Lampiran 1

Program *Zero Point Method*

fungsiMain.cpp

```
#include "zeropoint.h"

void main(){
 bool repeat = true;
 do{
       int s, d;
   input(s, d);
   cout<<endl<<endl;
   zeroPoint(s,d);

   exitProgram(repeat);
   cout<<endl<<endl<<endl;
 }while(repeat);
};
```

cij.h

```
#ifndef CIJ_H
#define CIJ_H

#include <stdlib.h>
#include <conio.h>
#include <iostream.h>
#include <iomanip.h>

class entry{
 public:
       int c;
   bool vLine;
   bool hLine;
   entry(){
               vLine = false;
               hLine = false;
       };
};

void exitProgram(bool& repeat){
  char exit[1];
  cout<<endl<<endl;
  cout<<"Anda ingin mengulang program?"<<endl;
  cout<<" _____________________"<<endl;
  cout<<"|tekan 0 untuk keluar|"<<endl;
  cout<<" --------------------"<<endl;
  exit[0] = getch();
  if(exit[0] == '0'){
cout<<"TERIMAKASIH telah menggunakan program ini..."<<endl;
cout<<"WINURSITA YEKTY WIRA YUDA, Departemen Matematika"<<endl;
cout<<"Fakultas Sains dan Teknologi Universitas Airlangga";
       repeat = false;
     getch();
  } //end if
};
```

```
void input(int& s, int& d){
 input1:
 cout<<"banyaknya suplay : "; cin>>s;
 cout<<"banyaknya demand : "; cin>>d;
 if(s<0 || d<0){
       cout<<"MAAF!! Banyaknya suplay dan demand tidak boleh negatif..."<<endl;
   cout<<"Tekan sebarang tombol untuk melanjutkan..."<<endl<<endl<<endl;
   getch();
   goto input1;
 }
};

#endif
```

ZEROPOINT.h

ZEROPOINT.h

```
#ifndef ZEROPOINT_H
#define ZEROPOINT_H

#include "langkah1.h"
#include "langkah2.h"
#include "langkah3.h"
#include "langkah4.h"
#include "langkah5.h"
#include "langkah6.h"
#include "modi.h"

void display(entry* c[10][10], int s, int d, int supply[10], int demand[10]){
  cout<<"TABEL TRANSPORTASI:"<<endl;
  for(int i=0;i<s+2;i++){
       if(i == s) for(int j=0;j<d;j++) cout<<"========";
      else if (i== s+1){
                      cout<<endl;
                for(int j=0;j<d;j++) cout<<demand[j]<<setw(8);
                }
      else
                for(int j=0;j<d+1;j++){
                if(j==d) cout<<"|| "<<supply[i]<<endl;
            else if(c[i][j]->c >100) cout<<"M"<<setw(8);
                 else
                                cout<<c[i][j]->c<<setw(8);
                }
   }
   cout<<endl<<endl<<endl;
};


void iterasi(entry* c[10][10], int s, int d, int supply[10], int demand[10], bool&
repeat){
  do{
          langkah4(c,s,d);
          cout<<endl<<endl<<endl;    //getch();

          langkah5(c,s,d); cout<<"LANGKAH 5: Merevisi tabel transportasi"<<endl;
          display(c,s,d,supply,demand);
          cout<<endl<<endl<<endl;          //getch();

     for(int i=0;i<s;i++)
     for(int j=0;j<d;j++){
      c[i][j]->vLine = false;
      c[i][j]->hLine = false;
     }

          langkah3(c,s,d,supply,demand, repeat);
          cout<<endl<<endl<<endl;         //getch();
```

```
  } while(repeat);
};

void displayAkhir(entry* asli[10][10], int s, int d, entry* x[10][10]){
 int sum = 0; int dummy; cout<<endl<<endl<<"KESIMPULAN:"<<endl<<endl;
 for(int i=0;i<s;i++)
 for(int j=0;j<d;j++)
       if(x[i][j]->c != -1){
     cout<<"Sumber ke-"<<(i+1)<<" ke tujuan ke-"<<(j+1)<<"= "<<x[i][j]->c<<endl;
     dummy = x[i][j]->c * asli[i][j]->c;
     sum+=dummy;
   }
 cout<<endl;cout<<"Jadi, total biaya adalah = "<<sum;
}

void zeroPoint(int s, int d){
  entry* c[10][10]; entry* x[10][10];
  entry* c1[10][10]; entry* asli[10][10];
  int supply[10]; int demand[10];
  bool continu = false;
  bool cekDeg;

  cout<<endl<<endl<<endl;
  langkah1(asli,c,c1,x,s,d,supply,demand);
  display(c,s,d,supply,demand);
  cout<<endl<<endl<<endl; //getch();

  langkah2(c,s,d);
  display(c,s,d,supply,demand);
  cout<<endl<<endl<<endl;  //getch();

  langkah3(c,s,d,supply,demand,continu);
  cout<<endl<<endl<<endl; //getch();

  if(continu) iterasi(c,s,d,supply,demand,continu);
  cout<<endl<<endl<<endl;

  langkah6(c,x,s,d,supply,demand,cekDeg); //getch();

  if(cekDeg == false){
               bool cekModi = true;
               modi(asli,c1,s,d,x,cekModi); //getch();
               if(cekModi){
               cout<<endl<<"Karena SEMUA Z(i,j)-c(i,j)<=0, maka PENYELESAIAN
OPTIMAL * "<<endl<<endl;
                              displayAkhir(asli,s,d,x);
      }
  }
  else{
        cout<<endl<<endl<<"MAAF...ini kasus DEGENERATE..tidak dapat diselesaikan
dengan zero point method.."<<endl<<endl;
  }
};

#endif
```

