#include<iostream>

#include<clocale>

using namespace std;

int\*\* PereChislenie(int key);

int\* Z(int left, int right); // заполнение побочных множеств

void Watch(int \*\*M, int mM);

void WatchMn(int \*M, int mM);

void KonZap();

void menu();

int\*\* InOne(int \*\*A, int \*\*B, int mA, int mB); // Объединение А и В

int\* InOneMn(int \*A, int \*B, int mA, int mB); // Объединение множеств

int\*\* Peresek(int \*\*A, int \*\*B, int mA, int mB); // Пересечение А и В

int\* PeresekMn(int \*A, int\*B, int mA, int mB); // Пересечение множеств

int\*\* Pa3HocTb(int \*\*U, int \*\*W, int mU, int mW); // Разность А и В

int\* Pa3HocTbMn(int \*U, int \*W, int mU, int mW); // Разность множеств

int\*\* SimRas(int \*\*A, int \*\*B, int mA, int mB); // Симметрическая разность A u B

int\* SimRasMn(int \*A, int \*B, int mA, int mB); // Симметрическая разность множеств

int\*\* Inversion(int \*\*M, int mM); // Инверсия

int\*\* Exposition(int \*\*A, int \*\*B, int mA, int mB); // Композиция соответствий

int\* Obraz(int \*Y, int \*\*G, int mG); // Образ

int\* ProObraz(int \*X, int \*\*G, int mG); // ПроОбраз

int\*\* Suzko(int \*Y, int \*\*G, int mG); // Сужение

int\*\* Dekart(int \*A, int \*B, int mA, int mB, int mD);// Декартово произведение множеств

int\*\* Continue(int \*\*G, int mG); // Продолжение соответствия

int main()

{

setlocale(LC\_ALL, "rus");

int \*X = Z(1, 10); int \*Y = Z(5, 15); // для соответствия А

int \*U = Z(15, 25); int \*V = Z(10, 20); // для соответствия В

cout << "Область отправления соответствия А:\t"; WatchMn(X, X[0]); cout << endl;

cout << "Область прибытия соответствия А:\t"; WatchMn(Y, Y[0]); cout << endl;

cout << "Введите мощность графика G соответствия А:\t";

int mG;

cin >> mG;

int \*\*G = PereChislenie(mG);

cout << "Область отправления соответствия B:\t"; WatchMn(U, U[0]); cout << endl;

cout << "Область прибытия соответствия B:\t"; WatchMn(V, V[0]); cout << endl;

cout << "Введите мощность графика F соответствия B:\t";

int mF;

cin >> mF;

int \*\*F = PereChislenie(mF);

KonZap();

while (true)

{

system("cls");

cout << "A = <"; WatchMn(X, X[0]); cout << ", "; WatchMn(Y, Y[0]); cout << ", "; Watch(G, mG); cout << ">" << endl;

cout << "B = <"; WatchMn(U, U[0]); cout << ", "; WatchMn(V, V[0]); cout << ", "; Watch(F, mF); cout << ">" << endl;

menu();

int Operaziya;

cin >> Operaziya;

system("cls");

cout << "A = <"; WatchMn(X, X[0]); cout << ", "; WatchMn(Y, Y[0]); cout << ", "; Watch(G, mG); cout << ">" << endl;

cout << "B = <"; WatchMn(U, U[0]); cout << ", "; WatchMn(V, V[0]); cout << ", "; Watch(F, mF); cout << ">" << endl << endl;

switch (Operaziya)

{

case 1: // Пересечение соответствий

{

cout << "Пересечение соответствий:" << endl;

int mD = mG + mF + 1;

int \*\*D = new int\*[mD];

for (int i = 0; i <= mD; i++)

D[i] = new int[3];

D = Peresek(G, F, mG, mF);

mD = D[0][0];

int \*H = new int[X[0] + U[0]];

H = PeresekMn(X, U, X[0], U[0]);

int \*J = new int[Y[0] + V[0]];

J = PeresekMn(Y, V, Y[0], V[0]);

cout << "<"; WatchMn(H, H[0]); cout << ", "; WatchMn(J, J[0]); cout << ", "; Watch(D, mD); cout << ">" << endl;

system("pause");

