 7:cout<<"\n EXIT!";
                    }
                    cout<<"\n Do you want to continue?"<<endl;
                    cin>>z;
            }while(z=='Y'||z=='y');
            return 0;
}
```

OUTPUT:-

```
******************CREATION OF AVL*****************
 1.Create
 2.Insert
 3.Delete
 4.Display
 5.Update
 6.Descending order
 7.Exit
 Enter your choice : 1
 Enter the number of words:    3
 Enter the 1 word:HELLO
 Enter the meaning:GREETING
 Enter the 2 word:GOODBYE
 Enter the meaning:FAREWELL
 Enter the 3 word:THERE
 Enter the meaning:ACTION
 Do you want to continue? Y

1.Create
 2.Insert
```

3.Delete

4.Display

5.Update

6.Descending order

7.Exit

Enter your choice : 4

Word in ascending order :

GOODBYE :: FAREWELL(Bf=0)

HELLO :: GREETING(Bf=0)

THERE :: ACTION(Bf=0)

Do you want to continue? Y

1.Create

2.Insert

3.Delete

4.Display

5.Update

6.Descending order

7.Exit

Enter your choice : 2

Enter the word :TITLE

Enter the meaning :NAME

Do you want to continue? Y

1.Create

2.Insert

3.Delete

4.Display

5.Update

6.Descending order

7.Exit

Enter your choice : 4

Word in ascending order :

GOODBYE :: FAREWELL(Bf=0)

HELLO :: GREETING(Bf=-1)

THERE :: ACTION(Bf=-1)

TITLE :: NAME(Bf=0)

Do you want to continue? Y

1.Create

2.Insert

3.Delete

4.Display

5.Update

6.Descending order

7.Exit

Enter your choice : 6

Word in decending order :

TITLE :: NAME(Bf=0)

THERE :: ACTION(Bf=1)

HELLO :: GREETING(Bf=0)

GOODBYE :: FAREWELL(Bf=0)

Do you want to continue? N