**Expt - 9**

Develop and execute a program in C using suitable data structures to create a binary tree for an expression. The tree traversals in some proper method should result in conversion of original expression into prefix, infix and postfix forms. Display the original expression along with the three different forms also.

#include <stdio.h>

#include <stdlib.h>

// Structure for a polynomial term

struct Node {

int coeff, exp;

struct Node\* next;

};

// Function to create and insert a new node

struct Node\* insert(struct Node\* head, int coeff, int exp) {

struct Node\* newNode = (struct Node\*)malloc(sizeof(struct Node));

newNode->coeff = coeff;

newNode->exp = exp;

newNode->next = NULL;

if (!head || head->exp < exp) {

newNode->next = head;

return newNode;

}

struct Node\* current = head;

while (current->next && current->next->exp >= exp)

current = current->next;

if (current->exp == exp) {

current->coeff += coeff; // Combine like terms

free(newNode);

} else {

newNode->next = current->next;

current->next = newNode;

}

return head;

}

// Function to add two polynomials

struct Node\* addPolynomials(struct Node\* poly1, struct Node\* poly2) {

while (poly2) {

poly1 = insert(poly1, poly2->coeff, poly2->exp);

poly2 = poly2->next;

}

return poly1;

}

// Function to print a polynomial

void printPolynomial(struct Node\* poly) {

while (poly) {

printf("%dx^%d", poly->coeff, poly->exp);

poly = poly->next;

if (poly) printf(" + ");

}

printf("\n");

}

int main() {

struct Node \*poly1 = NULL, \*poly2 = NULL;

int n, coeff, exp;

// Input for first polynomial

printf("Enter number of terms in first polynomial: ");

scanf("%d", &n);

for (int i = 0; i < n; i++) {

printf("Enter coeff and exp: ");

scanf("%d %d", &coeff, &exp);

poly1 = insert(poly1, coeff, exp);

}

// Input for second polynomial

printf("Enter number of terms in second polynomial: ");

scanf("%d", &n);

for (int i = 0; i < n; i++) {

printf("Enter coeff and exp: ");

scanf("%d %d", &coeff, &exp);

poly2 = insert(poly2, coeff, exp);

}

// Add the polynomials

struct Node\* sum = addPolynomials(poly1, poly2);

// Display results

printf("First Polynomial: ");

printPolynomial(poly1);

printf("Second Polynomial: ");

printPolynomial(poly2);

printf("Sum of Polynomials: ");

printPolynomial(sum);

return 0;

}