next step

```
// 3=XDimensionA.size()
// 4=XDimensionB.size()
double[] u  = new double[XDimensionA.size()]; // hochu cherez parametr zadavat razmernost a ne v lob
double[] u0 = new double[XDimensionA.size()];
double[] nu = new double[XDimensionB.size()];
double[][] unu = new double[XDimensionA.size()][XDimensionB.size()];
int vihod=0; // vihod =1 , vijdem iz while, t.e. naidet optimal, esli =0, to prodolzhaem

int i,j;
int temp_i, control_i=1;
int temp_j, control_j=1;

double maxElement =1;
int maxElementI=0;
int maxElementJ=0;
double sign=1;
Couple check_couple;
Cikl tcikl;

//while(maxElement>0){//ishem optimalnij plan do teh por poka vse v matrice ne budet <=0
while(vihod<3){

u=u0; // zanulaem massiv, mozhen i ne nado
//u[0]=u[1]=u[2]=0;
//nu[0]=nu[1]=nu[2]=nu[3]=0;


nu[0]=0;
u[0]=initC.get(1,1);

i=0;
j=0;
temp_i=0;
temp_j=0;

// stroim vectora u i nu:
for (int m = 0; m <XDimensionA.size(); m++) {
for (int n = m; n <XDimensionB.size(); n++) {

i=m;
j=n;
while(initX.get(i,j)>0){ // hodim po stolbcu
```

```
 if (i>0 && i>= temp_i){
  u[i]=initC.get(i+1,j+1)-nu[j];
 }
 i++;
}
temp_i=max(i,temp_i);

i=m;
j=n;
while(initX.get(i,j)>0){ // hodim po stroke
 if (j>0 && j>= temp_j){
  nu[j]=initC.get(i+1,j+1)-u[i];
 }
 j++;
}
temp_j=max(j,temp_j);
m=min(temp_i,temp_j);
} // end n
} // end m

//nasli nuzhnoe u,vivodim na ekran:
traceln("vector u");
traceln(u[0]);
traceln(u[1]);
traceln(u[2]);

//nasli nuzhnoe nu,vivodim na ekran:
traceln("vector nu");
traceln(nu[0]);
traceln(nu[1]);
traceln(nu[2]);
traceln(nu[3]);


maxElement =0;
maxElementI=0;
maxElementJ=0;

couples2.clear(); // iznachalno par net, poetomu chistim esli chto bilo...
for ( i = 0; i <couples.size(); i++){
 remove_couples( couples.get( i ) );
}
```

```
for ( i = 0; i <XDimensionA.size(); i++) {
 for ( j = 0; j <XDimensionB.size(); j++) {
  // nahodim maksimalnij element matrici (U+NU-C) i ego index:
  unu[i][j]=u[i]+nu[j]-initC.get(i+1,j+1);
  unuMatrix.set(unu[i][j],i,j);
  if (maxElement<unu[i][j]){
   maxElement=unu[i][j];
   maxElementI=i;
   maxElementJ=j;
  }
  // zapolnyaem massiv nenulevimi elementami X
  if (initX.get(i,j)>0){
   Couple couple = add_couples(i,j,initX.get(i,j));
   couples2.add(couple);
  }
 }
}


traceln("size nenulevih");
traceln(couples2.size());
traceln("Max element i ego indeksi");
traceln(maxElement);
traceln(maxElementI);
traceln(maxElementJ);

if (maxElement>0){
control_i=maxElementI+1;
control_j=maxElementJ+1;

int check_i=maxElementI;
int check_j=maxElementJ;

cikl.clear(); // iznachalno cikl pustoi, poetomu ubiraem vse elementi
for ( i = 0; i <cikles.size(); i++){
 remove_cikles( cikles.get( i ) );
}

tcikl = add_cikles(maxElementI,maxElementJ,-1);
cikl.add(tcikl);// dobavlyaem pervuju vershinu, kotoraja soderzhit maksimalnij element
sign=-1;

while(control_i>0 || control_j>0){ // ishem zamknutij cikl

for (int k = 1; k <= couples2.size(); k++) {
 check_couple=couples2.get(k-1);
//for (Couple check_couple : couples) {
  if (sign == -1){ // nechetnie shagi
   if (check_couple.j==check_j){
   tcikl = add_cikles(check_couple.i,check_couple.j,sign*(-1));
   cikl.add(tcikl);