langkah 1.h

```
#ifndef LANGKAH1_H
#define LANGKAH1_H

#include "cij.h"

void langkah1(entry* asli[10][10], entry* c[10][10], entry* c1[10][10],entry*
x[10][10], int& s,
       int& d, int supply[10], int demand[10]){
```

```
int jmlPersediaan = 0, jmlPermintaan = 0;
int pilihan;

pilihan1:
cout<<"PILIHAN TUJUAN (Z) :"<<endl;
cout<<"1. Memaksimumkan Z"<<endl<<"2. Meminimumkan Z"<<endl;
cout<<endl<<"Pilihan Tujuan = ";
cin>>pilihan;
if(pilihan > 2 || pilihan < 1){
     cout<<endl;
    cout<<"MAAF...Pilihan 1 atau 2...";
    cout<<"Tekan sebarang tombol untuk melanjutkan..."<<endl<<endl<<endl;
    getch();
    goto pilihan1;
}
cout<<endl<<endl<<endl;
cout<<"LANGKAH 1: Membuat tabel transportasi"<<endl;
cout<<"Tabel biaya (matrix C)"<<endl;
  for(int i=0;i<s;i++)
             for(int j=0;j<d;j++){
     c[i][j] = new entry;
             cout<<"c["<<(i+1)<<"]["<<(j+1)<<"]= ";
        cin>>c[i][j]->c;
        }

  for(int i=0;i<s;i++)
             for(int j=0;j<d;j++){
     if(pilihan == 1) c[i][j]->c *= -1;
        }
cout<<endl<<endl;
cout<<"Tabel supply (matrix S)"<<endl;
for(int i=0;i<s;i++){
     cout<<"Supply"<<(i+1)<<": ";
    cin>>supply[i];
    jmlPersediaan += supply[i];
}
cout<<endl<<endl;
cout<<"Tabel demand (matrix D)"<<endl;
for(int i=0;i<d;i++){
     cout<<"Demand"<<(i+1)<<": ";
    cin>>demand[i];
    jmlPermintaan += demand[i];
}

if(jmlPermintaan < jmlPersediaan){
    for(int i=0;i<s;i++){
       c[i][d] = new entry;
     c[i][d]->c = 0;
    }
    demand[d] = (jmlPersediaan - jmlPermintaan);
    d++;
}

else if(jmlPersediaan < jmlPermintaan){
    for(int j=0;j<d;j++){
     c[s][j] = new entry;
     c[s][j]->c = 0;
    }
    supply[s] = (jmlPermintaan - jmlPersediaan);
    s++;
}

for(int i=0;i<s;i++)
     for(int j=0;j<d;j++){
     x[i][j] = new entry;
    x[i][j]->c = -1;
     c1[i][j] = new entry;
     c1[i][j]->c = c[i][j]->c;
    asli[i][j] = new entry;
    if(pilihan == 1) asli[i][j]->c = c[i][j]->c * (-1);
```

```
      else asli[i][j]->c = c[i][j]->c;
   }

};

#endif
```

langkah 2.h

```
#ifndef LANGKAH2_H
#define LANGKAH2_H

#include "cij.h"

void langkah2(entry* c[10][10], int s, int d){
  int cMin;
  cout<<"LANGKAH 2: Mereduksi tabel transportasi dengan"<<endl;
  cout<<"elemen terkecil di tiap baris dan kolom"<<endl;
 //mengenolkan baris
 for(int i=0;i<s;i++){
   cMin = 1000;
        for(int j=0;j<d;j++) if(cMin > c[i][j]->c) cMin = c[i][j]->c;
   for(int j=0;j<d;j++) c[i][j]->c = c[i][j]->c - cMin;
 }


 //mengenolkan kolom
 for(int j=0;j<d;j++){
   cMin = 1000;
        for(int i=0;i<s;i++) if(cMin > c[i][j]->c) cMin = c[i][j]->c;
   for(int i=0;i<s;i++) c[i][j]->c = c[i][j]->c - cMin;
 }
};

#endif
```

langkah 3.h

```
#ifndef LANGKAH3_H
#define LANGKAH3_H

#include "cij.h"

void langkah3(entry* c[10][10], int s, int d, int supply[10], int demand[10],
bool& repeat){
  int counter;
  repeat = false;
  bool optimalBaris = false, optimalKolom = false;
  int tidakOptimalKolom, tidakOptimalBaris;
  cout<<"LANGKAH 3: Pengecekan Kolom dan Baris"<<endl;

  cout<<"Cek Kolom"<<endl;
  tidakOptimalKolom = 0;
  for(int j=0;j<d;j++){
              cout<<"kolom ke-"<<(j+1)<<": ";
       counter = 0;
      for(int i=0;i<s;i++)
       if(c[i][j]->c == 0)
              counter += supply[i];
      if(demand[j] > counter){
       cout<<"demand > total suplay : tidak memenuhi"<<endl;
         tidakOptimalKolom++;
         for(int i=0;i<s;i++)
              if(c[i][j]->c == 0)
```

```
                for(int k=0;k<d;k++)
                        c[i][k]->hLine = true;
        }
        else{
          cout<<"demand <= total suplay : SUDAH MEMENUHI (V)"<<endl;
          }
    }

    cout<<"Cek Baris"<<endl;
    tidakOptimalBaris = 0;
    for(int i=0;i<s;i++){
                cout<<"baris ke-"<<(i+1)<<": ";
         counter = 0;
        for(int j=0;j<d;j++)
          if(c[i][j]->c == 0)
                counter += demand[j];
        if(supply[i] > counter){
          cout<<"supply > total demand : tidak memenuhi"<<endl;
           tidakOptimalBaris++;
           for(int j=0;j<d;j++)
                if(c[i][j]->c == 0)
                for(int k=0;k<s;k++)
                        c[k][j]->vLine = true;
        }
        else{
          cout<<"supply <= total demand : SUDAH MEMENUHI (V)"<<endl;
          }
    }

    if (tidakOptimalBaris > 0 || tidakOptimalKolom >0) repeat = true;
    else repeat = false;

};
#endif
```

