// Ankit Kumar 2K20/CO/072

#include <stdio.h>

#include <stdlib.h>

struct poly {

    float cof;

    int exp;

    struct poly \*next;

};

void input(struct poly \*\*s) {

    int size;

    struct poly \*p, \*q;

    printf("\nInput Polynomial::");

    printf("\nInput size of Polynomial:");

    scanf("%d", &size);

    for (int i = 0; i < size; i++) {

        p = malloc(sizeof(struct poly));

        p->next = NULL;

        printf("\nDegree of element:%d:", i + 1);

        scanf("%d", &(p->exp));

        printf("Coeffcient of element:%d:", i + 1);

        scanf("%f", &(p->cof));

        if (\*s == NULL)

            \*s = p;

        else {

            q = \*s;

            while (q->next != NULL) {

                q = q->next;

            }

            q->next = p;

        }

    }

}

void output(struct poly \*s) {

    struct poly \*p;

    p = s;

    if (p == NULL) printf("0");

    while (p != NULL) {

        if (p->next == NULL)

            printf("(%.2f)x^%d", p->cof, p->exp);

        else

            printf("(%.2f)x^%d+", p->cof, p->exp);

        p = p->next;

    }

}

struct poly \*polyadd(struct poly \*p1, struct poly \*p2) {

    struct poly \*p3 = NULL, \*p, \*q;

    if (p1 == NULL) {

        p3 = p2;

        return p3;

    } else if (p2 == NULL) {

        p3 = p1;

        return p3;

    }

    while (p1 != NULL || p2 != NULL) {

        p = malloc(sizeof(struct poly));

        p->next = NULL;

        if (p1 != NULL && p2 != NULL) {

            if (p1->exp == p2->exp) {

                p->cof = p1->cof + p2->cof;

                p->exp = p1->exp;

                p1 = p1->next;

                p2 = p2->next;

            } else if (p1->exp > p2->exp) {

                p->cof = p1->cof;

                p->exp = p1->exp;

                p1 = p1->next;

            } else {

                p->cof = p2->cof;

                p->exp = p2->exp;

                p2 = p2->next;

            }

        } else {

            if (p2 == NULL) {

                p->cof = p1->cof;

                p->exp = p1->exp;

                p1 = p1->next;

            } else if (p1 == NULL) {

                p->cof = p2->cof;

                p->exp = p2->exp;

                p2 = p2->next;

            }

        }

        if (p3 == NULL)

            p3 = p;

        else {

            q = p3;

            while (q->next != NULL) {

                q = q->next;

            }

            q->next = p;

        }

    }

    return p3;

}

struct poly \*polymult(struct poly \*p1, struct poly \*p2) {

    struct poly \*p3 = NULL, \*p, \*q, \*l = NULL, \*m;

    while (p1 != NULL) {

        p = p2;

        l = NULL;

        while (p != NULL) {

            q = malloc(sizeof(struct poly));

            q->cof = p1->cof \* p->cof;

            q->exp = p1->exp + p->exp;

            q->next = NULL;

            if (l == NULL) {

                l = q;

            } else {

                m = l;

                while (m->next != NULL) {

                    m = m->next;

                }

                m->next = q;

            }

            p = p->next;

        }

        p3 = polyadd(p3, l);

        p1 = p1->next;

    }

    return p3;

}

int main() {

    struct poly \*p1 = NULL, \*p2 = NULL;

    input(&p1);

    output(p1);

    input(&p2);

    output(p2);

    printf("\n\nAddition:");

    output(polyadd(p1, p2));

    printf("\nMultiplication:");

    output(polymult(p1, p2));

}