}

break;

case 2: // Объединение соответствий

{

cout << "Объединение графиков :" << endl;

int mC = mG + mF + 1;

int \*\*C = new int\*[mC];

for (int i = 0; i <= mC; i++)

C[i] = new int[3];

C = InOne(G, F, mG, mF);

mC = C[0][0];

int \*K = new int[X[0] + U[0]];

K = InOneMn(X, U, X[0], U[0]);

int \*L = new int[Y[0] + V[0]];

L = InOneMn(Y, V, Y[0], V[0]);

cout << "<"; WatchMn(K, K[0]); cout << ", "; WatchMn(L, L[0]); cout << ", "; Watch(C, mC); cout << ">" << endl;

system("pause");

}

break;

case 3: // Разность A и В

{

cout << "Разность A и В :" << endl;

int mR = mG + 1;

int \*\*R = new int\*[mR];

for (int i = 0; i <= mR; i++)

R[i] = new int[3];

R = Pa3HocTb(G, F, mG, mF);

mR = R[0][0];

int \*T = new int[X[0] + 1];

T = Pa3HocTbMn(X, U, X[0], U[0]);

int \*Q = new int[Y[0] + 1];

Q = Pa3HocTbMn(Y, V, Y[0], V[0]);

cout << "<"; WatchMn(T, T[0]); cout << ", "; WatchMn(Q, Q[0]); cout << ", "; Watch(R, mR); cout << ">" << endl;

system("pause");

}

break;

case 4:// Разность B и A

{

cout << "Разность B и A :" << endl;

int mR = mF + 1;

int \*\*R = new int\*[mR];

for (int i = 0; i <= mR; i++)

R[i] = new int[3];

R = Pa3HocTb(F, G, mF, mG);

mR = R[0][0];

int \*T = new int[U[0] + 1];

T = Pa3HocTbMn(U, X, U[0], X[0]);

int \*Q = new int[V[0] + 1];

Q = Pa3HocTbMn(V, Y, V[0], Y[0]);

cout << "<"; WatchMn(T, T[0]); cout << ", "; WatchMn(Q, Q[0]); cout << ", "; Watch(R, mR); cout << ">" << endl;

system("pause");

}

break;

case 5:// Инверсия А

{

cout << "Инверсия А :" << endl;

int mI = mG;

int \*\*I = new int\*[mI];

for (int i = 0; i <= mI; i++)

I[i] = new int[3];

I = Inversion(G, mG);

mI = I[0][0];

cout << "<"; WatchMn(Y, Y[0]); cout << ", "; WatchMn(X, X[0]); cout << ", "; Watch(I, mI); cout << ">" << endl;

system("pause");

}

break;

case 6:// Инверсия В

{

cout << "Инверсия B :" << endl;

int mI = mF;

int \*\*I = new int\*[mI];

for (int i = 0; i <= mI; i++)

I[i] = new int[3];

I = Inversion(F, mF);

mI = I[0][0];

cout << "<"; WatchMn(V, V[0]); cout << ", "; WatchMn(U, U[0]); cout << ", "; Watch(I, mI); cout << ">" << endl;

system("pause");

}

break;

case 7:// Симметричная разность A u B

{

cout << "Симметричная разность A u B :" << endl;

int mS = mF + mG + 1;

int \*\*S = new int\*[mS];

for (int i = 0; i <= mS; i++)

S[i] = new int[3];

S = SimRas(F, G, mF, mG);

mS = S[0][0];

int \*O = new int[X[0] + U[0] + 2];

O = SimRasMn(X, U, X[0], U[0]);

int \*H = new int[Y[0] + V[0] + 2];

H = SimRasMn(Y, V, Y[0], V[0]);

cout << "<"; WatchMn(O, O[0]); cout << ", "; WatchMn(H, H[0]); cout << ", "; Watch(S, mS); cout << ">" << endl;

system("pause");

}

break;

case 8:// Композиция соответствий A и B

{

cout << "Композиция соответствий A и B :" << endl;

int mK = mF\*mG + 1;

int \*\*K = new int\*[mK];

for (int i = 0; i <= mK; i++)

K[i] = new int[3];

K = Exposition(G, F, mG, mF);

mK = K[0][0];

cout << "<"; WatchMn(X, X[0]); cout << ", "; WatchMn(V, V[0]); cout << ", "; Watch(K, mK); cout << ">" << endl;

system("pause");