langkah 4.h

```
#ifndef LANGKAH4_H
#define LANGKAH4_H

#include  "langkah5.h"
#include "cij.h"

bool cariNol(entry* c[10][10], int s, int d){
 for(int i=0;i<s;i++)
        for(int j=0;j<d;j++)
        if(c[i][j]->c == 0 && c[i][j]->vLine == false && c[i][j]->hLine == false)
        return true;
 return false;
};

void garisHorizontal(entry* c[10][10], int i, int d){
   for(int a=0;a<d;a++) c[i][a]->hLine = true;
};

void garisVertikal(entry* c[10][10], int j, int s){
   for(int b=0;b<s;b++) c[b][j]->vLine = true;
};


void makeLineS(entry* c[10][10], int s, int d, int i, int j, int jNol){
        int sumS = 0;
   for(int a=0;a<d;a++)
        if(c[i][a]->c == 0 && c[i][a]->vLine == false && c[i][a]->hLine == false)
        sumS++;

   if(jNol == sumS) garisHorizontal(c,i,d);
};

void makeLineD(entry* c[10][10], int s, int d, int i, int j, int jNol){
```

```
        int sumD = 0;
   for(int b=0;b<s;b++)
        if(c[b][j]->c == 0 && c[b][j]->vLine == false && c[b][j]->hLine == false)
        sumD++;

        if(jNol == sumD) garisVertikal(c,j,s);
};



void langkah4(entry* c[10][10], int s, int d){
        cout<<"LANGKAH 4: Memberi garis pada setiap elemen nol"<<endl;

 entry* c1[10][10];
 for(int i=0;i<s;i++)
 for(int j=0;j<d;j++){
        c1[i][j] = new entry;
   c1[i][j]->c = c[i][j]->c;
   c1[i][j]->vLine = c[i][j]->vLine;
   c1[i][j]->hLine = c[i][j]->hLine;
 }

  int jNol;
   if(s <= d) jNol = d;
   else jNol = s;

   if(cariNol(c1,s,d)) do{
        for(int i=0;i<s;i++)
      for(int j=0;j<d;j++)
       if(c1[i][j]->c == 0 && c1[i][j]->vLine == false && c1[i][j]->hLine ==
false){
                makeLineS(c1,s,d,i,j,jNol);
             }

      for(int j=0;j<d;j++)
      for(int i=0;i<s;i++)
       if(c1[i][j]->c == 0 && c1[i][j]->vLine == false && c1[i][j]->hLine ==
false){
                makeLineD(c1,s,d,i,j,jNol);
             }
       jNol--; if(jNol <= 1) jNol = 1;
   }while(cariNol(c1,s,d));

   entry* c2[10][10];
 for(int i=0;i<s;i++)
 for(int j=0;j<d;j++){
        c2[i][j] = new entry;
   c2[i][j]->c = c1[i][j]->c;
   c2[i][j]->vLine = c1[i][j]->vLine;
   c2[i][j]->hLine = c1[i][j]->hLine;
 }


   langkah5(c2,s,d);

   bool cekMatriksC = true;
   for(int i=0;i<s;i++)
   for(int j=0;j<d;j++)
        if (c[i][j]->c != c2[i][j]->c){
          cekMatriksC = false;
          break;
      }

   if(cekMatriksC){
        //delete c1;
      //entry* c1[10][10];
                for(int i=0;i<s;i++)
       for(int j=0;j<d;j++){
                        c1[i][j] = new entry;
                c1[i][j]->c = c[i][j]->c;
```

```
                c1[i][j]->vLine = c[i][j]->vLine;
                c1[i][j]->hLine = c[i][j]->hLine;
        }

        int jNol;
        if(s <= d) jNol = d;
        else jNol = s;

        if(cariNol(c1,s,d))    do{
          for(int j=0;j<d;j++)
          for(int i=0;i<s;i++)
                if(c1[i][j]->c == 0 && c1[i][j]->vLine == false && c1[i][j]->hLine
== false){
                makeLineD(c1,s,d,i,j,jNol);
              }

        for(int i=0;i<s;i++)
        for(int j=0;j<d;j++)
                if(c1[i][j]->c == 0 && c1[i][j]->vLine == false && c1[i][j]->hLine
== false){
                makeLineS(c1,s,d,i,j,jNol);
              }

        jNol--; if(jNol <= 1) jNol = 1;
        }while(cariNol(c1,s,d));

   }

   for(int i=0;i<s;i++){
   for(int j=0;j<d;j++){
     if(c1[i][j]->vLine == true && c1[i][j]->hLine == true) cout<<"+"<<setw(8);
     else if(c1[i][j]->vLine == true && c1[i][j]->hLine == false)
cout<<"|"<<setw(8);
     else if(c1[i][j]->hLine == true && c1[i][j]->vLine == false) cout<<"-
"<<setw(8);
     else if(c1[i][j]->c >100) cout<<"M"<<setw(8);
     else cout<<c1[i][j]->c<<setw(8);
    }
    cout<<endl;
   }

  for(int i=0;i<s;i++)
       for(int j=0;j<d;j++){
                c[i][j]->c = c1[i][j]->c;
                c[i][j]->vLine = c1[i][j]->vLine;
                c[i][j]->hLine = c1[i][j]->hLine;
        }

}

#endif
```

