VIGNESH BALAJI S

205001126 CSE B

adt.h

```c
#include<stdio.h>
struct polyADT{
    int c;
    int e;
    struct polyADT *next;
};
void display(struct polyADT *p);
struct polyADT* insertEnd(struct polyADT *p, int c, int e);
struct polyADT* polyAdd(struct polyADT *p1, struct polyADT *p2);
struct polyADT* polyMUL(struct polyADT *p1, struct polyADT *p2);
struct polyADT* polySimplify(struct polyADT *p);
struct polyADT* createPoly(struct polyADT *res);
void polyDegree(struct polyADT *p);
int polyEvaluate(struct polyADT *p);
```

impl.h

```c
#include<stdio.h>
#include<stdlib.h>
#include"adt.h"
void display(struct polyADT *p){
    struct polyADT *ptr = p;
    printf("\n------------\n");
    while(ptr!=NULL){
        printf("%dx^%d ",ptr->c,ptr->e);
        if(ptr->next!=NULL && ptr->next->c>0) printf("+ ");
        ptr=ptr->next;
    }
    printf("\n------------\n");
}

struct polyADT* insertEnd(struct polyADT *p, int c, int e){
    struct polyADT *temp=(struct polyADT*) malloc(sizeof(struct polyADT));
    temp->next=NULL; temp->c=c; temp->e=e;
    if(p==NULL) p=temp;
    else{
        struct polyADT *ptr=p;
        while(ptr->next!=NULL) ptr=ptr->next;
        ptr->next=temp;
    }
    return p;
}
```

```
struct polyADT* polyAdd(struct polyADT *p1, struct polyADT *p2){
    static struct polyADT *p3 = NULL;
    struct polyADT *ptr1=p1, *ptr2=p2;
    while(ptr1!=NULL && ptr2!=NULL){
        if(ptr1->e > ptr2->e){
            p3=insertEnd(p3, ptr1->c, ptr1->e); ptr1 = ptr1->next;
        }
        else if(ptr2->e > ptr1->e){
            p3=insertEnd(p3, ptr2->c, ptr2->e); ptr2 = ptr2->next;
        }
        else{
            p3=insertEnd(p3, ptr1->c + ptr2->c, ptr1->e);
            ptr1 = ptr1->next; ptr2 = ptr2->next;
        }
    }
    while(ptr1!=NULL){
        p3=insertEnd(p3, ptr1->c, ptr1->e); ptr1 = ptr1->next;
    }
    while(ptr2!=NULL){
        p3=insertEnd(p3, ptr2->c, ptr2->e); ptr2 = ptr2->next;
    }
    return p3;
}


struct polyADT* polySimplify(struct polyADT *p){
    struct polyADT *res=(struct polyADT*)malloc(sizeof(struct polyADT)),*temp;
    res->next=NULL;
    while(p!=NULL){
        temp=res;
        while(temp->next!=NULL){
            if(temp->next->e==p->e){
                temp->next->c+=p->c;break;
            }
            else if(temp->next->e<p->e){
                struct polyADT *new=(struct polyADT*)malloc(sizeof(struct polyADT));
                new->c=p->c; new->e=p->e;
                new->next=temp->next;
                temp->next=new; break;
            }
            else{
                temp=temp->next;
            }
        }
        if(temp->next==NULL){
            struct polyADT *new=(struct polyADT*)malloc(sizeof(struct polyADT));
            new->c=p->c; new->e=p->e; new->next=NULL;
            temp->next=new;
        }
        p=p->next;
    }
    return res->next;
}
```

```c
struct polyADT* polyMul(struct polyADT *p1, struct polyADT *p2){
    struct polyADT *p3 = NULL;
    struct polyADT *ptr1=p1, *ptr2=p2;
    while(ptr1!=NULL){
        ptr2=p2;
        while(ptr2!=NULL){
            p3=insertEnd(p3, ptr1->c * ptr2->c, ptr1->e + ptr2->e); ptr2 = ptr2->next;
        }
        ptr1 = ptr1->next;
    }
    display(p3);
    printf("After simplifying: \n");
    return polySimplify(p3);
}

struct polyADT* createPoly(struct polyADT *res){
    printf("\n\nEnter the number of terms: ");
    int x; scanf("%d",&x);
    int c,e;
    for(int i=0; i<x; i++){
        printf("\nEnter coeff and exp sep by space: ");
        scanf("%d %d",&c,&e);
        res=insertEnd(res,c,e);
    }
    return res;
}

void polyDegree(struct polyADT *p){
    int deg;
    while(p!=NULL){
        if(p->e>deg) deg=p->e;
        p=p->next;
    }
    printf("\n----------\nDegree: %d\n----------\n",deg);
}

int polyEvaluate(struct polyADT *p){
    int x; printf("\nEnter x value: ");
    scanf("%d",&x);
    int res;
    while(p!=NULL){
        int pow=1, i=p->e;
        while(i!=0){
            pow*=x;i--;
        }
        res+=p->c*pow;
        p=p->next;
    }
    return res;
}
```

appl.c

```c
#include<stdio.h>
#include"impl.h"

int main(){
    int done=0, user;
    struct polyADT *p1=NULL, *p2=NULL;
    while(done==0){
        printf("\n\n----------------------
\nEnter\n1 to ADD\n2 to MULTIPLY\n3 to SIMPLIFY\n4 to FIND DEGREE\n5 to EVALUATE\n6 to Quit\n
------------------------\ninput: ");
        scanf("%d",&user);
        switch(user){
            case 1:{
                p1=NULL, p2=NULL;
                p1=createPoly(p1); p2=createPoly(p2);
                display(polyAdd(p1,p2));
            };break;
            case 2:{
                p1=NULL, p2=NULL;
                p1=createPoly(p1); p2=createPoly(p2);
                display(polyMul(p1,p2));
            };break;
            case 3:{
                p1=NULL;
                p1=createPoly(p1);
                p1=polySimplify(p1);
                display(p1);
            };break;
            case 4:{
                p1=NULL;
                p1=createPoly(p1);
                polyDegree(p1);
            };break;
            case 5:{
                p1=NULL;
                p1=createPoly(p1);
                printf("\nValue: %d\n",polyEvaluate(p1));
            };break;
            case 6:{
                done = 1;
            };break;
        };
    }
    return 0;
}
```