}

break;

case 9:// Композиция соответствий B и A

{

cout << "Композиция соответствий B и A :" << endl;

int mK = mF\*mG + 1;

int \*\*K = new int\*[mK];

for (int i = 0; i <= mK; i++)

K[i] = new int[3];

K = Exposition(F, G, mF, mG);

mK = K[0][0];

cout << "<"; WatchMn(U, U[0]); cout << ", "; WatchMn(Y, Y[0]); cout << ", "; Watch(K, mK); cout << ">" << endl;

system("pause");

}

break;

case 10:// Образ А

{

cout << "Образ A :" << endl;

int mR = (Y[0]\*mG/2);

int \*R = new int[mR];

R = Obraz(Y, G, mG);

mR = R[0];

WatchMn(R, R[0]); cout << endl;

system("pause");

}

break;

case 11:// прообраз А

{

cout << "ПРОобраз A :" << endl;

int mZ = (Y[0] \* mG);

int \*Z = new int[mZ];

Z = ProObraz(X, G, mG);

mZ = Z[0];

WatchMn(Z, Z[0]); cout << endl;

system("pause");

}

break;

case 12:// образ В

{

cout << "Образ B :" << endl;

int mR = (Y[0] \* mG / 2);

int \*R = new int[mR];

R = Obraz(V, F, mF);

mR = R[0];

WatchMn(R, R[0]); cout << endl;

system("pause");

}

break;

case 13:// прообраз B

{

cout << "ПРОобраз B :" << endl;

int mZ = (U[0] \* mF);

int \*Z = new int[mZ];

Z = ProObraz(U, F, mF);

mZ = Z[0];

WatchMn(Z, Z[0]); cout << endl;

system("pause");

}

break;

case 14:// сужение А

{

cout << "Cужение А :" << endl;

int mE = mG + Y[0]\*Y[0];

int \*\*E = new int\*[mE];

for (int i = 1; i <= mE; i++)

E[i] = new int[3];

E = Suzko(Y, G, mG);

mE = E[0][0];

cout << "<"; WatchMn(X, X[0]); cout << ", "; WatchMn(Y, Y[0]); cout << ", "; Watch(E, mE); cout << ">" << endl;

system("pause");

}

break;

case 15:// продолжение А

{

cout << "Продолжение А:\n";

int mJ = mG + 2;

int \*\*J = new int\*[mJ];

for (int i = 1; i <= mJ; i++)

J[i] = new int[3];

J = Continue(G, mG);

mJ = J[0][0];

system("cls");

cout << "Продолжение А:\n";

cout << "<"; WatchMn(X, X[0]); cout << ", "; WatchMn(Y, Y[0]); cout << ", "; Watch(J, mJ); cout << ">" << endl;

system("pause");

}

break;

case 16:// сужение В

{

cout << "Cужение B :" << endl;

int mE = mF + V[0] \* V[0];

int \*\*E = new int\*[mE];

for (int i = 1; i <= mE; i++)

E[i] = new int[3];

E = Suzko(V, F, mF);

mE = E[0][0];

cout << "<"; WatchMn(U, U[0]); cout << ", "; WatchMn(V, V[0]); cout << ", "; Watch(E, mE); cout << ">" << endl;

system("pause");

}

break;

case 17:// продолжение В

{

cout << "Продолжение В\n";

int mJ = mG + 2;

int \*\*J = new int\*[mJ];

for (int i = 1; i <= mJ; i++)

J[i] = new int[3];

J = Continue(F, mF);

mJ = J[0][0];

system("cls");

cout << "Продолжение В:\n";

cout << "<"; WatchMn(U, U[0]); cout << ", "; WatchMn(V, V[0]); cout << ", "; Watch(J, mJ); cout << ">" << endl;

system("pause");

}

break;

case 0:// Выход

return 0;

}

}

}

int\*\* PereChislenie(int mM)

{

int \*\*M;

M = new int \*[mM];

for (int i = 1; i <= mM; i++)

M[i] = new int[3];

if (mM < 1)

{

cout << "Пустое соответствие!" << endl;

return M;

}

cout << "Введите пары графика соответствия:\n";

for (int i = 1; i <= mM; i++)

{

cout << "Пара №" << i << endl;

for (int j = 1; j < 3; j++)

{

cin >> M[i][j];

}

}

cout << endl;

return M;