langkah 5.h

```
#ifndef LANGKAH5_H
#define LANGKAH5_H

#include "cij.h"

int cariMinimal(entry* c[10][10], int s, int d){
  int min = 1000;
  for(int i=0;i<s;i++)
        for(int j=0;j<d;j++)
    if(c[i][j]->vLine == false && c[i][j]->hLine == false)
        if (c[i][j]->c < min) min = c[i][j]->c;

      return min;
```

```
};

void langkah5(entry* c[10][10], int s, int d){
 int cMin = cariMinimal(c,s,d);

 for(int i=0;i<s;i++)
       for(int j=0;j<d;j++){
       if(c[i][j]->vLine == false && c[i][j]->hLine == false)
       c[i][j]->c -= cMin;
      else if(c[i][j]->vLine == true && c[i][j]->hLine == true)
       c[i][j]->c += cMin;
   }

};

#endif
```

langkah 6.h

```
#ifndef LANGKAH6_H
#define LANGKAH6_H

#include "cij.h"

int cekNolBaris(int alfa, int d, entry* c1[10][10]){
  int jnb = 0;
  for(int a=0;a<d;a++) if(c1[alfa][a]->c == 0) jnb++;
  return jnb;
};


int cekNolKolom(int beta, int s, entry* c1[10][10]){
  int jnk = 0;
  for(int a=0;a<s;a++) if(c1[a][beta]->c == 0) jnk++;
  return jnk;
};

void allotmentBaris(int alfa, entry* c1[10][10], int d, entry* x[10][10], int
sup1[10], int dem1[10]){
   int j;
   for(j=0;j<d;j++) if(c1[alfa][j]->c == 0){
        c1[alfa][j]->c = -1;
        if(sup1[alfa] < dem1[j]){
               x[alfa][j]->c = sup1[alfa];
        dem1[j] -= sup1[alfa];
        sup1[alfa] -= sup1[alfa];
        }
        else{
               x[alfa][j]->c = dem1[j];
        sup1[alfa] -= dem1[j];
        dem1[j] -= dem1[j];
        }
   }
};

void allotmentKolom(int beta, entry* c1[10][10], int s, entry* x[10][10], int
sup1[10], int dem1[10]){
   int i;
   for(i=0;i<s;i++) if(c1[i][beta]->c == 0){
        c1[i][beta]->c = -1;
        if(sup1[i] < dem1[beta]){
               x[i][beta]->c = sup1[i];
        dem1[beta] -= sup1[i];
        sup1[i] -= sup1[i];
        }
        else{
               x[i][beta]->c = dem1[beta];
```

```
        sup1[i] -= dem1[beta];
        dem1[beta] -= dem1[beta];
        }
   }
};

bool cekSuplayDemand(int sup1[10], int dem1[10], int s, int d){
  for(int i=0;i<s;i++) if(sup1[i] > 0) return true;
  for(int j=0;j<d;j++) if(dem1[j] > 0) return true;
  return false;
};

void displayAllotment(entry* x[10][10], int s, int d, int supply[10], int
demand[10]){
  cout<<"TABEL ALLOTMENT:"<<endl;
  for(int i=0;i<s+2;i++){
       if(i == s) for(int j=0;j<d;j++) cout<<"========";
      else if (i== s+1){
                      cout<<endl;
               for(int j=0;j<d;j++) cout<<demand[j]<<setw(8);
               }
      else
               for(int j=0;j<d+1;j++){
               if(j==d) cout<<"|| "<<supply[i]<<endl;
                 else{
               if(x[i][j]->c == -1) cout<<"**"<<setw(8);
                              else cout<<x[i][j]->c<<setw(8);
            }
               }
   }
   cout<<endl<<endl<<endl;
};

void cariMaximum(entry* c1[10][10], int s, int d, float& max, int& alfa, int&
beta){
 max = 0.5;
 for(int i=0;i<s;i++)
 for(int j=0;j<d;j++)
       if(c1[i][j]->c > max){
       max = c1[i][j]->c;
      alfa = i;
      beta = j;
   }

   c1[alfa][beta]->c = -10;
};

void langkah6(entry* c[10][10], entry* x[10][10], int s, int d, int supply[10],
       int demand[10], bool& cekDeg){

  int sup1[10], dem1[10]; int repeat = 0, alfa, beta;
  float max;
  bool again;
  int jnb =0, jnk =0;
  for(int i=0;i<s;i++) sup1[i] = supply[i];
  for(int j=0;j<d;j++) dem1[j] = demand[j];

  entry* c1[10][10];
  for(int i=0;i<s;i++)
  for(int j=0;j<d;j++){
   c1[i][j] = new entry;
   c1[i][j]->c = c[i][j]->c;
   c1[i][j]->vLine = c[i][j]->vLine;
   c1[i][j]->hLine = c[i][j]->hLine;
  }

  int deg = 0;
  cekDeg = false;
  for(int i=0;i<s;i++)
  for(int j=0;j<d;j++){
```

```
      if(c1[i][j]->c == 0) deg++;
}

if(deg < (s+d-1)){
      cekDeg = true;
}
else{
      allotmentProcess:
      do{
      cariMaximum(c1,s,d,max,alfa,beta);
      jnb = cekNolBaris(alfa,d,c1);
      if(jnb == 1)   allotmentBaris(alfa,c1,d,x,sup1,dem1);
      jnk = cekNolKolom(beta,s,c1);
      if(jnk == 1)   allotmentKolom(beta,c1,s,x,sup1,dem1);
      again = cekSuplayDemand(sup1,dem1,s,d);
      repeat++;

      }while(repeat<(s*d) && again);

      for(int i=0;i<s;i++)
      for(int j=0;j<d;j++){
             if(c1[i][j]->c == -10){
      c1[i][j]->c = c[i][j]->c;
        c1[i][j]->vLine = c[i][j]->vLine;
        c1[i][j]->hLine = c[i][j]->hLine;
    }
      }

      if(cekSuplayDemand(sup1,dem1,s,d)) do{
     cariMaximum(c1,s,d,max,alfa,beta);
         for(int j=0;j<d;j++)
              if(c1[alfa][j]->c == 0){
        c1[alfa][j]->c = -1;
                    if(sup1[alfa] < dem1[j]){
                          x[alfa][j]->c = sup1[alfa];
                    dem1[j] -= sup1[alfa];
                    sup1[alfa] -= sup1[alfa];
                       }
                       else{
                          x[alfa][j]->c = dem1[j];
                       sup1[alfa] -= dem1[j];
                       dem1[j] -= dem1[j];
                       }

        }

        for(int i=0;i<s;i++)
              if(c1[i][beta]->c == 0){
         c1[i][beta]->c = -1;
                    if(sup1[i] < dem1[beta]){
                           x[i][beta]->c = sup1[i];
                    dem1[beta] -= sup1[i];
                    sup1[i] -= sup1[i];
                       }
                       else{
                            x[i][beta]->c = dem1[beta];
                          sup1[i] -= dem1[beta];
                          dem1[beta] -= dem1[beta];
                       }
        }

         for(int i=0;i<s;i++)
         for(int j=0;j<d;j++){
           if(c1[i][j]->c == -10){
                             c1[i][j]->c = c[i][j]->c;
                             c1[i][j]->vLine = c[i][j]->vLine;
                             c1[i][j]->hLine = c[i][j]->hLine;
           }
         }
         repeat = 0;
         goto allotmentProcess;
```

```
        }while(cekSuplayDemand(sup1,dem1,s,d));

   displayAllotment(x,s,d,supply,demand);
  }
};


#endif
```