Output screenshots

1)ADD

```
E:\sem3\dsLab\ass2>a.exe


---------------------
Enter
1 to ADD
2 to MULTIPLY
3 to SIMPLIFY
4 to FIND DEGREE
5 to EVALUATE
6 to Quit
-------------------------
input: 1


Enter the number of terms: 3

Enter coeff and exp sep by space: 5 2

Enter coeff and exp sep by space: 4 1

Enter coeff and exp sep by space: 2 0


Enter the number of terms: 2

Enter coeff and exp sep by space: -5 1

Enter coeff and exp sep by space: -5 0

------------
5x^2 -1x^1 -3x^0
------------
```

2)MULTIPLY

```
---------------------
Enter
1 to ADD
2 to MULTIPLY
3 to SIMPLIFY
4 to FIND DEGREE
5 to EVALUATE
6 to Quit
-----------------------
input: 2


Enter the number of terms: 3

Enter coeff and exp sep by space: 5 2

Enter coeff and exp sep by space: 4 1

Enter coeff and exp sep by space: 2 0


Enter the number of terms: 2

Enter coeff and exp sep by space: -5 1

Enter coeff and exp sep by space: -5 0

------------
-25x^3 -25x^2 -20x^2 -20x^1 -10x^1 -10x^0
------------
After simplifying:

------------
-25x^3 -45x^2 -30x^1 -10x^0
------------
```

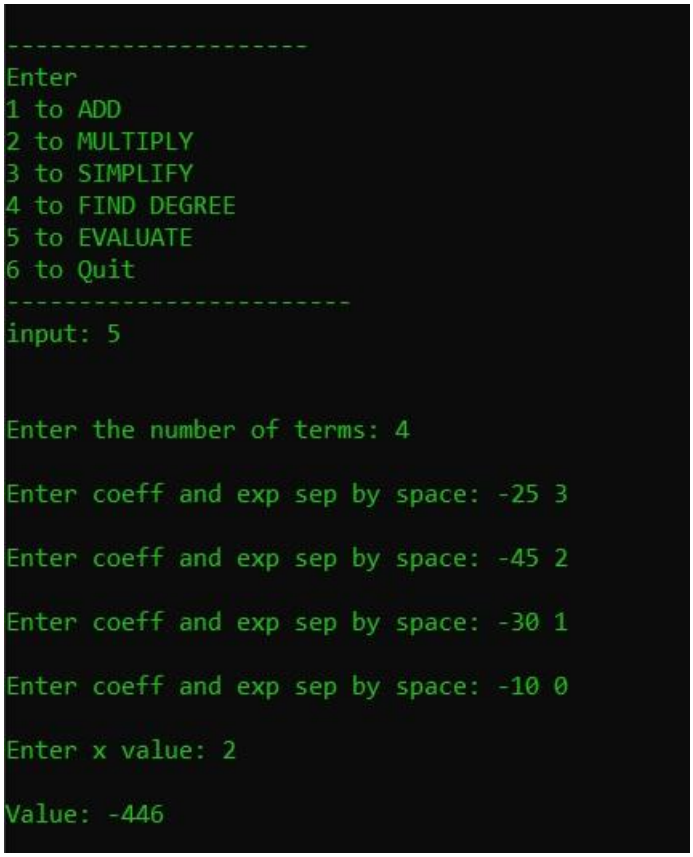
3)SIMPLIFY

```
---------------------
Enter
1 to ADD
2 to MULTIPLY
3 to SIMPLIFY
4 to FIND DEGREE
5 to EVALUATE
6 to Quit
----------------------
input: 3


Enter the number of terms: 6

Enter coeff and exp sep by space: -25 3

Enter coeff and exp sep by space: -20 2

Enter coeff and exp sep by space: -10 1

Enter coeff and exp sep by space: -25 2

Enter coeff and exp sep by space: -20 1

Enter coeff and exp sep by space: -10 0

-----------
-25x^3 -45x^2 -30x^1 -10x^0
-----------
```

4) FIND DEGREE

```
---------------------
Enter
1 to ADD
2 to MULTIPLY
3 to SIMPLIFY
4 to FIND DEGREE
5 to EVALUATE
6 to Quit
----------------------
input: 4


Enter the number of terms: 4

Enter coeff and exp sep by space: 25 3

Enter coeff and exp sep by space: -45 2

Enter coeff and exp sep by space: -30 1

Enter coeff and exp sep by space: -10 0

----------
Degree: 4
----------
```

5)EVALUATE



Enter
1 to ADD
2 to MULTIPLY
3 to SIMPLIFY
4 to FIND DEGREE
5 to EVALUATE
6 to Quit
----
input: 5
Enter the number of terms: 4
Enter coeff and exp sep by space: -25 3
Enter coeff and exp sep by space: -45 2
Enter coeff and exp sep by space: -30 1
Enter coeff and exp sep by space: -10 0
Enter x value: 2
Value: -446