#include <stdio.h>

#include <stdlib.h>

// 多项式的一个项

typedef struct Node

{

int coe; // 系数

int exp; // 指数

struct Node \*next;

} Node;

// 创建一个项节点

Node \*createNode(int coe, int exp)

{

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

if (newNode)

{

newNode->coe = coe, newNode->exp = exp;

newNode->next = NULL;

}

return newNode;

}

// 向多项式末尾增加一个项节点

void addNode(Node \*\*poly, int coe, int exp)

{

Node \*newNode = createNode(coe, exp);

if (\*poly == NULL) // 第一个节点

\*poly = newNode;

else

{

Node \*nail = \*poly;

while (nail->next)

nail = nail->next;

nail->next = newNode;

}

return;

}

// 求C=A+B

Node \*calC(Node \*A, Node \*B)

{

Node \*C = NULL;

Node \*nailA = A, \*nailB = B;

while (nailA && nailB)

{

if (nailA->exp > nailB->exp)

{

addNode(&C, nailA->coe, nailA->exp);

nailA = nailA->next;

}

else if (nailA->exp < nailB->exp)

{

addNode(&C, nailB->coe, nailB->exp);

nailB = nailB->next;

}

else

{

int sumCoe = nailA->coe + nailB->coe;

if (sumCoe != 0)

addNode(&C, sumCoe, nailA->exp);

nailA = nailA->next, nailB = nailB->next;

}

}

while (nailA)

{

addNode(&C, nailA->coe, nailA->exp);

nailA = nailA->next;

}

while (nailB)

{

addNode(&C, nailB->coe, nailB->exp);

nailB = nailB->next;

}

return C;

}

// 打印多项式C

void printC(Node \*C)

{

if (C == NULL)

printf("0");

else

{

Node \*nail = C;

while (nail)

{

if (nail->coe != 0) // 系数不为0

{

printf("%d", nail->coe);

if (nail->exp != 0) // 指数不为0

printf("x^%d", nail->exp);

if (nail->next)

{

if (nail->next->coe > 0)

printf(" + ");

else if (nail->next->coe < 0)

printf(" - ");

}

}

nail = nail->next;

}

}

printf("\n");

return;

}

// 释放多项式链表的内存

void freeNode(Node \*poly)

{

Node \*nail = poly;

while (nail)

{

Node \*temp = nail;

nail = nail->next;

free(temp);

}

return;

}

int main()

{

int coe, exp; // 系数，指数

// 创建多项式 A

Node \*A = NULL;

printf("请输入多项式A的系数和指数（以0 0加回车结尾）：");

while (scanf("%d%d", &coe, &exp) && (coe != 0 || exp != 0))

addNode(&A, coe, exp);

// 创建多项式 B

Node \*B = NULL;

printf("请输入多项式B的系数和指数（以0 0加回车结尾）：");

while (scanf("%d%d", &coe, &exp) && (coe != 0 || exp != 0))

addNode(&B, coe, exp);

// 计算多项式 C

Node \*C = calC(A, B);

// 打印多项式 C

printf("多项式C=A+B的和为：");

printC(C);

// 释放内存

freeNode(A), freeNode(B), freeNode(C);

return 0;

}