POLYNOMIAL ADDITION USING LINKED LIST

ALGORITHM:

Step 1: Start

Step 2: Define Structure - Create a structure Term with members: coeff, exp, and next (pointer to next term).Each node represents one term of the polynomial.

Step 3: Create Functions

(a) createTerm(coeff, exp)

– Dynamically allocate memory for a new node.

– Assign coeff and exp, set next = NULL and return the new node.

(b) insertTerm(&poly, coeff, exp)

– Create a node using createTerm.

– If list empty, make it head; else, insert at the end.

(c) displayPoly(poly)

– Traverse the list and print terms as coeff x^exp.

– Print "+" between terms.

(d) addPoly(polyA, polyB)

– Create a new list result = NULL.

– Traverse both lists:

• If expA > expB, copy term from polyA.

• If expB > expA, copy term from polyB.

• If exponents equal, add coefficients and insert.

– Copy remaining terms from either list and return result.

Step 4: Main Function (Menu Driven)

1. Insert term in Polynomial A
2. Insert term in Polynomial B
3. Display Polynomial A
4. Display Polynomial B
5. Add Polynomials
6. Display Resultant Polynomial
7. Exit

Repeat menu until user chooses Exit.

Step 5: Stop

PROGRAM:-

```
#include <stdio.h>
#include <stdlib.h>
struct Term {
   int coeff;
   int exp;
   struct Term* next;
};
struct Term* createTerm(int coeff,int exp){
   struct Term* node=(struct
Term*)malloc(sizeof(struct Term));
   node->coeff=coeff;
   node->exp=exp;
   node->next=NULL;
   return node;
}
void insertTerm(struct Term** poly,int
coeff,int exp){
   struct Term* node=createTerm(coeff,exp);
   if(*poly==NULL){*poly=node;return;}
   struct Term* temp=*poly;
   while(temp->next!=NULL)temp=temp-
>next;
   temp->next=node;
}
void displayPoly(struct Term* poly){
   if(poly==NULL){printf("Polynomial is
empty\n");return;}
   while(poly!=NULL){
      printf("%dx^%d",poly->coeff,poly-
>exp);
      poly=poly->next;
      if(poly!=NULL)printf(" + ");
   }
   printf("\n");
}

struct Term* addPoly(struct Term*
polyA,struct Term* polyB){
   struct Term* result=NULL;
   while(polyA!=NULL&&polyB!=NULL){
      if(polyA->exp>polyB->exp){
         insertTerm(&result,polyA-
>coeff,polyA->exp);
         polyA=polyA->next;
      }
      else if(polyB->exp>polyA->exp){
         insertTerm(&result,polyB-
>coeff,polyB->exp);
         polyB=polyB->next;
      }
      else{
         insertTerm(&result,polyA-
>coeff+polyB->coeff,polyA->exp);
         polyA=polyA->next;
         polyB=polyB->next;
      }
   }
   while(polyA!=NULL){
      insertTerm(&result,polyA->coeff,polyA-
>exp);
      polyA=polyA->next;
   }
   while(polyB!=NULL){
      insertTerm(&result,polyB->coeff,polyB-
>exp);
      polyB=polyB->next;
```

```
    }
    return result;
}

int main(){
    struct Term* polyA=NULL;
    struct Term* polyB=NULL;
    struct Term* result=NULL;
    int choice,coeff,exp;
    printf("=== POLYNOMIAL ADDITION USING LINKED LIST ===\n");
    do{
        printf("\nMENU:\n1.Insert term in Polynomial A\n2.Insert term in Polynomial B\n3.Display Polynomial A\n4.Display Polynomial B\n5.Add Polynomials\n6.Display Result\n7.Exit\n");
        printf("Enter your choice: ");
        scanf("%d",&choice);
        switch(choice){
            case 1:
                printf("Enter coefficient: ");
                scanf("%d",&coeff);
                printf("Enter exponent: ");
                scanf("%d",&exp);
                insertTerm(&polyA,coeff,exp);
                break;
            case 2:
                printf("Enter coefficient: ");
                scanf("%d",&coeff);
                printf("Enter exponent: ");
                scanf("%d",&exp);
                insertTerm(&polyB,coeff,exp);
                break;
            case 3:
                printf("Polynomial A: ");
                displayPoly(polyA);
                break;
            case 4:
                printf("Polynomial B: ");
                displayPoly(polyB);
                break;
            case 5:
                result=addPoly(polyA,polyB);
                printf("Polynomials added successfully!\n");
                break;
            case 6:
                printf("Resultant Polynomial: ");
                displayPoly(result);
                break;
            case 7:
                printf("Program terminated.\n");
                break;
            default:
                printf("Invalid choice! Try again.\n");
        }
    }while(choice!=7);
    return 0;
}
```

OUTPUT:-

=== POLYNOMIAL ADDITION USING LINKED LIST ===

MENU:

1.Insert term in Polynomial A

2.Insert term in Polynomial B

3.Display Polynomial A

4.Display Polynomial B

5.Add Polynomials

6.Display Result

7.Exit

Enter your choice: 1

Enter coefficient: 4

Enter exponent: 3

Enter your choice: 1

Enter coefficient: 2

Enter exponent: 1

Enter your choice: 2

Enter coefficient: 3

Enter exponent: 3

Enter your choice: 2

Enter coefficient: 5

Enter exponent: 0

Enter your choice: 3

Polynomial A: 4x^3 + 2x^1

Enter your choice: 4

Polynomial B: 3x^3 + 5x^0

Enter your choice: 5

Polynomials added successfully!

Enter your choice: 6

Resultant Polynomial: 4x^3 + 2x^1 + 3x^3 + 5x^0

Enter your choice: 7

Program terminated.