출력 결과

텍스트이(가) 표시된 사진

자동 생성된 설명

그냥 생각했을 때 쉬울 줄 알았지만 코드를 짜보니 생각해야 할 경유가 많아 생각보다 많이 어려웠습니다.

나머지를 시도는 해보았으나 해결하지 못했습니다.

코드

#include <iostream>

using namespace std;

typedef struct Polynomial{

int hight;

float farr[100];

} Polynomial;

void add\_Poly(Polynomial A, Polynomial B, Polynomial \*ret){

int tmp=0;

if (A.hight == B.hight){ *//차수가 같은 경우*

ret->hight = A.hight;

for(int i=0; i < A.hight+1; i++){

ret->farr[i] = A.farr[i] + B.farr[i];

}

}

else if (A.hight > B.hight){ *// A의 차수가 큰 경우*

ret->hight = A.hight;

int h\_tmp = A.hight - B.hight;

for (int i = 0; i < B.hight + 1; i++) {*// 차수 +1(상수항)*

if (i == 0){

for (int j = 0; j < h\_tmp; j++){

ret->farr[tmp++] = A.farr[j];

}

}

ret->farr[tmp++] = A.farr[i + h\_tmp] + B.farr[i];

}

}

else{ *// B의 차수가 큰 경우*

ret->hight = B.hight;

int h\_tmp = B.hight - A.hight;

for (int i = 0; i < A.hight + 1; i++){ *// 차수 +1(상수항)*

if (i == 0){

for (int j = 0; j < h\_tmp; j++) {

ret->farr[tmp++] = B.farr[j];

}

}

ret->farr[tmp++] = B.farr[i + h\_tmp] + A.farr[i];

}

}

}

void sub\_Poly(Polynomial A, Polynomial B, Polynomial \*ret) {

if(A.hight == B.hight){

ret->hight = A.hight;

for(int i =0; i < A.hight+1; i++){

ret->farr[i] = A.farr[i] - B.farr[i];

}

}

else if(A.hight > B.hight){

ret->hight = A.hight;

int tmp=0, h\_tmp = A.hight - B.hight;

for(int i=0 ; i <B.hight+1; i++) {

if(i ==0){

for(int j=0; j < h\_tmp; j++){

ret->farr[tmp++] = A.farr[j];

}

}

ret->farr[tmp++] = A.farr[i+h\_tmp] - B.farr[i];

}

}

else{

ret->hight = B.hight;

int tmp=0, h\_tmp = B.hight - A.hight;

for(int i=0 ; i <A.hight+1; i++) {

if(i ==0){

for(int j=0; j < h\_tmp; j++){

ret->farr[tmp++] = -B.farr[j];

}

}

ret->farr[tmp++] = A.farr[i] - B.farr[i+h\_tmp];

}

}

}

void mult\_Poly(Polynomial A, Polynomial B, Polynomial \*ret){

int n\_tmp=0;

Polynomial \*P\_tmp;

*//곱하기*

if(A.hight >= B.hight){

P\_tmp = (Polynomial\*)malloc(sizeof(Polynomial) \* A.hight);

for(int i=0; i < A.hight; i++){ *//초기화*

for(int j=0; j < 100; j++){

P\_tmp[i].farr[j] =0;

}

}

for(int i=0; i < B.hight+1; i++){ *//모든 차수 다 곱하기*

for(int j=0; j < A.hight+1; j++){

P\_tmp[n\_tmp].farr[i+j] = B.farr[i] \* A.farr[j];

}

P\_tmp[n\_tmp].hight = B.hight+(A.hight-i);

n\_tmp++;

}

}

else{

P\_tmp = (Polynomial\*)malloc(sizeof(Polynomial) \* B.hight);

for(int i=0; i < B.hight; i++){ *//초기화*

for(int j=0; j < 100; j++){

P\_tmp[i].farr[j] =0;

}

}

for(int i=0; i < A.hight+1; i++){ *//모든 차수 다 곱하기*

for(int j=0; j < B.hight+1; j++){

P\_tmp[n\_tmp].farr[i+j] = A.farr[i] \* B.farr[j];

}

P\_tmp[n\_tmp].hight = A.hight+(B.hight-i);

n\_tmp++;

