exp_7/a.c

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

struct Term {
    int coeff;
    int exp;
    struct Term* next;
};

struct Term* createTerm(int c, int e) {
    struct Term* newTerm = (struct Term*) malloc(sizeof(struct Term));
    newTerm->coeff = c;
    newTerm->exp = e;
    newTerm->next = NULL;
    return newTerm;
}

void insertTerm(struct Term** poly, int coeff, int exp) {
    struct Term* newTerm = createTerm(coeff, exp);
    if (*poly == NULL || (*poly)->exp < exp) {
        newTerm->next = *poly;
        *poly = newTerm;
    } else {
        struct Term* temp = *poly;
        while (temp->next != NULL && temp->next->exp > exp)
            temp = temp->next;
        if (temp->next != NULL && temp->next->exp == exp) {
            temp->next->coeff += coeff;
            free(newTerm);
            if (temp->next->coeff == 0) {
                struct Term* toDelete = temp->next;
                temp->next = toDelete->next;
                free(toDelete);
            }
        } else {
            newTerm->next = temp->next;
            temp->next = newTerm;
        }
    }
}

void printPoly(struct Term* poly) {
    if (poly == NULL) {
        printf("0\n");
        return;
    }
    struct Term* temp = poly;
    while (temp != NULL) {
```

```c
        if (temp->coeff != 0) {
            if (temp != poly && temp->coeff > 0)
                printf(" + ");
            else if (temp->coeff < 0)
                printf(" - ");
            int c = temp->coeff < 0 ? -temp->coeff : temp->coeff;
            if (temp->exp == 0)
                printf("%d", c);
            else if (temp->exp == 1)
                printf("%dx", c);
            else
                printf("%dx^%d", c, temp->exp);
        }
        temp = temp->next;
    }
    printf("\n");
}

struct Term* addPoly(struct Term* poly1, struct Term* poly2) {
    struct Term* result = NULL;
    struct Term* t1 = poly1, *t2 = poly2;
    while (t1 != NULL && t2 != NULL) {
        if (t1->exp > t2->exp) {
            insertTerm(&result, t1->coeff, t1->exp);
            t1 = t1->next;
        } else if (t1->exp < t2->exp) {
            insertTerm(&result, t2->coeff, t2->exp);
            t2 = t2->next;
        } else {
            insertTerm(&result, t1->coeff + t2->coeff, t1->exp);
            t1 = t1->next;
            t2 = t2->next;
        }
    }
    while (t1 != NULL) {
        insertTerm(&result, t1->coeff, t1->exp);
        t1 = t1->next;
    }
    while (t2 != NULL) {
        insertTerm(&result, t2->coeff, t2->exp);
        t2 = t2->next;
    }
    return result;
}

struct Term* multiplyPoly(struct Term* poly1, struct Term* poly2) {
    struct Term* result = NULL;
    for (struct Term* t1 = poly1; t1 != NULL; t1 = t1->next) {
        for (struct Term* t2 = poly2; t2 != NULL; t2 = t2->next) {
            insertTerm(&result, t1->coeff * t2->coeff, t1->exp + t2->exp);
```

```
        }
    }
    return result;
}

void freePoly(struct Term* poly) {
    while (poly != NULL) {
        struct Term* temp = poly;
        poly = poly->next;
        free(temp);
    }
}

int main() {
    struct Term *poly1 = NULL, *poly2 = NULL, *sum, *product;
    int n, coeff, exp;

    printf("Enter number of terms in first polynomial: ");
    scanf("%d", &n);
    printf("Enter coefficients and exponents:\n");
    for (int i = 0; i < n; i++) {
        scanf("%d %d", &coeff, &exp);
        insertTerm(&poly1, coeff, exp);
    }

    printf("Enter number of terms in second polynomial: ");
    scanf("%d", &n);
    printf("Enter coefficients and exponents:\n");
    for (int i = 0; i < n; i++) {
        scanf("%d %d", &coeff, &exp);
        insertTerm(&poly2, coeff, exp);
    }

    printf("First Polynomial: ");
    printPoly(poly1);
    printf("Second Polynomial: ");
    printPoly(poly2);

    sum = addPoly(poly1, poly2);
    printf("Sum: ");
    printPoly(sum);

    product = multiplyPoly(poly1, poly2);
    printf("Product: ");
    printPoly(product);

    freePoly(poly1);
    freePoly(poly2);
    freePoly(sum);
    freePoly(product);
```

```
149
150     return 0;
151 }
152
```