MODI.h

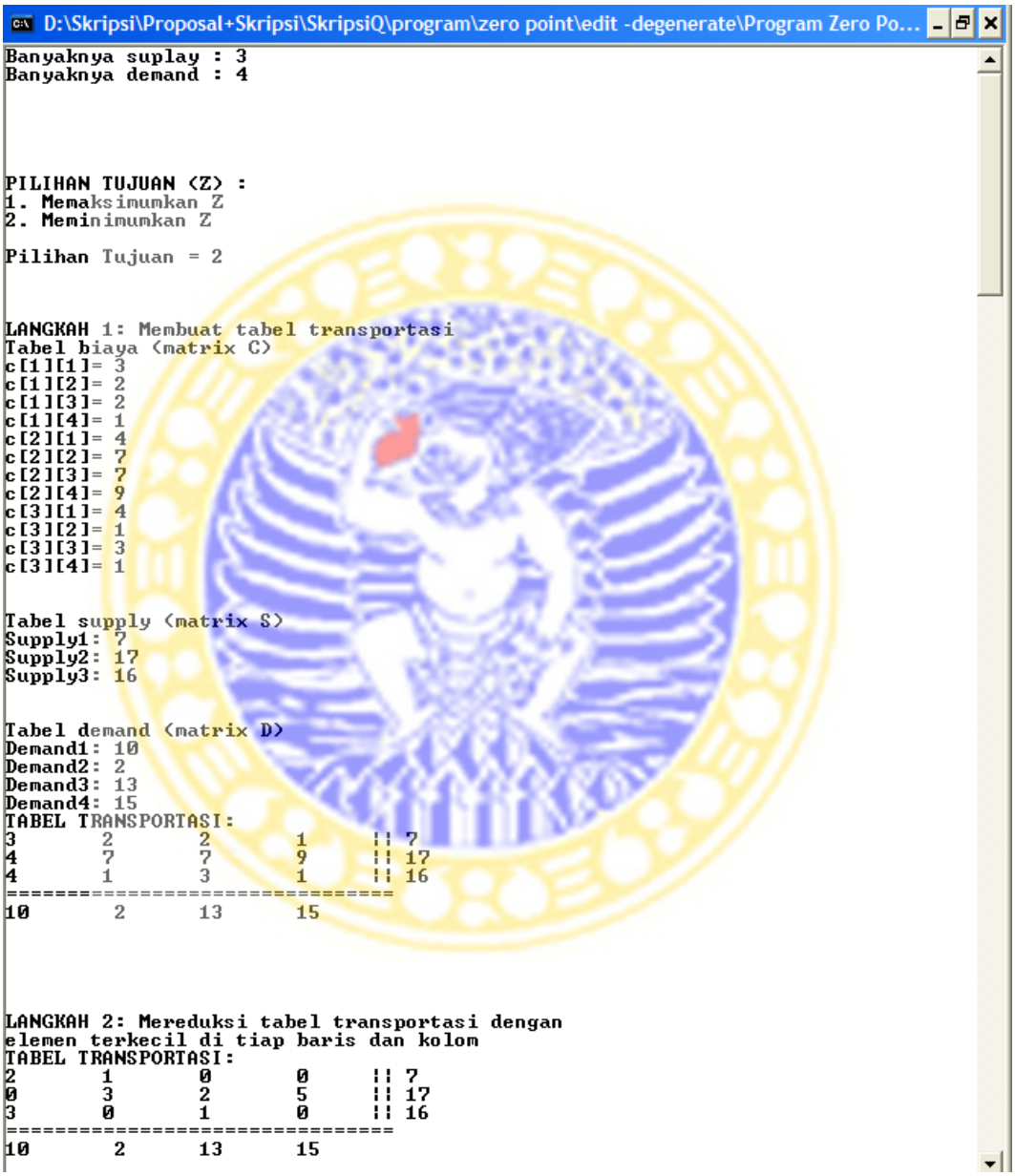
```
#define MODI_H

#include "cij.h"

class uivj{
  public:
  int a;
  bool fill;
  uivj(){
   fill = false;
  }
};

bool check(uivj* u[10], uivj* v[10], int s, int d){
       for(int i=0;i<s;i++) if(u[i]->fill == false) return true;
   for(int j=0;j<d;j++) if(v[j]->fill == false) return true;
   return false;
};

void displayModi(entry* c1[10][10], uivj* u[10], uivj* v[10], int s, int d){
  cout<<"UJI OPTIMALITAS -> Modified Distribution (MODI):"<<endl;
  for(int i=0;i<s+2;i++){
       if(i == s) for(int j=0;j<d;j++) cout<<"========";
      else if (i== s+1){
                    cout<<endl;
                for(int j=0;j<d;j++) cout<<v[j]->a<<setw(8);
                }
      else
                for(int j=0;j<d+1;j++){
                if(j==d) cout<<"|| "<<u[i]->a<<endl;
                 else{
                if(c1[i][j]->c == -1) cout<<"**"<<setw(8);
                                else cout<<c1[i][j]->c<<setw(8);
            }
                }
   }
   cout<<endl<<endl<<endl;
};

void displayNonbasis(entry* asli[10][10], entry* x[10][10], int s, int d, uivj*
u[10], uivj* v[10], bool& cekModi){
   cout<<endl<<"VARIABEL NON BASIS:"<<endl;

       for(int i=0;i<s;i++)
   for(int j=0;j<d;j++){
       if(x[i][j]->c == -1){
       cout<<"Z["<<i<<"]["<<j<<" - c["<<i<<"]["<<j<<"] = "<<((u[i]->a + v[j]->a) -
asli[i][j]->c);
        if((u[i]->a + v[j]->a) - asli[i][j]->c <= 0)
              cout<<" <= 0"<<endl;
        else cekModi = false;
      }
   }
};

void modi(entry* asli[10][10], entry* c1[10][10], int s, int d, entry*
x[10][10],bool& cekModi){
```