}

}

for(int i=0; i < P\_tmp[0].hight+1; i++){ *//ret 초기화*

ret->farr[i] = P\_tmp[0].farr[i];

}

ret->hight = P\_tmp[0].hight;

*//곱한 걸 더해주기*

for(int i=1; i < n\_tmp; i++){ *//P\_tmp의 크기만큼 돌려서 ret에 더해준다.*

int tmp = 0;

if(P\_tmp[i].hight == ret->hight){*//차수가 같다*

for(int j = 0 ; j< P\_tmp[i].hight + 1; j++){

ret->farr[j] += P\_tmp[i].farr[j];

}

}

else if(P\_tmp[i].hight > ret->hight){

int h\_tmp = P\_tmp[i].hight - ret->hight;

for(int j =0; j < ret->hight+1; j++){

if(j ==0){

float r\_tmp[100]={0,};

for(int q =0; q < ret->hight; q++){*//ret arr를 임시저장*

r\_tmp[q]= ret->farr[q];

ret->farr[q] = 0;

}

for(int k=0; k < h\_tmp; k++){

ret->farr[tmp++] += P\_tmp[i].farr[k];

}

for(int q = h\_tmp; q < ret->hight+1; q++){

r\_tmp[q]+= P\_tmp[i].farr[q];

}

}

ret->farr[tmp++] += P\_tmp[i].farr[j+h\_tmp];

}

ret->hight = P\_tmp[i].hight; *//더 큰 차수로 바꿔준다.*

}

else{

int h\_tmp = ret->hight- P\_tmp[i].hight;

for(int q =0; q < ret->hight+1; q++){

if(P\_tmp[i].farr[q]==0){continue;}

else{

ret->farr[q] += P\_tmp[i].farr[q];

}

}

}

}

}

void div\_Poly(Polynomial A, Polynomial B, Polynomial \*ret1, Polynomial \*ret2){*//ret1 = 몫 ret2 = 나머지*

Polynomial P\_tmp;

for(int i=0; i < 100; i++){

P\_tmp.farr[i] = 0;

}

ret1->hight = A.hight - B.hight;

for(int i=0; i< 100; i++){

ret1->farr[i] = 0;

ret2->farr[i] = 0;

}

for(int i=0; i< B.hight; i++){

int n\_tmp=0;

*//몫 저장*

ret1->farr[(A.hight) -(B.hight)] += A.farr[i] / B.farr[0];

*//B랑 ret1이랑 곱하기 tmp에 저장*

Polynomial mult\_tmp[B.hight+1];

for(int j =0; j < B.hight+1; j++){*//임시변수 초기화*

for(int k =0; k < 100; k++){

mult\_tmp[j].farr[k]=0;

}

}

for(int j =0; j <ret1->hight+1 ; j++){

for(int q= 0; q< B.hight+1; q++){

mult\_tmp[n\_tmp].farr[j+q] = ret1->farr[j] \* B.farr[q];

}

mult\_tmp[n\_tmp].hight = ret1->hight + (B.hight -j);

n\_tmp++;

}

*//P\_tmp 에 위에서 계산한 곱한 값 다 더하기*

for(int j=0; j < mult\_tmp[0].hight; j++){ *//임시변수 초기화*

P\_tmp.farr[j] = mult\_tmp[0].farr[j];

}

P\_tmp.hight = mult\_tmp[0].hight;

for(int j=1; j< n\_tmp; j++){

int tmp =0;

if(P\_tmp.hight == mult\_tmp[j].hight){

for(int k =0; k < P\_tmp.hight;k++){

P\_tmp.farr[k] += mult\_tmp[j].farr[k];

}

}

else if(P\_tmp.hight > mult\_tmp[j].hight){

int h\_gap= P\_tmp.hight - mult\_tmp[j].hight;

for(int k =h\_gap; k < P\_tmp.hight+1; k++){

P\_tmp.farr[k] += mult\_tmp[j].farr[k];

}

}

else{

int h\_gap= mult\_tmp[j].hight - P\_tmp.hight ;

float Pf\_tmp[100] = {0,};

for(int k= 0; k< P\_tmp.hight+1; k++){

Pf\_tmp[k] = P\_tmp.farr[k];

