Laboratory Component 9

Design, Develop and Implement 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>
#define COMPARE(x, y)       ( (x == y) ? 0 : (x > y) ? 1 : -1)

struct node
{
        int coef;
        int xexp, yexp, zexp;
        struct node *link;
};
typedef struct node *NODE;

NODE getnode()
{
        NODE x;
        x = (NODE) malloc(sizeof(struct node));
        if(x == NULL)
        {
                printf("Running out of memory \n");
                return NULL;
        }
        return x;
}

NODE attach(int coef, int xexp, int yexp, int zexp, NODE head)
{
        NODE temp, cur;
        temp = getnode();
        temp->coef = coef;
        temp->xexp = xexp;
        temp->yexp = yexp;
        temp->zexp = zexp;
        cur = head->link;
        while(cur->link != head)
        {
                cur = cur->link;
        }
        cur->link = temp;
        temp->link = head;
```

```
    return head;
}

NODE read_poly(NODE head)
{
    int i, j, coef, xexp, yexp, zexp, n;
    printf("\nEnter the no of terms in the polynomial: ");
    scanf("%d", &n);
    for(i=1; i<=n; i++)
    {
        printf("\n\tEnter the %d term: ",i);
        printf("\n\t\tCoef =  ");
        scanf("%d", &coef);
        printf("\n\t\tEnter Pow(x) Pow(y) and Pow(z): ");
        scanf("%d", &xexp);
        scanf("%d", &yexp);
        scanf("%d", &zexp);
        head = attach(coef, xexp, yexp, zexp, head);
    }
      return head;
}

void display(NODE head)
{
    NODE temp;
    if(head->link == head)
    {
         printf("\n Polynomial does not exist.");
         return;
    }
    temp = head->link;

    while(temp != head)
    {
         printf("%dx^%dy^%dz^%d", temp->coef, temp->xexp, temp->yexp, temp->zexp);
         temp = temp->link;
         if(temp != head)
             printf(" + ");
    }
}

int poly_evaluate(NODE head)
{
    int x, y, z, sum = 0;
    NODE poly;

    printf("\nEnter the value of x,y and z: ");
    scanf("%d %d %d", &x, &y, &z);

    poly = head->link;
    while(poly != head)
```

```
        {
                sum += poly->coef * pow(x,poly->xexp)* pow(y,poly->yexp) * pow(z,poly->zexp);
                poly = poly->link;
        }
        return sum;
}

NODE poly_sum(NODE head1, NODE head2, NODE head3)
{
   NODE a, b;
   int coef;
   a = head1->link;
   b = head2->link;

   while(a!=head1 && b!=head2)
   {
        while(1)
        {
                if(a->xexp == b->xexp && a->yexp == b->yexp && a->zexp == b->zexp)
                {
                        coef = a->coef + b->coef;
                        head3 = attach(coef, a->xexp, a->yexp, a->zexp, head3);
                        a = a->link;
                        b = b->link;
                        break;
                } //if ends here
                if(a->xexp!=0 || b->xexp!=0)
                {
                        switch(COMPARE(a->xexp, b->xexp))
                        {
                         case -1 :  head3 = attach(b->coef, b->xexp, b->yexp, b->zexp, head3);
                                    b = b->link;
                                    break;

                        case 0 :    if(a->yexp > b->yexp)
                                    {
                                         head3 = attach(a->coef, a->xexp, a->yexp, a->zexp, head3);
                                         a = a->link;
                                         break;
                                     }
                                     else if(a->yexp < b->yexp)
                                     {
                                          head3 = attach(b->coef, b->xexp, b->yexp, b->zexp, head3);
                                          b = b->link;
                                          break;
                                     }
                                    else if(a->zexp > b->zexp)
                                    {
                                         head3 = attach(a->coef, a->xexp, a->yexp, a->zexp, head3);
                                         a = a->link;
                                         break;
```

```
                   }
                   else if(a->zexp < b->zexp)
                  {
                         head3 = attach(b->coef, b->xexp, b->yexp, b->zexp, head3);
                         b = b->link;
                         break;
                  }
        case 1 :   head3 = attach(a->coef,a->xexp,a->yexp,a->zexp,head3);
                   a = a->link;
                   break;
        }  //switch ends here
        break;
} //if ends here
if(a->yexp!=0 || b->yexp!=0)
{
        switch(COMPARE(a->yexp, b->yexp))
        {
            case -1 :  head3 = attach(b->coef, b->xexp, b->yexp, b->zexp, head3);
                       b = b->link;
                       break;
             case 0 :  if(a->zexp > b->zexp)
                       {
                            head3 = attach(a->coef, a->xexp, a->yexp, a->zexp, head3);
                            a = a->link;
                            break;
                         }
                         else if(a->zexp < b->zexp)
                         {
                            head3 = attach(b->coef, b->xexp, b->yexp, b->zexp, head3);
                            b = b->link;
                            break;
                         }
              case 1 :   head3 = attach(a->coef, a->xexp, a->yexp, a->zexp, head3);
                         a = a->link;
                         break;
        }
        break;
}
if(a->zexp!=0 || b->zexp!=0)
{
        switch(COMPARE(a->zexp,b->zexp))
        {
                case -1 :   head3 = attach(b->coef,b->xexp,b->yexp,b->zexp,head3);
                            b = b->link;
                            break;
                case 1 :    head3 = attach(a->coef, a->xexp, a->yexp, a->zexp, head3);
                            a = a->link;
                            break;
        }
        break;
}
```

```
            }
    }
    while(a!= head1)
    {
             head3 = attach(a->coef,a->xexp,a->yexp,a->zexp,head3);
            a = a->link;
    }
    while(b!= head2)
    {
            head3 = attach(b->coef,b->xexp,b->yexp,b->zexp,head3);
            b = b->link;
    }
    return head3;
}




void main()
{
  NODE head, head1, head2, head3;
  int res, ch;
  head =  getnode();     /* For polynomial evalaution */
  head1 = getnode();    /* To hold POLY1 */
  head2 = getnode();    /* To hold POLY2 */
  head3 = getnode();    /* To hold POLYSUM */

  head->link=head;
  head1->link=head1;
  head2->link=head2;
  head3->link= head3;

  while(1)
  {
        printf("\n~~~Menu~~~");
        printf("\n1.Represent and Evaluate a Polynomial P(x,y,z)");
        printf("\n2.Find the sum of two polynomials POLY1(x,y,z)");
        printf("\nEnter your choice:");
        scanf("%d",&ch);
        switch(ch)
        {
            case 1:            printf("\n~~~~Polynomial evaluation P(x,y,z)~~~\n");
                               head = read_poly(head);
                               printf("\nRepresentation of Polynomial for evaluation: \n");
                               display(head);
                               res = poly_evaluate(head);
                               printf("\nResult of polynomial evaluation is : %d \n", res);
                               break;

            case 2:            printf("\nEnter the POLY1(x,y,z):  \n");
```

```
                         head1 = read_poly(head1);
                         printf("\nPolynomial 1 is:  \n");
                         display(head1);

                         printf("\nEnter the POLY2(x,y,z):  \n");
                         head2 = read_poly(head2);
                         printf("\nPolynomial 2 is: \n");
                         display(head2);

                         printf("\nPolynomial addition result: \n");
                         head3 = poly_sum(head1,head2,head3);
                         display(head3);
                         break;
            case 3:      exit(0);
      }
   }
}
```