Name– Shreya Mishra

PRN – PES1202102193

SRN – PES1UG21CS574

# QUES 1

poly.c file

#include <stdio.h>

#include <stdlib.h>

struct poly

{

    int pow1;

    int pow2;

    int coeff;

    struct poly \*next;

};

struct poly \*poly1 = NULL;

struct poly \*poly2 = NULL;

struct poly \*polys = NULL;

struct poly \*p\_insert(int x, int y, int c, struct poly \*p)

{

    struct poly \*temp = (struct poly \*)malloc(sizeof(struct poly));

    temp->pow1 = x;

    temp->pow2 = y;

    temp->coeff = c;

    temp->next = NULL;

    if (p == NULL)

    {

        p = temp;

        return p;

    }

    struct poly \*ptr = p;

    while (ptr->next != NULL)

    {

        ptr = ptr->next;

    }

    ptr->next = temp;

    return p;

}

void show(struct poly \*p)

{

    if (p == NULL)

    {

        printf("Polynomial is Empty!\n");

        return;

    }

    struct poly \*head = p;

    while (head != NULL)

    {

        printf("%dx^%dy^%d", head->coeff, head->pow1, head->pow2);

        if (head->next != NULL)

        {

            printf(" + ");

        }

        head = head->next;

    }

}

struct poly \*poly\_add(struct poly \*p1, struct poly \*p2)

{

    struct poly \*to\_p1 = p1;

    struct poly \*to\_p2 = p2;

    struct poly \*sum = NULL;

    while (p1 != NULL && p2 != NULL)

    {

        if ((p1->pow1 == p2->pow1) && (p1->pow2 == p2->pow2))

        {

            if (p1->coeff + p2->coeff != 0)

            {

                sum = p\_insert(p1->pow1, p1->pow2, p1->coeff + p2->coeff, sum);

                p1 = p1->next;

                p2 = p2->next;

            }

            else if (p1->coeff + p2->coeff == 0)

            {

                p1 = p1->next;

                p2 = p2->next;

                continue;

            }

        }

        else if (((p1->pow1 > p2->pow1) || (p1->pow1 == p2->pow1)) && ((p1->pow2 > p2->pow2) || (p1->pow2 == p2->pow2)))

        {

            sum = p\_insert(p1->pow1, p1->pow2, p1->coeff, sum);

            p1 = p1->next;

        }

        else

        {

            sum = p\_insert(p2->pow1, p2->pow2, p2->coeff, sum);

            p2 = p2->next;

        }

    }

    struct poly \*last\_sum = sum;

    while (last\_sum->next != NULL)

    {

        last\_sum = last\_sum->next;

    }

    if (p1 != NULL)

    {

        last\_sum->next = p1;

    }

    else if (p2 != NULL)

    {

        last\_sum->next = p2;

    }

    p1 = to\_p1;

    p2 = to\_p2;

    return sum;

}

int main()

{

    int x, y, c;

    printf("Enter Coeff, Power of x,y for Poly 1:\n");

    scanf("%d %d %d", &c, &x, &y);

    while (c != 0)

    {

        poly1 = p\_insert(x, y, c, poly1);

        printf("Enter Coeff, Power of x,y for Poly 1:\n");

        scanf("%d %d %d", &c, &x, &y);

    }

    printf("Enter Coeff, Power of x,y for Poly 2: \n");

    scanf("%d %d %d", &c, &x, &y);

    while (c != 0)

    {

        poly2 = p\_insert(x, y, c, poly2);

        printf("Enter Coeff, Power of x,y for Poly 2:\n");

        scanf("%d %d %d", &c, &x, &y);

    }

    printf("\npolynomial 1:\n");

    show(poly1);

    printf("\npolynomial 2:\n");

    show(poly2);

    polys = poly\_add(poly1, poly2);

    printf("\n");

    printf("\nSum of the polynomial\n");