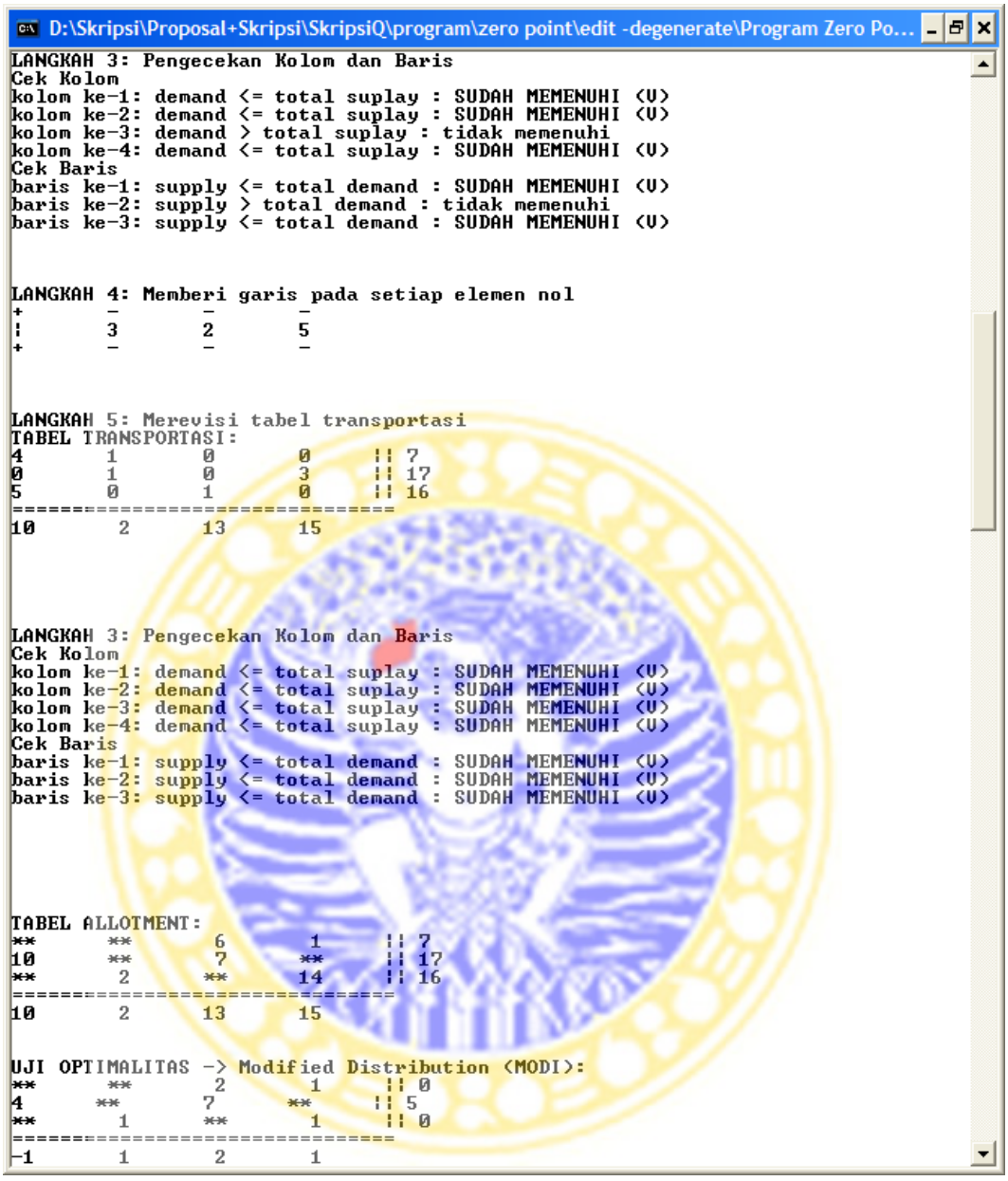
```
 uivj* u[10]; uivj* v[10];
 for(int i=0;i<s;i++) u[i] = new uivj;
 for(int j=0;j<d;j++) v[j] = new uivj;

 for(int i=0;i<s;i++)
 for(int j=0;j<d;j++)
       if(x[i][j]->c == -1) c1[i][j]->c = -1;

 do{
    u[0]->a = 0; u[0]->fill = true;
    for(int i=0;i<s;i++){
        for(int j=0;j<d;j++){
        if(u[i]->fill == true)
        if(c1[i][j]->c != -1){
                v[j]->a = c1[i][j]->c - u[i]->a;
             v[j]->fill = true;
          }
      }
    }

    for(int j=0;j<d;j++){
        for(int i=0;i<s;i++){
        if(v[j]->fill == true)
        if(c1[i][j]->c != -1){
                u[i]->a = c1[i][j]->c - v[j]->a;
             u[i]->fill = true;
          }
      }
    }
 }while(check(u,v,s,d));

 displayModi(c1,u,v,s,d);
 cout<<endl<<endl;
 displayNonbasis(asli,x,s,d,u,v,cekModi);
};


#endif
```

