代码：

#include <iostream>

using namespace std;

class Olnode {

private:

int row,col,data;

Olnode\* down, \* right;

public:

Olnode(int row1, int col1, int data1) {

row = row1;

col = col1;

data = data1;

down = nullptr;

right = nullptr;

}

int getrow() { return row; }

int getcol() { return col; }

int getdata() { return data; }

Olnode\* getdown() { return down; }

Olnode\* getright() { return right; }

void cright(Olnode\* right1) { right = right1; }

void cdown(Olnode\* down1) { down = down1; }

void cdata(int data1) { data = data1; }

friend ostream& operator << (ostream& o, const Olnode& L1);

};

class Crosslist {

private:

int mu, nu, tu;

Olnode\*\* rhead, \*\* chead;

public:

Crosslist(int mu1, int nu1) {

mu = mu1; nu = nu1; tu = 0;

rhead = new Olnode \* [mu];

chead = new Olnode \* [nu];

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

rhead[i] = nullptr;

}

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

chead[j] = nullptr;

}

}

Crosslist(const Crosslist & list) {

mu = list.mu ; nu = list.nu ; tu = 0;

rhead = new Olnode \* [mu];

chead = new Olnode \* [nu];

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

rhead[i] = nullptr;

}

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

chead[j] = nullptr;

}

Olnode\* temp;

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

if (list.rhead[i]) {

temp = list.rhead[i];

this->add(temp->getrow(), temp->getcol(), temp->getdata());

while (temp->getright ()) {

temp = temp->getright();

this->add(temp->getrow(), temp->getcol(), temp->getdata());

}

}

else rhead[i] = nullptr;

}

}//拷贝构造

~Crosslist() {

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

if (rhead[i]) {

Olnode\* temp= rhead[i];

for (int j = 1; j <= nu; j++) {

if (temp->getright()) {

Olnode\* tempp = temp->getright();

delete temp;

temp = tempp;

}

else {

delete temp;

break;

}

}

}

}

delete[]chead;

delete[]rhead;

}

void add(int row1, int col1, int data1) {

tu ++;

if (rhead[row1] == nullptr&& chead[col1]==nullptr) {

rhead[row1] = new Olnode(row1, col1, data1);

chead[col1] = rhead[row1];

}

else if(rhead[row1] && (!chead[col1])){

Olnode\* temp = rhead[row1];

Olnode\* temp2 = temp;

while (temp->getright()&&temp->getcol()<col1) {

temp2 = temp;

temp = temp->getright();

}

chead[col1] = new Olnode(row1, col1, data1);

if (temp->getcol() < col1) {

temp->cright(chead[col1]);

}

else {

if (temp->getcol() == col1) {

temp->cdata(temp->getdata() + data1);

}

else {

if (temp->getcol() > col1) {

temp2->cright(chead[col1]);

temp2->getright()->cright(temp);

}

}

}

}

else if ((!rhead[row1]) && chead[col1]) {

Olnode\* temp = chead[col1];

Olnode\* temp2 = temp;

while (temp->getdown()&&temp->getrow ()<row1) {

temp2 = temp;

temp = temp->getdown();

}

rhead[row1] = new Olnode(row1, col1, data1);

if(temp->getrow ()<row1)temp->cdown(rhead[row1]);

else {

if (temp->getrow() == row1)temp->cdata(temp->getdata() + data1);

else {

temp2->cdown(rhead[row1]);

temp2->getdown()->cdown(temp);

}

}

}

else if (rhead[row1] && chead[col1]) {

Olnode\* temp = rhead[row1];

Olnode\* temp2 = temp;

while (temp->getright() && temp->getcol() < col1) {

temp2 = temp;

temp = temp->getright();

}

Olnode\* tempp = chead[col1];

Olnode\* tempp2 = temp;

while (tempp->getdown() && tempp->getrow() < row1) {

tempp2 = tempp;

tempp = tempp->getdown();

}

if (temp->getcol() < col1) {

if (tempp->getrow() < row1) {

temp->cright(new Olnode(row1, col1, data1));

tempp->cdown(temp->getright());