}

int\* Z(int left, int right)

{

int \*Spisok = new int[50];

int j = 0;

for (int i = left; i <= right; i++)

{

j++;

Spisok[j] = i;

}

Spisok[0] = j;

return Spisok;

}

void Watch(int \*\*M, int mM)

{

cout << "{";

for (int i = 1; i <= mM; i++)

{

cout << "<";

for (int j = 1; j < 3; j++)

{

cout << M[i][j];

if (j == 1)

cout << ", ";

}

cout << ">";

if (i < mM)

cout << ", ";

}

cout << "}";;

}

void WatchMn(int \*M, int mM)

{

cout << "{";

for (int i = 1; i <= mM; i++)

{

cout << M[i];

if (i < mM)

cout << ", ";

}

cout << "}";

}

void KonZap()

{

cout << "Cоответствия заполнены." << endl;

system("pause");

system("cla");

}

void menu()

{

cout << endl << "Выберите операцию:" << endl;

cout << " 1. Пересечение соответствий.\n 2. Объединение соответствий.\n 3. Разность A и В.\n 4. Разность B и A.\n 5. Инверсия А.\n";

cout << " 6. Инверсия В.\n 7. Симметрическая разность A u B.\n 8. Композиция соответствий A и B.\n 9. Композиция соответствий B и A.\n";

cout << " 10. Образ А.\n 11. Прообраз А.\n 12. Образ В.\n 13. Прообраз В.\n 14. Cужение А.\n 15. Продолжение А.\n 16. Сужение В.\n";

cout << " 17. Продолжение В.\n 0. Выход.\n";

cout << "-->";

}

int\*\* InOne(int \*\*A, int \*\*B, int mA, int mB)

{

int mC = mA + mB + 1;

int i, j;

int \*\*C = new int\*[mC];

for (int i = 0; i <= mC; i++)

C[i] = new int[3];

if (mB == 0 && mA == 0)

return C;

if (mA == 0)

{

for (i = 1; i <= mB; i++)

for (j = 1; j < 3; j++)

{

C[i][j] = B[i][j];

}

C[0][0] = mB;

return C;

}

if (mB == 0)

{

for (i = 1; i <= mA; i++)

for (j = 1; j < 3; j++)

{

C[i][j] = A[i][j];

}

C[0][0] = mA;

return C;

}

for (i = 1; i <= mA; i++)

for (j = 1; j < 3; j++)

{

C[i][j] = A[i][j];

}

for (int b = 1; b <= mB; b++)

{

for (int a = 1; a <= mA; a++)

{

if (B[b][1] == A[a][1] && B[b][2] == A[a][2])

break;

if (B[b][1] != A[a][1] || B[b][2] != A[a][2])

{

if (a == (mA))

{

C[i][1] = B[b][1];

C[i][2] = B[b][2];

i++;

}

else continue;

}

}

}

C[0][0] = i - 1;

return C;

}

int\* InOneMn(int \*A, int \*B, int mA, int mB)

{

int mO = mA + mB + 2, i = 0;

int \*C = new int[mO];

for (i = 1; i<mA; i++)

C[i] = A[i];

C[i] = A[i];

for (int b = 1; b <= mB; b++)

for (int a = 1; a <= mA; a++)

{

if (B[b] == A[a])

break;

if (B[b] != A[a])

{

if (a == (mA))

{

i++;

C[i] = B[b];

}

else continue;

}

}

C[0] = i;

return C;

}

int\*\* Peresek(int \*\*A, int \*\*B, int mA, int mB)

{

int mC = mA + mB + 1;

int \*\*C = new int\*[mC];

for (int i = 0; i <= mC; i++)

C[i] = new int[3];

int i = 0;

if (mA == 0 || mB == 0)

return C;

for (int b = 1; b <= mB; b++)

for (int a = 1; a <= mA; a++)

if (B[b][1] == A[a][1] && B[b][2] == A[a][2])

{

i++;

C[i][1] = B[b][1];

C[i][2] = B[b][2];

}

C[0][0] = i;

return C;

}

int\* PeresekMn(int \*A, int\*B, int mA, int mB)

{

int \*D = new int[mB + mA];

int i = 0;

if (mA == 0 || mB == 0)

return D;

for (int b = 1; b <= mB; b++)

for (int a = 1; a <= mA; a++)

if (B[b] == A[a])