   if(check_couple.i==maxElementI){
    control_i=control_i-(check_couple.i+1);
   }
   else{
    control_i=control_i+(check_couple.i+1);
   }
   control_j=control_j-(check_couple.j+1);
   sign=sign*(-1);

   check_i=check_couple.i;
   check_j=check_couple.j;

   couples2.remove(check_couple); // udalili iz kollekcii
   remove_couples(check_couple);  // udalili iz Klassa
   //continue; // voobshe k=0 chtob zanovo
   k=0; // chtob s pervogo elementa massiva nachali rabotat
  }
  }
  else{ // chetnie shagi
   if (check_couple.i==check_i){
   tcikl = add_cikles(check_couple.i,check_couple.j,sign*(-1));
   cikl.add(tcikl);
   control_i=control_i-(check_couple.i+1);
   control_j=control_j+(check_couple.j+1);
   sign=sign*(-1);

   check_i=check_couple.i;
   check_j=check_couple.j;

   couples2.remove(check_couple); // udalili iz kollekcii
   remove_couples(check_couple);  // udalili iz Klassa
   //continue; // voobshe k=0 chtob zanovo
   k=0; // chtob s pervogo elementa massiva nachali rabotat
```

```
        }
        } end else (return more)
    if (control_i==0 && control_j==0){// vihodim iz cikla for, nashli nuzhnij nam zamknutij cikl
        traceln("Nashli zamknutij cikl");
     break;
    }

        if (k == couples2.size() && control_i + control_j > 0){// znachit poslednij element bil vibran
nepravilno, udalyaem ego iz kollekcii i iz klassa, i nachinaem perebirat' s predidushego
        cikl.remove(tcikl); // udalili iz kollekcii ciklov poslednij nepravilnij
        remove_cikles(tcikl);  // udalili iz Klassa ciklov

        couples2.remove(check_couple); // udalili iz kollekcii
        remove_couples(check_couple);  // udalili iz Klassa

        tcikl=cikl.get(cikl.size()-1); // teper' budem iskat prodolzhenie k etomu elementu, poslednij v
cikle, posle udalenija nepravilnogo
        check_i=tcikl.i;
        check_j=tcikl.j;
        k=0; // nachnem iskat zanovo
        }
    } — end for k=1; k<=couples2.size2.
traceln("Nashli zamknutij cikl");
traceln(control_i);
traceln(control_j);
// teper' nado izmenit znachenija plana na Theta!
//nahodim theta
tcikl=cikl.get(1);
double theta =initX.get(tcikl.i,tcikl.j);
for (int k = 1; k < cikl.size(); k=k+2) {
 tcikl=cikl.get(k);
 if (theta>initX.get(tcikl.i,tcikl.j)){theta=initX.get(tcikl.i,tcikl.j);}
}
traceln("Theta");
traceln(theta);
// menyaem plan na plus i minus theta:
double tempValue;
for (int k = 0; k < cikl.size(); k++) {
 tcikl=cikl.get(k);
 tempValue=initX.get(tcikl.i,tcikl.j);
 initX.set(tempValue-tcikl.sign*theta,tcikl.i,tcikl.j);
}

double criteria=0;
for (i = 0; i <3; i++) {
 for (j = 0; j <4; j++) {
  criteria=criteria+initX.get(i,j)*initC.get(i+1,j+1);
 }
}
traceln("Criteria");
traceln(criteria);

for ( i = 0; i <couples.size(); i++){
 remove_couples( couples.get( i ) );
}
couples2.clear(); // iznachalno par net, poetomu chistim esli chto bilo...

for ( i = 0; i <cikles.size(); i++){
 remove_cikles( cikles.get( i ) );
}
cikl.clear(); // iznachalno cikl pustoi, poetomu ubiraem vse elementi

}// otnositsya k while(control_i>0 || control_j>0)
//maxElement =0;
vihod++;
}// otnositsya k if (maxElement>0)
} // otnositsya k while(maxElement>0)
```