Program

```c
#include <stdio.h>
#include <stdlib.h>

struct Node {
    int coeff;
    int pow;
    struct Node* next;
};

struct Node* createNode(int c, int p) {
    struct Node* newNode = (struct Node*)malloc(sizeof(struct Node));
    newNode->coeff = c;
    newNode->pow = p;
    newNode->next = NULL;
    return newNode;
}

void insertTerm(struct Node** head, int c, int p) {
    struct Node* newNode = createNode(c, p);
    if (*head == NULL || (*head)->pow < p) {
        newNode->next = *head;
        *head = newNode;
    } else {
        struct Node* curr = *head;
        while (curr->next != NULL && curr->next->pow >= p) {
            curr = curr->next;
        }
        newNode->next = curr->next;
        curr->next = newNode;
    }
}

struct Node* addPolynomial(struct Node* head1, struct Node* head2) {
    if (head1 == NULL) return head2;
    if (head2 == NULL) return head1;

    if (head1->pow > head2->pow) {
        head1->next = addPolynomial(head1->next, head2);
        return head1;
    } else if (head1->pow < head2->pow) {
        head2->next = addPolynomial(head1, head2->next);
        return head2;
    }
    head1->coeff += head2->coeff;
    head1->next = addPolynomial(head1->next, head2->next);
    return head1;
}

void printList(struct Node* head) {
    struct Node* curr = head;
    int first = 1;
    while (curr != NULL) {
        if (!first) printf(" + ");
        printf("%dx^%d", curr->coeff, curr->pow);
        curr = curr->next;
        first = 0;
    }
    printf("\n");
}
```

Experiment No: 16
Date:

POLYNOMIAL ADDITION USING LINKED LIST

Aim:

To read two polynomials and display their sum using a linked list.

Algorithm:

1. **Start**
2. **Define a structure** `Node`**:**

```
struct Node
    int coeff
    int pow
    struct Node* next
```

3. **Create a new node:** `createNode(c, p)`

```
struct Node* newNode = (struct Node*)malloc(sizeof(struct Node))
newNode->coeff = c
newNode->pow = p
newNode->next = NULL
return newNode
```

4. **Insert a term into the list:** `insertTerm(head, c, p)`

```
if *head == NULL or (*head)->pow < p
    newNode->next = *head
    *head = newNode
else
    struct Node* curr = *head
    while curr->next != NULL and curr->next->pow >= p
        curr = curr->next
    newNode->next = curr->next
    curr->next = newNode
End If
End While
```

5. **Add two polynomials:** `addPolynomial(head1, head2)`

```
if head1 == NULL return head2
if head2 == NULL return head1
```

```c
void freeList(struct Node* head) {
    struct Node* curr = head;
    while (curr != NULL) {
        struct Node* temp = curr;
        curr = curr->next;
        free(temp);
    }
}

struct Node* inputPolynomial() {
    struct Node* head = NULL;
    int coeff, pow;
    char choice;

    printf("Enter polynomial terms (coeff, pow):\n");
    do {
        printf("Coefficient: ");
        scanf("%d", &coeff);
        printf("Power: ");
        scanf("%d", &pow);
        insertTerm(&head, coeff, pow);

        printf("Do you want to add another term? (y/n): ");
        scanf(" %c", &choice);
    } while (choice == 'y' || choice == 'Y');

    return head;
}

int main() {
    printf("Enter the first polynomial:\n");
    struct Node* head1 = inputPolynomial();

    printf("Enter the second polynomial:\n");
    struct Node* head2 = inputPolynomial();

    struct Node* head = addPolynomial(head1, head2);

    printf("Resultant Polynomial: ");
    printList(head);

    freeList(head1);
    freeList(head2);
    freeList(head);

    return 0;
}
```

Output

```
Enter the first polynomial:
Coefficient: 5
Power: 2
Do you want to add another term? (y/n): y
Coefficient: 4
Power: 1
Do you want to add another term? (y/n): y
Coefficient: 2
Power: 0
Do you want to add another term? (y/n): n
```

```
if head1->pow > head2->pow
    head1->next = addPolynomial(head1->next, head2)
    return head1
else if head1->pow < head2->pow
    head2->next = addPolynomial(head1, head2->next)
    return head2
else
    head1->coeff += head2->coeff
    head1->next = addPolynomial(head1->next, head2->next)
    return head1
End If
```

6. **Print a polynomial:** `printList(head)`

```
struct Node* curr = head
int first = 1
while curr != NULL
    if !first print " + "
    print curr->coeff + "x^" + curr->pow
    curr = curr->next
    first = 0
End While
```

7. **Free memory:** `freeList(head)`

```
struct Node* curr = head
while curr != NULL
    struct Node* temp = curr
    curr = curr->next
    free(temp)
End While
```

8. **Input polynomial terms from the user:** `inputPolynomial()`

```
struct Node* head = NULL
int coeff, pow
char choice
do
    print "Coefficient: "
    read coeff
    print "Power: "
    read pow
    insertTerm(&head, coeff, pow)

    print "Do you want to add another term? (y/n): "
    read choice
while choice == 'y' or choice == 'Y'
End Do-While
return head
```

9. **In main function:**

```
call inputPolynomial() to get head1
call inputPolynomial() to get head2
call addPolynomial(head1, head2) to get head
call printList(head)
call freeList(head1)
call freeList(head2)
call freeList(head)
```

10. **Stop**.

```
Enter the second polynomial:
Coefficient: -5
Power: 1
Do you want to add another term? (y/n): y
Coefficient: -5
Power: 0
Do you want to add another term? (y/n): n

Resultant Polynomial: 5x^2 - 1x^1 - 3x^0
```

Result:

Program has been executed successfully and obtained the output.