#include<stdio.h>

#include<stdlib.h>

struct node

{

int coefficient;

int degree;

struct node \*next;// link to the next node. NULL when the last node

struct node \*prev;// link to the previous node. NULL when the first node

};

struct node\* inputpoly(struct node\*);

struct node\* insert(struct node\*, int, int);

void printNode(struct node\* ptr);

void multiply(struct node\*, struct node\*);

main()

{

//set the heads

struct node\* start1 = NULL, \* start2 = NULL;

//implemnet inputpoly function to get the numbers

//to calculate multiplication, we need two polynomials

printf("-------- polynomial 1 --------\n");

start1 = inputpoly(start1);

printf("\n-------- polynomial 2 --------\n");

start2 = inputpoly(start2);

//each of the polynomials are printed out

printf("Polynomial 1 is : ");

printNode(start1);

printf("Polynomial 2 is : ");

printNode(start2);

//result of multiplication is printed out

multiply(start1, start2);

}

struct node\* inputpoly(struct node\* start)

{

int co, deg;

//request the numbers for coefficient and degree to be inputted until negative numbers are inputted

while(1)

{

//I thought it is better to input the coefficiet first, so it requests the number for coefficient first.

printf(" Input (coefficient) (degree) : ");

scanf("%d %d", &co, &deg);

if(deg<0 && co<0){

printf("\n Done!\n");

break;

//if the negative numbers are inputted, inputting process is stopped with a message 'Done!'

}

start = insert(start, co, deg);

}

return start;

}

//structure - insert

struct node\* insert(struct node\* start, int co, int deg)

{

struct node\* ptr, \* tmp;

//allocate the memory for the node

tmp = (struct node\*)malloc(sizeof(struct node));

tmp->coefficient = co;

tmp->degree = deg;

//for the case of list empty or degree greater than first one

if (start == NULL || deg > start->degree)

{

tmp->next = start;

start = tmp;

}

else

{

ptr = start;

//for the case of list not empty and degree less than previous one

while (ptr->next != NULL && ptr->next->degree >= deg)

ptr = ptr->next;

tmp->next = ptr->next;

ptr->next = tmp;

}

return start;

}

void printNode(struct node\* ptr)

{

//while struct pointer is not empty, allocate numbers inputted.

while (ptr != NULL)

{

printf("(%dx^%d)", ptr->coefficient, ptr->degree);

ptr = ptr->next;

if (ptr != NULL)

printf(" + ");

else

printf("\n");

}

}

void multiply(struct node\* p1, struct node\* p2)

{

struct node\* start3;

struct node\* p2\_mul = p2;

start3 = NULL;

//if one of the nodes is NULL, it notices that the result of zero polynomial.

if (p1 == NULL || p2 == NULL)

{

printf("Multiplied polynomial is zero polynomial\n");

return;

}

while (p1 != NULL)

{

p2 = p2\_mul;

//according to the rule of multiplication, it multiplies each of the coefficients and adds their degrees.

while (p2 != NULL)

{

start3 = insert(start3, p1->coefficient \* p2->coefficient, p1->degree + p2->degree);

p2 = p2->next;

}

p1 = p1->next;

}

//print out the result of multiplication

printf("Multiplied polynomial is : ");

printNode(start3);

}