P\_tmp.farr[k] =0;

}

for(int k=0; k < mult\_tmp[j].hight+1; k++){

P\_tmp.farr[k] = mult\_tmp[j].farr[k];

}

for(int k= h\_gap; k< P\_tmp.hight+1;k++){

P\_tmp.farr[k] += Pf\_tmp[k];

}

P\_tmp.hight = mult\_tmp[j].hight;

}

}

*//A와 tmp를 뺀다.*

if(A.hight == P\_tmp.hight){

for(int j = 0; j< P\_tmp.hight+1; j++){

A.farr[j] -= P\_tmp.farr[j];

}

}

else if(A.hight > P\_tmp.hight){

int h\_tmp = A.hight - P\_tmp.hight, tmp =0;

for(int j=h\_tmp; j <P\_tmp.hight+1; j++){

A.farr[j] -= P\_tmp.farr[j];

}

}

else{

A.hight = P\_tmp.hight;

}

}

*//ret1*

for(int i=0; i < P\_tmp.hight+1; i++){

ret1->farr[i] = P\_tmp.farr[i];

}

*//끝나고 남은 A가 나머지값이다*

for(int i=0; i < 100; i++){ *//ret2 초기화*

ret2->farr[i]= 0;

}

for(int i =0; i < A.hight; i++){

ret2->farr[i] = A.farr[i];

}

ret2->hight = A.hight;

}

void print\_Polynomial(Polynomial \_in){

for (int i = 0; i < \_in.hight ; i++){

if(i==0 && \_in.farr[0]==0){continue;}

cout<<\_in.farr[i]<<".0x^"<<\_in.hight-i;

if(\_in.farr[i+1] < 0){ *//음수이면*

continue;

}

else{

cout << '+';

}

}

cout<<\_in.farr[\_in.hight] << ".0"<<endl;

}

void print\_div\_Polynomial(Polynomial \_in1, Polynomial \_in2){*//나누기 결과를 출력해주는 함수*

cout << "나머지 결과 :"<<endl;

cout << "몫 :";

for(int i=0 ; i < \_in1.hight; i++){

if( i ==0 && \_in1.farr[i]){

continue;

}

if(\_in1.farr[i] == 0){

cout << "0.0x^"<<\_in1.hight - i;

if(\_in1.farr[i+1] < 0){ *//음수이면*

continue;

}

else{

cout << '+';

}

}

else{

cout << \_in1.farr[i]<<"x^"<<\_in1.hight-i;

if(\_in2.farr[i+1] < 0){ *//음수이면*

continue;

}

else{

cout << '+';

}

}

}

cout << \_in1.farr[\_in1.hight];

cout<<endl;

if(\_in2.farr[0] != 0){

cout << "나머지 :";

for(int i=0; i < \_in2.hight+1; i++){

if(\_in1.farr[i] == 0){

cout << "0.0x^"<<\_in1.hight - i;

}

else{

cout << \_in1.farr[i]<<"x^"<<\_in1.hight-i;

}

}

}

}

int main(){

Polynomial A[3] = {{2, {1,0,-1}},{5, {3, 6, 0, 0, 0, 10}},{4, {7, 0, 5, 0, 1}} };

Polynomial B[3] = {1, {1,1},{4, {7, 0, 5, 0, 1}},{5, {3, 6, 0, 0, 0, 10}}};

Polynomial ret;

Polynomial ret1;*//몫*

Polynomial ret2;*//나머지*

for(int i=0; i< 3; i++){

cout << "A :";

print\_Polynomial(A[i]);

cout<< "B :";

print\_Polynomial(B[i]);

add\_Poly(A[i], B[i], &ret);

cout << "더하기 결과 : ";

print\_Polynomial(ret);

sub\_Poly(A[i], B[i], &ret);

cout << "빼기 결과 : ";

print\_Polynomial(ret);

mult\_Poly(A[i], B[i], &ret);

cout << "곱하기 결과 : ";

print\_Polynomial(ret);

div\_Poly(A[i], B[i], &ret1, &ret2);

print\_div\_Polynomial(ret1,ret2);

cout<<endl<<endl;

}

return 0;

}