{

i++;

D[i] = B[b];

}

D[0] = i;

return D;

}

int\*\* Pa3HocTb(int \*\*U, int \*\*W, int mU, int mW)

{

int mR = mU + 1;

int \*\*R = new int\*[mR];

for (int i = 0; i <= mR; i++)

R[i] = new int[3];

int r = 0;

if (mU == 0)

return R;

if (mW == 0)

{

for (int i = 1; i <= mU; i++)

{

R[i][1] = U[i][1];

R[i][2] = U[i][2];

}

R[0][0] = mU;

return R;

}

for (int i = 1; i <= mU; i++)

for (int j = 1; j <= mW; j++)

{

if (U[i][1] == W[j][1] && U[i][2] == W[j][2])

break;

if (j == (mW))

{

r++;

R[r][1] = U[i][1];

R[r][2] = U[i][2];

}

}

R[0][0] = r;

return R;

}

int\* Pa3HocTbMn(int \*U, int \*W, int mU, int mW)

{

int \*R = new int[mU + 1];

int r = 0;

for (int i = 1; i <= mU; i++)

for (int j = 1; j <= mW; j++)

{

if (U[i] == W[j])

break;

if (j == (mW))

{

r++;

R[r] = U[i];

}

}

R[0] = r;

return R;

}

int\*\* SimRas(int \*\*A, int \*\*B, int mA, int mB)

{

int mS = mA \* mB + 2;

int \*\*S = new int\*[mS];

for (int i = 0; i <= mS; i++)

S[i] = new int[3];

int s = 0, o = 0;

if (mA == 0 && mB == 0)

return S;

if (mA == 0)

{

for (int i = 1; i <= mB; i++)

{

S[i][1] = B[i][1];

S[i][2] = B[i][2];

}

S[0][0] = mB;

return S;

}

if (mB == 0)

{

for (int i = 1; i <= mA; i++)

{

S[i][1] = A[i][1];

S[i][2] = A[i][2];

}

S[0][0] = mA;

return S;

}

for (int i = 1; i <= mA; i++)

for (int j = 1; j <= mB; j++)

{

o = 0;

if (A[i][1] == B[j][1] && A[i][2] == B[j][2])

continue;

for (int l = 1; l <= s; l++)

{

if (S[l][1] == A[i][1] && S[l][2] == A[i][2])

{

o = 1;

break;

}

}

for (int v = 1; v <= mB; v++)

{

if (A[i][1] == B[v][1] && A[i][2] == B[v][2])

{

o = 1;

break;

}

}

if (o == 1)

continue;

s++;

S[s][1] = A[i][1];

S[s][2] = A[i][2];

}

for (int i = 1; i <= mB; i++)

for (int j = 1; j <= mA; j++)

{

o = 0;

if (B[i][1] == A[j][1] && B[i][2] == A[j][2])

continue;

for (int l = 1; l <= s; l++)

{

if (S[l][1] == B[i][1] && S[l][2] == B[i][2])

{

o = 1;

break;

}

}

for (int v = 1; v <= mA; v++)

{

if (B[i][1] == A[v][1] && B[i][2] == A[v][2])

{

o = 1;

break;

}

}

if (o == 1)

continue;

s++;

S[s][1] = B[i][1];

S[s][2] = B[i][2];

}

S[0][0] = s;

return S;

}

int\* SimRasMn(int \*A, int \*B, int mA, int mB)

{

int \*S = new int[mA + mB + 2];

int s = 0, o = 0;

for (int i = 1; i <= mA; i++)

for (int j = 1; j <= mB; j++)

{

o = 0;

if (A[i] == B[j])

continue;

for (int l = 1; l <= s; l++)

{

if (S[l] == A[i])

{

o = 1;

break;

}

}

for (int v = 1; v <= mB; v++)

{

if (A[i] == B[v])

{

o = 1;

break;

}

}

if (o == 1)

continue;

s++;

S[s] = A[i];

}

for (int i = 1; i <= mA; i++)

for (int j = 1; j <= mB; j++)

{

o = 0;

for (int l = 1; l <= s; l++)

{

if (S[l] == B[j])

{

o = 1;

break;

}

}

for (int v = 1; v <= mA; v++)

{

if (A[v] == B[j])

{

o = 1;

break;

