//GAYATHRI .M-185001050

//AVL TREE

Contents of functions.h file

```
#include<stdio.h>
#include<stdlib.h>
#include<string.h>
typedef struct avltree
{
        char eng[50],tamil[50],hindi[50];
        int ht;
        struct avltree * left,*right;
}tree;
int height(tree * p)
{
        if(p==NULL)
                return -1;
        else
                return p->ht;
}
int max(tree * t1,tree * t2)
{
        if(height(t1)>height(t2))
                return height(t1);
        else
                return height(t2);
}
void inorder(tree *t)
{
   if(t!=NULL)
   {
       inorder(t->left);
       printf(" %s  \t%s  \t%s\n",t->eng,t->tamil,t->hindi);
       inorder(t->right);
   }
}
void meaning(tree * t,char eng[50])
{
        if(t)
        {
                if(strcmp(eng,t->eng)<0)
```

```
                meaning(t->left,eng);
        else if(strcmp(eng,t->eng)>0)
                meaning(t->right,eng);
        else
                {
                        printf("\nWord: %s",t->eng);
                        printf("\nTamil: %s",t->tamil);
                        printf("\nHindi: %s\n",t->hindi);
                }
    }
}
tree * singlerotateleft(tree * k2)
{
    printf("\nPerforming Single Rotate with left.....");
    tree * k1;
    k1=k2->left;
    k2->left=k1->right;
    k1->right=k2;
    k2->ht=max(k2->left,k2->right)+1;
    k1->ht=max(k1->left,k2)+1;
    return k1;
}
tree * singlerotateright(tree * k2)
{
    printf("\nPerforming Single Rotate with right.....");
    tree * k1;
    k1=k2->right;
    k2->right=k1->left;
    k1->left=k2;
    k2->ht=max(k2->left,k2->right)+1;
    k1->ht=max(k1->left,k2)+1;
    return k1;
}
tree * doublelerotateleft(tree * k3)
{
    printf("\nPerforming Double Rotate with left.....");
    k3->left=singlerotateright(k3->left);
    return singlerotateleft(k3);
}
tree * doublerotateright(tree * k3)
{
    printf("\nPerforming Double Rotate with right.....");
```

```
        k3->right=singlerotateleft(k3->right);
        return singlerotateright(k3);
}
tree * insert(tree * t,char eng[50],char tamil[50],char hindi[50])
{
        if(t==NULL)
        {
                t=malloc(sizeof(tree));
                if(t==NULL)
                        printf("\nNULL");
                strcpy(t->eng,eng);
                strcpy(t->tamil,tamil);
                strcpy(t->hindi,hindi);
                t->left=t->right=NULL;
                t->ht=0;
        }
        else if(strcmp(eng,t->eng)<0)
        {
                t->left=insert(t->left,eng,tamil,hindi);
                if(height(t->left)-height(t->right)==2)
                {
                        if(strcmp(eng,t->left->eng)<0)
                                t=singlerotateleft(t);
                        else
                                t=doublelerotateleft(t);
                }
        }
        else if(strcmp(eng,t->eng)>0)
        {
                t->right=insert(t->right,eng,tamil,hindi);
                if(height(t->left)-height(t->right)==-2)
                {
                        if(strcmp(eng,t->right->eng)>0)
                                t=singlerotateright(t);
                        else
                                t=doublerotateright(t);
                }
        }
        t->ht=max(t->left,t->right)+1;
        return t;
}
```

Contents of avl.c file

```
#include"functions.h"
int main()
{
	tree * root=NULL;
	char str[50],m1[50],m2[50];
	printf("\n\nEnter English word: ");
	scanf(" %[^\n]",str);
	while(strcmp(str,"0"))
	{
		printf("\nEnter its Tamil meaning: ");
		scanf(" %[^\n]",m1);
		printf("\nEnter its Hindi meaning: ");
		scanf(" %[^\n]",m2);
		root=insert(root,str,m1,m2);
		printf("\n\nEnglish\t\tTamil\t\tHindi\n");
		inorder(root);
		printf("\n\nEnter English word: ");
		scanf(" %[^\n]",str);
	}
	strcpy(str,"");
	while(strcmp(str,"0"))
	{
		printf("\nFind meaning of: ");
		scanf(" %[^\n]",str);
		meaning(root,str);
	}
	return 0;
}
```