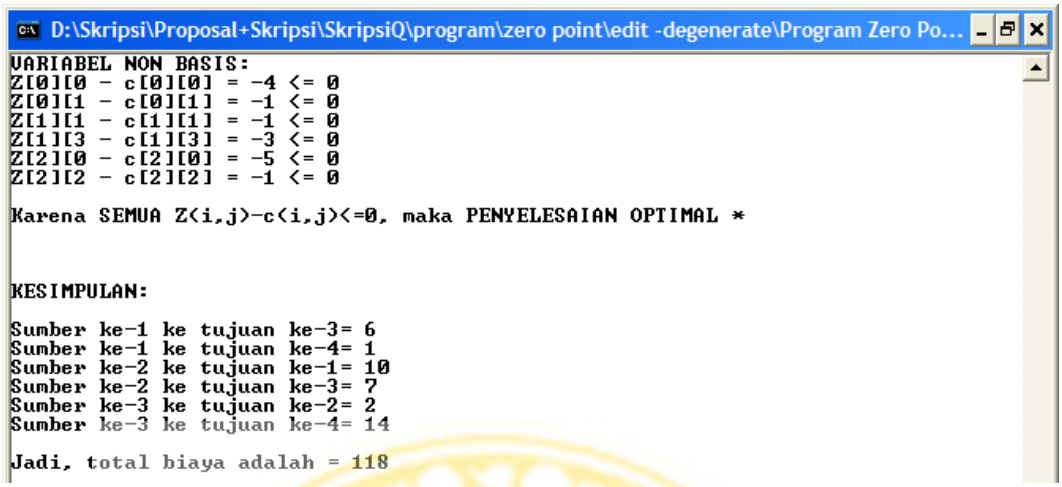
Lampiran 2

Hasil penyelesaian contoh soal dengan program *Zero Point Method*

