```
/* 9. Develop a Program in C for the following
operations on Singly Circular Linked List
(SCLL) with header nodes.
a. Represent and Evaluate a Polynomial P(x,y,z) =
6x2y2z-4yz5+3x3yz+2xy5z-2xyz3

b. Find the sum of two polynomials POLY1(x,y,z) and
POLY2(x,y,z) and store the result in
POLYSUM(x,y,z) Support the program with appropriate
functions for each of the above
operations
*/

#include <stdio.h>
#include <stdlib.h>
#include <math.h>

struct node
{
    int c;
    int px;
    int py;
    int pz;
    struct node *link;
};

typedef struct node * NODE;

float evaluate(float x, float y, float z, NODE head)
{
    float sum;
    NODE cur;
    sum = 0;

    cur = head->link;

    while (cur != head)
    {
```

```
            sum = sum + cur->c * pow(x, cur->px) * pow(y
            ,cur->py) * pow(z, cur->pz);
            cur = cur->link;
        }

        return sum;
}

void print_polynomial(NODE head)
{
        NODE p;

        for (p = head->link; p != head; p = p->link)
        {
            if (p->c > 0)
                printf("+%dx^%dy^%dz^%d ", p->c, p->px,
                p->py, p->pz);
            else
                printf("%dx^%dy^%dz^%d ", p->c, p->px, p
                ->py, p->pz);
        }

        printf("\n");
}

NODE getnode()
{
        NODE x;

        x = (NODE) malloc(sizeof(struct node));

        if (x == NULL)
        {
            printf("Not enough memory\n");
            exit(0);
        }

        return x;
```

```
}

NODE insert_rear(int c, int px, int py, int pz, NODE
 head)
{
    NODE cur, temp;

    temp = getnode();

    temp->c = c;
    temp->px = px;
    temp->py = py;
    temp->pz = pz;

    cur = head->link;

    while (cur->link != head)
    {
        cur = cur->link;
    }

    cur->link = temp;
    temp->link = head;

    return head;
}

NODE read_polynomial()
{
    NODE head;
    int c, px, py, pz;

    head = getnode();

    head->link = head;

    for(;;)
    {
```

```
            scanf("%d", &c);
            if (c == 0) break;

            scanf("%d", &px);
            scanf("%d", &py);
            scanf("%d", &pz);

            head = insert_rear(c, px, py, pz, head);
        }

        return head;
}

NODE add_2_polynomials(NODE h1, NODE h2)
{
        int sum;
        NODE p, q, h3;

        h3 = getnode();

        h3->link = h3;

        for (p = h1->link; p != h1; p = p->link)
        {
            for (q = h2->link; q != h2; q = q->link)
            {
                if (p->px == q->px && p->py == q->py &&
                p->pz == q->pz)
                {
                    sum = p->c + q->c;

                    q->c = 0;
                    if (sum != 0) h3 = insert_rear(sum,
                    p->px, p->py, p->pz, h3);
                    break;
                }
            }
```

```
        if (q == h2) h3 = insert_rear(p->c, p->px, p
        ->py, p->pz, h3);
    }

    for (q = h2->link; q != h2; q = q->link)
    {
        if (q->c == 0) continue;

        h3 = insert_rear(q->c, q->px, q->py, q->pz,
        h3);
    }

    return h3;
}


int main()
{
    NODE h1, h2, h3;

    int choice;
    float x, y, z, sum;

    for(;;)
    {
        printf("1:Evaluate  2:Add  3:Exit : ");
        scanf("%d", &choice);

        switch(choice)
        {
            case 1:
                printf("Enter a polynomial : ");
                h1 = read_polynomial();

                printf("Enter x y and z : ");
                scanf("%f %f %f", &x, &y, &z);

                printf("Poly1: ");
```

```
                print_polynomial(h1);

                sum = evaluate(x, y, z, h1);

                printf("Result = %f\n", sum);
                break;

            case 2:
                printf("Enter first polynomial : ");
                h1 = read_polynomial();

                printf("Enter second polynomial : "
                );
                h2 = read_polynomial();

                printf("Poly1: ");
                print_polynomial(h1);

                printf("Poly2: ");
                print_polynomial(h2);

                h3 = add_2_polynomials(h1, h2);

                printf("Poly3: ");
                print_polynomial(h3);

                break;

            default:
                exit(0);
        }
    }
}


/* OUTPUT
a)
Case 1:
```

```
1:Evaluate  2:Add  3:Exit : 1

Enter a polynomial : 6 2 2 1 -4 0 1 5 3 3 1 1 2 1 5
1 -2 1 1 3
0
Enter x y and z : 2 3 4

Poly1: +6x^2y^2z^1 -4x^0y^1z^5 +3x^3y^1z^1
+2x^1y^5z^1 -2x^1y^1z^3

Result = -8016.000000

Case 2:

1:Evaluate  2:Add  3:Exit : 1
Enter a polynomial : 6 2 2 1 -4 0 1 5 3 3 1 1 2 1 5
1 -2 1 1 3
0
Enter x y and z : 1 1 1

Poly1: +6x^2y^2z^1 -4x^0y^1z^5 +3x^3y^1z^1
+2x^1y^5z^1 -2x^1y^1z^3

Result = 5.000000



b)

Case1:

1:Evaluate  2:Add  3:Exit : 2

Enter first polynomial : 2 3 2 1 2 1 0 1 0

Enter second polynomial : 3 3 2 1 -2 1 0 1 0
```

```
Poly1: +2x^3y^2z^1 +2x^1y^0z^1

Poly2: +3x^3y^2z^1 -2x^1y^0z^1

Poly3: +5x^3y^2z^1


*/
```