}

}

if (o == 1)

continue;

s++;

S[s] = B[j];

}

S[0] = s;

return S;

}

int\*\* Inversion(int \*\*M, int mM)

{

int \*\*I = new int\*[mM];

int i = 0;

for (int l = 0; l <= mM; l++)

I[l] = new int[3];

if (mM == 0)

return I;

for (i = 1; i <= mM; i++)

{

I[i][1] = M[i][2];

I[i][2] = M[i][1];

}

I[0][0] = mM;

return I;

}

int\*\* Exposition(int \*\*A, int \*\*B, int mA, int mB)

{

int mO = mA \* mB + 1;

int \*\*K = new int\*[mO];

int k = 0;

for (int p = 0; p <= mO; p++)

K[p] = new int[3];

if (mA == 0 || mB == 0)

return K;

for (int i = 1; i <= mA; i++)

for (int j = 1; j <= mB; j++)

{

int o = 0;

if (A[i][2] != B[j][1])

continue;

for (int l = 1; l <= k; l++)

if (A[i][1] == K[l][1] && B[j][2] == K[l][2])

{

o = 1;

break;

}

if (o == 1)

continue;

k++;

K[k][1] = A[i][1];

K[k][2] = B[j][2];

o = 0;

}

K[0][0] = k;

return K;

}

int\* Obraz(int \*Y, int \*\*G, int mG)

{

int \*O = new int[Y[0]];

int mO = 0, mR = 0;

for (int i = 2; i <= Y[0]; i++)

{

mO++;

O[mO] = Y[i];

i++;

}

int \*R = new int[mG\*mO];

for (int i = 1; i <= mO; i++)

for (int j = 1; j <= mG; j++)

if (O[i] == G[j][2])

{

mR++;

R[mR] = G[j][1];

}

R[0] = mR;

return R;

}

int\* ProObraz(int \*X, int \*\*G, int mG)

{

int \*P = new int[X[0]];

int mP = 0; int mZ = 0;

for (int i = 1; i <= X[0]; i++)

{

mP++;

P[mP] = X[i];

i++;

}

int \*Z = new int[mG\*mP];

for (int i = 1; i <= mP; i++)

for (int j = 1; j <= mG; j++)

if (P[i] == G[j][1])

{

mZ++;

Z[mZ] = G[j][2];

}

Z[0] = mZ;

return Z;

}

int\*\* Suzko(int \*Y, int \*\*G, int mG)

{

int \*S = new int[Y[0]];

int mS = 0, mR = 0;

for (int i = 2; i <= Y[0]; i++)

{

mS++;

S[mS] = Y[i];

i++;

}

int mD = Y[0] \* mS;

int \*\*D = new int \*[mD];

for (int i = 1; i <= mD; i++)

D[i] = new int[3];

D = Dekart(S, Y, mS, Y[0], mD);

mD = D[0][0];

int mE = mG + mD;

int \*\*E = new int\*[mE];

for (int i = 1; i <= mE; i++)

E[i] = new int[3];

E = Peresek(G, D, mG, mD);

return E;

}

int\*\* Dekart(int \*A, int \*B, int mA, int mB, int mD)

{

int \*\*D = new int \*[mD];

for (int i = 0; i <= mD; i++)

D[i] = new int[3];

int o = 0; int d = 0;

for (int i = 1; i <= mA; i++)

for (int j = 1; j <= mB; j++)

{

o = 0;

for (int k = 1; k < d; k++)

{

if (A[i] == D[k][1] && B[j] == D[k][2])

{

o = 1;

break;

}

}

if (o == 1)

continue;

d++;

D[d][1] = A[i];

D[d][2] = B[j];

}

D[0][0] = d;

return D;

}

int\*\* Continue(int \*\*G, int mG)

{

int mJ = mG + 2;

int \*\*J = new int\*[mJ];

for (int i = 0; i <= mJ; i++)

J[i] = new int[3];

for (int i = 1; i <= mG; i++)

{

J[i][1] = G[i][1];

J[i][2] = G[i][2];

}

cout << "Введите 2 пары для для добавления в соответствие:\n";

for (int u = mG + 1; u <= mJ; u++)

{

cout << "Пара №" << u << ":\n";

cin >> J[u][1];

cin >> J[u][2];

}

J[0][0] = mJ;

return J;

}