}

else {

if (tempp->getrow() == row1) {

tempp->cdata(tempp->getdata() + data1);

temp->cright(tempp);

}

else {

temp->cright(new Olnode(row1, col1, data1));

tempp2->cdown(temp->getright());

tempp2->getdown()->cdown(tempp);

}

}

}

else {

if (temp->getcol() == col1) {

if (tempp->getrow() < row1) {

temp->cdata(temp->getdata ()+data1);

tempp->cdown(temp);

}

else {

if (tempp->getrow() == row1) {

temp->cdata(tempp->getdata() + data1);

}

else {

temp->cdata(temp->getdata() + data1);

tempp2->cdown(temp->getright());

tempp2->getdown()->cdown(tempp);

}

}

}

else {

if (tempp->getrow() < row1) {

temp2->cright(new Olnode(row1, col1, data1));

temp2->getright()->cright(temp);

tempp->cdown(temp2->getright());

}

else {

if (tempp->getrow() == row1) {

tempp->cdata(tempp->getdata() + data1);

temp2->cright(tempp);

}

else {

temp2->cright(new Olnode(row1, col1, data1));

temp2->getright()->cright(temp);

tempp2->cdown(temp2->getright());

tempp2->getdown()->cdown(tempp);

}

}

}

}

}

}//加入坐标为（row1，col1）且数据为data1的节点，若已存在，则原data加上data1；

friend Crosslist operator + (Crosslist& L1, Crosslist& L2);

friend ostream& operator << (ostream& o, const Crosslist& L1);

};

Crosslist operator + (Crosslist& L1, Crosslist& L2) {

Crosslist L3(L1.mu, L1.nu);

for (int i = 0; i < L1.mu; i++) {

if ((!L1.rhead[i]) && (!L2.rhead[i]))continue;

else {

if ((!L1.rhead[i]) && L2.rhead[i]) {

Olnode\* temp = L2.rhead[i];

L3.add(temp->getrow(), temp->getcol(), temp->getdata());

while (temp->getright()) {

temp = temp->getright();

L3.add(temp->getrow(), temp->getcol(), temp->getdata());

}

}else {

if (L1.rhead[i] && (!L2.rhead[i])) {

Olnode\* temp = L1.rhead[i];

L3.add(temp->getrow(), temp->getcol(), temp->getdata());

while (temp->getright()) {

temp = temp->getright();

L3.add(temp->getrow(), temp->getcol(), temp->getdata());

}

}else {

Olnode\* temp1 = L1.rhead[i];

Olnode\* temp2 = L2.rhead[i];

L3.add(temp1->getrow(), temp1->getcol(), temp1->getdata());

while (temp1->getright()) {

temp1 = temp1->getright();

L3.add(temp1->getrow(), temp1->getcol(), temp1->getdata());

}

L3.add(temp2->getrow(), temp2->getcol(), temp2->getdata());

while (temp2->getright()) {

temp2 = temp2->getright();

L3.add(temp2->getrow(), temp2->getcol(), temp2->getdata());

}

}

}

}

}

return L3;

}

ostream& operator << (ostream& o, const Olnode& L1) {

o << "[(" << L1.row << "," << L1.col << ")| " << L1.data << "] ";

return o;

}

ostream& operator << (ostream& o, const Crosslist& L1) {

for (int i = 0; i < L1.mu; i++) {

if (!L1.rhead[i])continue;

else {

Olnode\* temp = L1.rhead[i];

o << \*temp;

while (temp->getright()) {

temp = temp->getright();

o << \*temp;

}

}

}

return o;

}

int main()

{

cout << "[(row,col)| data]" << endl;

Crosslist L1(3,3),L2(3, 3);

L1.add(0, 0, 2);

L1.add(1, 1, 5);

L1.add(2, 1, 13);

cout <<"L1"<<endl<< L1 << endl;

L2.add(0, 0, 2);

L2.add(1, 2, 5);

L2.add(2, 2, 15);

cout <<"L2" <<endl<< L2 << endl;

Crosslist L3= L1 + L2;

cout << "L3" << endl<< L3 << endl;

}

结果：

