#include<iostream>

using namespace std;

struct node {

int coef;

int exp;

node\* next;

node\* prev;

};

node\* createNode(int c, int e) {

node\* t = new node;

t->coef = c;

t->exp = e;

t->next = NULL;

t->prev = NULL;

return t;

}

node\* findLast(node\* head) {

node\* ptr = head;

while (ptr->next != NULL) {

ptr = ptr->next;

}

return ptr;

}

node\* createPoly() {

node\* head = NULL;

node\* ptr, \* temp;

int c, e, no, i;

cout << "Enter no. of terms: ";

cin >> no;

for (i = 0; i < no; i++) {

cout << "Enter coefficient & exponential: ";

cin >> c >> e;

temp = createNode(c, e);

if (head == NULL) {

head = temp;

temp->next = NULL;

temp->prev = NULL;

}

else {

ptr = findLast(head);

ptr->next = temp;

temp->prev = ptr;

}

}

return head;

}

node\* addition(node\* head1, node\* head2) {

node\* ptr1 = head1;

node\* ptr2 = head2;

node\* head = NULL;

node\* ptr, \* temp;

while (ptr1 != NULL && ptr2 != NULL) {

if (ptr1->exp > ptr2->exp) {

temp = createNode(ptr1->coef, ptr1->exp);

ptr1 = ptr1->next;

}

else if (ptr1->exp < ptr2->exp) {

temp = createNode(ptr2->coef, ptr2->exp);

ptr2 = ptr2->next;

}

else {

temp = createNode(ptr1->coef + ptr2->coef, ptr1->exp);

ptr1 = ptr1->next;

ptr2 = ptr2->next;

}

if (head == NULL) {

head = temp;

temp->next = NULL;

temp->prev = NULL;

}

else {

ptr = findLast(head);

ptr->next = temp;

temp->prev = ptr;

}

}

while (ptr1 != NULL) {

temp = createNode(ptr1->coef, ptr1->exp);

ptr1 = ptr1->next;

ptr = findLast(head);

ptr->next = temp;

temp->prev = ptr;

}

while (ptr2 != NULL) {

temp = createNode(ptr2->coef, ptr2->exp);

ptr2 = ptr2->next;

ptr = findLast(head);

ptr->next = temp;

temp->prev = ptr;

}

return head;

}

void display(node\* head) {

node\* ptr = head;

while (ptr->next != NULL) {

cout << ptr->coef << "x^" << ptr->exp << " + ";

ptr = ptr->next;

}

cout << ptr->coef << "x^" << ptr->exp;

}

int findTerms(node\* head) {

node\* ptr = head;

int count = 0;

while (ptr != NULL) {

count++;

ptr = ptr->next;

}

return count;

}

node\* sorting(node\* head) {

int terms = findTerms(head);

node\* ptr1,\* ptr2;

int i, j, tempCoef, tempExp;

for (i = 0, ptr1 = head; i < terms - 1; i++, ptr1 = ptr1->next)

{

for (j = i + 1, ptr2 = ptr1->next; j < terms; j++, ptr2 = ptr2->next)

{

if (ptr1->exp < ptr2->exp)

{

tempCoef = ptr1->coef;

tempExp = ptr1->exp;

ptr1->coef = ptr2->coef;

ptr1->exp = ptr2->exp;

ptr2->coef = tempCoef;

ptr2->exp = tempExp;

}

}

}

return head;

}

int main()

{

node \*head1,\*head2,\*head3,\*head4;

cout<<"\nPolynomial 1:\n";

head1=createPoly();

cout<<"\nPolynomial 2:\n";

head2=createPoly();

cout<<"\n\nPolynomial 1:\n";

display(head1);

cout<<"\n\nPolynomial 2:\n";

display(head2);

head1=sorting(head1);

head2=sorting(head2);

cout<<"\n\nPolynomial 1 after sorting:\n";

display(head1);

cout<<"\n\nPolynomial 2 after sorting:\n";

display(head2);

head3=addition(head1,head2);

cout<<"\n\nAddition:\n";

display(head3);

cout<<endl;

return 0;

}