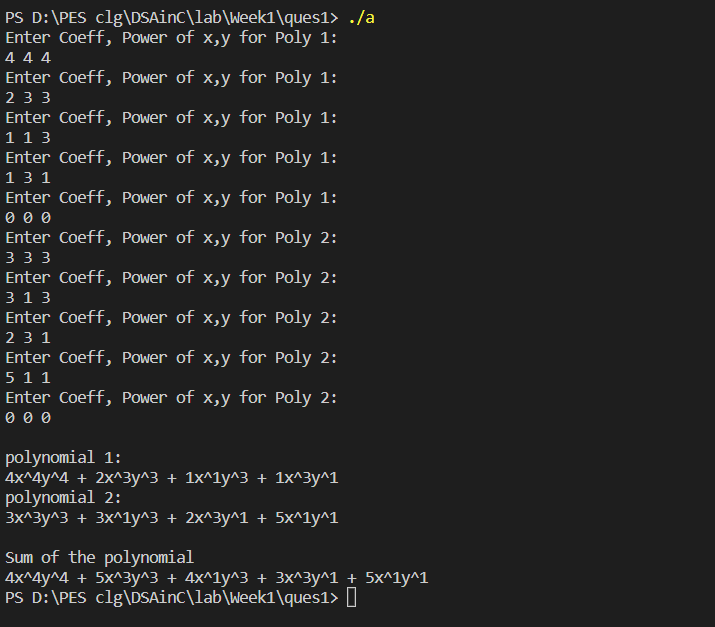
    show(polys);

    printf("\n");

    return 0;

}

Output



# QUES-2

deleteAlt.c

#include <stdio.h>

#include <stdlib.h>

typedef struct node{

    int key;

    struct node\* link;

}node\_t;

typedef struct list{

    node\_t\* h;

}list\_t;

void display(list\_t\* p\_list){

    if(p\_list->h== NULL){

        printf("\n\nNo elements in the list\n");

    }

    else{

        node\_t\* temp = (node\_t\*)malloc(sizeof(node\_t));

        temp = p\_list->h;

        printf("The elements are: ");

        while(temp != NULL){

            printf("%d ",temp->key);

            temp = temp->link;

        }

        printf("\n\n\n");

    }

}

void init\_list(list\_t\* p\_list){

    p\_list->h = NULL;

}

void insert\_head(list\_t\* p\_list, int data){

    node\_t\* temp = (node\_t\*)malloc(sizeof(node\_t));

    temp->key = data;

    temp->link = NULL;

    if(p\_list->h == NULL){

        p\_list->h = temp;

    }

    else{

        temp->link = p\_list->h;

        p\_list->h = temp;

    }

}

void del\_alt(list\_t\* p\_list)

{

    if(p\_list->h == NULL)

    {

        printf("The list is empty\n");

    }

    else

    { /\*initialzing node to be deleted\*/

        node\_t\* pres;

        node\_t\* prev;

        pres = p\_list->h;

        prev = NULL;

        while(pres != NULL)

        {

            if(prev == NULL) /\* Deleting first element \*/

            {

                p\_list->h = pres->link;

            }

            else

            {

                prev->link = pres->link;

            }

            prev = pres;

            pres = pres->link; /\* changing link \*/

        }

    }

}

int main()

{

    list\_t l;

    init\_list(&l);

    int choice, ele;

    while(1)

    {

        display(&l);

        printf("1) Insert at head\n");

        printf("2) Delete alternate nodes\n");

        printf("3) Quit\n");

        scanf("%d",&choice);

        switch(choice){

            case 1:

                printf("Enter the element: ");

                scanf("%d",&ele);

                printf("\n\n");

                insert\_head(&l,ele);

                break;

            case 2:

                del\_alt(&l);

                break;

            case 3:

                exit(0);

                break;

            default:

                printf("Invalid Choice \n");

                break;

        }

    }

}

OUTPUT

