Code

```c
// Author: Gokul Raj, 235, R3A, CSE
// QUESTION: Implement polynomial addition using a SLL
#include <stdio.h>
#include <stdlib.h>


// Structure to hold each term of polynomial. Also, node in singly linked list
struct Term{
    int coff;
    int exp;
    struct Term *next;
};


// Function to add two polynomials
struct Term *polyAddn(struct Term *poly1, struct Term *poly2){
    struct Term *res = NULL, *i = poly1, *j = poly2, *newNode, *lastNode = NULL;
    // while terms not exhausted
    while(i != NULL || j != NULL){
        newNode = malloc(sizeof(struct Term));
        // if exp equal, add coff and copy exp into newNode
        if(i->exp == j->exp){
            newNode->coff = i->coff + j->coff;
            newNode->exp = i->exp;
            i = i->next; j = j->next;
        }
        // if exp of term1 bigger, or 2nd poly exhausted, copy term1
        else if(i->exp > j->exp || j == NULL){
            newNode->coff = i->coff;
            newNode->exp = i->exp;
            i = i->next;
        }
        // if exp of term2 bigger, or 1st poly exhausted, copy term2
        else{
            newNode->coff = j->coff;
            newNode->exp = j->exp;
            j = j->next;
        }
        // Add node to end of result polynomial
        if(lastNode == NULL)
            res = newNode;
        else
            lastNode->next = newNode;
        lastNode = newNode;
        newNode->next = NULL;
    }
    return res;
}
```

```c
// Function to get a polynomial from user and store as linked list
struct Term *polyRead(char* query){
    printf("%s\n", query);
    int len;
    printf("Enter number of terms: ");
    scanf("%d", &len);
    struct Term *res = malloc(sizeof(struct Term));
    printf("Term 1 | Coff and Exp: ");
    scanf("%d %d", &res->coff, &res->exp);
    res->next = NULL;
    struct Term *prev = res;
    for(int i=1; i<len; i++){
        struct Term *newNode = malloc(sizeof(struct Term));
        printf("Term %d | Coff and Exp: ", i+1);
        scanf("%d %d", &newNode->coff, &newNode->exp);
        prev->next = newNode;
        newNode->next = NULL;
        prev = newNode;
    }
    return res;
}

// Function to display a polynomial
void polyShow(struct Term *poly, char *name){
    printf("%s\n", name);
    for(struct Term *p = poly; p != NULL; p=p->next){
        if(p->coff > 0)
            printf("+ %dx^%d ", p->coff, p->exp);
        else
            printf(" %dx^%d ", p->coff, p->exp);
    }
    printf("\n");
}

// Driver Code
void main(){
    struct Term *poly1 = polyRead("Enter Polynomial 1");
    struct Term *poly2 = polyRead("Enter Polynomial 2");
    polyShow(poly1, "Poly1");
    polyShow(poly2, "Poly2");
    struct Term *polysum = polyAddn(poly1, poly2);
    polyShow(polysum, "Polynomial Sum");
}

// String for testing: 4 4 3 3 2 12 1 1 0 3 6 3 8 1 9 0
// Expected OP: + 10x^3 + 3x^2 + 20x^1 + 10x^0
```

Output

```
PS C:\Users\Gokul\Desktop\MyProjects\S3_Code\DS> gcc .\polyadd_ll.c; .\a.exe
Enter Polynomial 1
Enter number of terms: 4
Term 1 | Coff and Exp: 4 3
Term 2 | Coff and Exp: 3 2
Term 3 | Coff and Exp: 12 1
Term 4 | Coff and Exp: 1 0
Enter Polynomial 2
Enter number of terms: 3
Term 1 | Coff and Exp: 6 3
Term 2 | Coff and Exp: 8 1
Term 3 | Coff and Exp: 9 0
Poly1
+ 4x^3 + 3x^2 + 12x^1 + 1x^0
Poly2
+ 6x^3 + 8x^1 + 9x^0
Polynomial Sum
+ 10x^3 + 3x^2 + 20x^1 + 10x^0
```

```
PS C:\Users\Gokul\Desktop\MyProjects\S3_Code\DS> gcc .\polyadd_ll.c; .\a.exe
Enter Polynomial 1
Enter number of terms: 3
Term 1 | Coff and Exp: 5 2
Term 2 | Coff and Exp: 7 1
Term 3 | Coff and Exp: 1 0
Enter Polynomial 2
Enter number of terms: 3
Term 1 | Coff and Exp: 5 2
Term 2 | Coff and Exp: -10 1
Term 3 | Coff and Exp: -11 0
Poly1
+ 5x^2 + 7x^1 + 1x^0
Poly2
+ 5x^2  -10x^1  -11x^0
Polynomial Sum
+ 10x^2  -3x^1  -10x^0
```