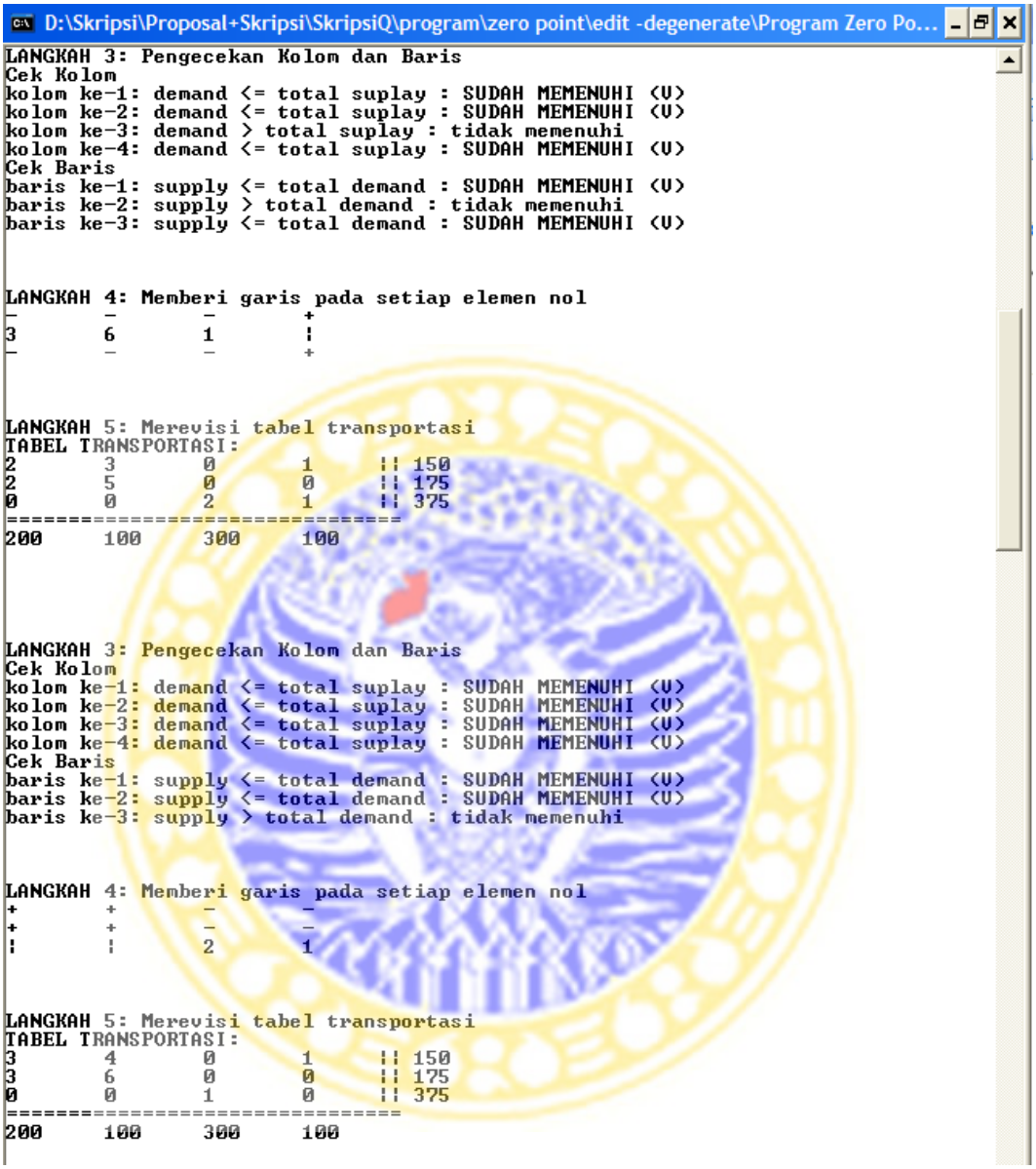
Contoh 1

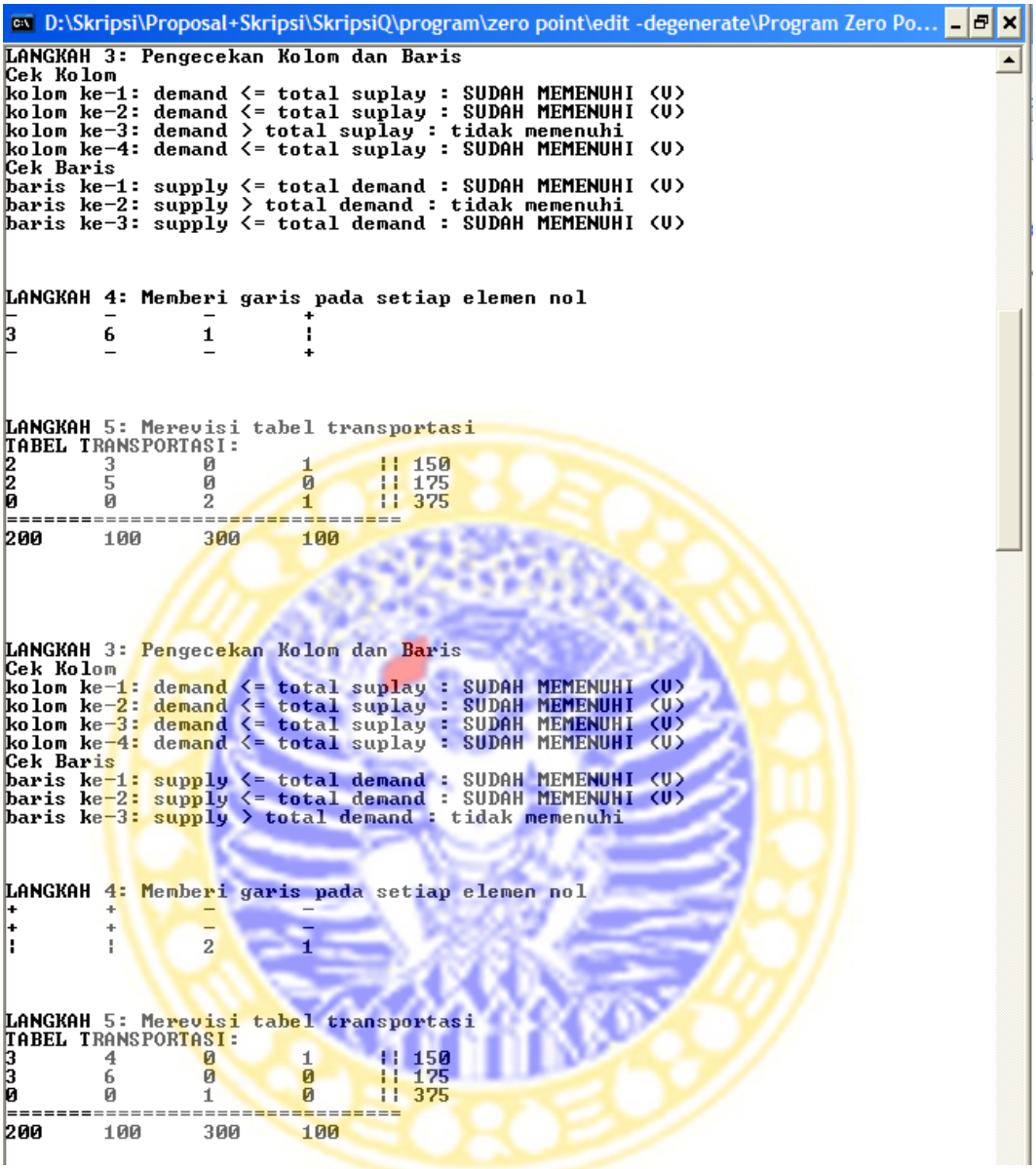


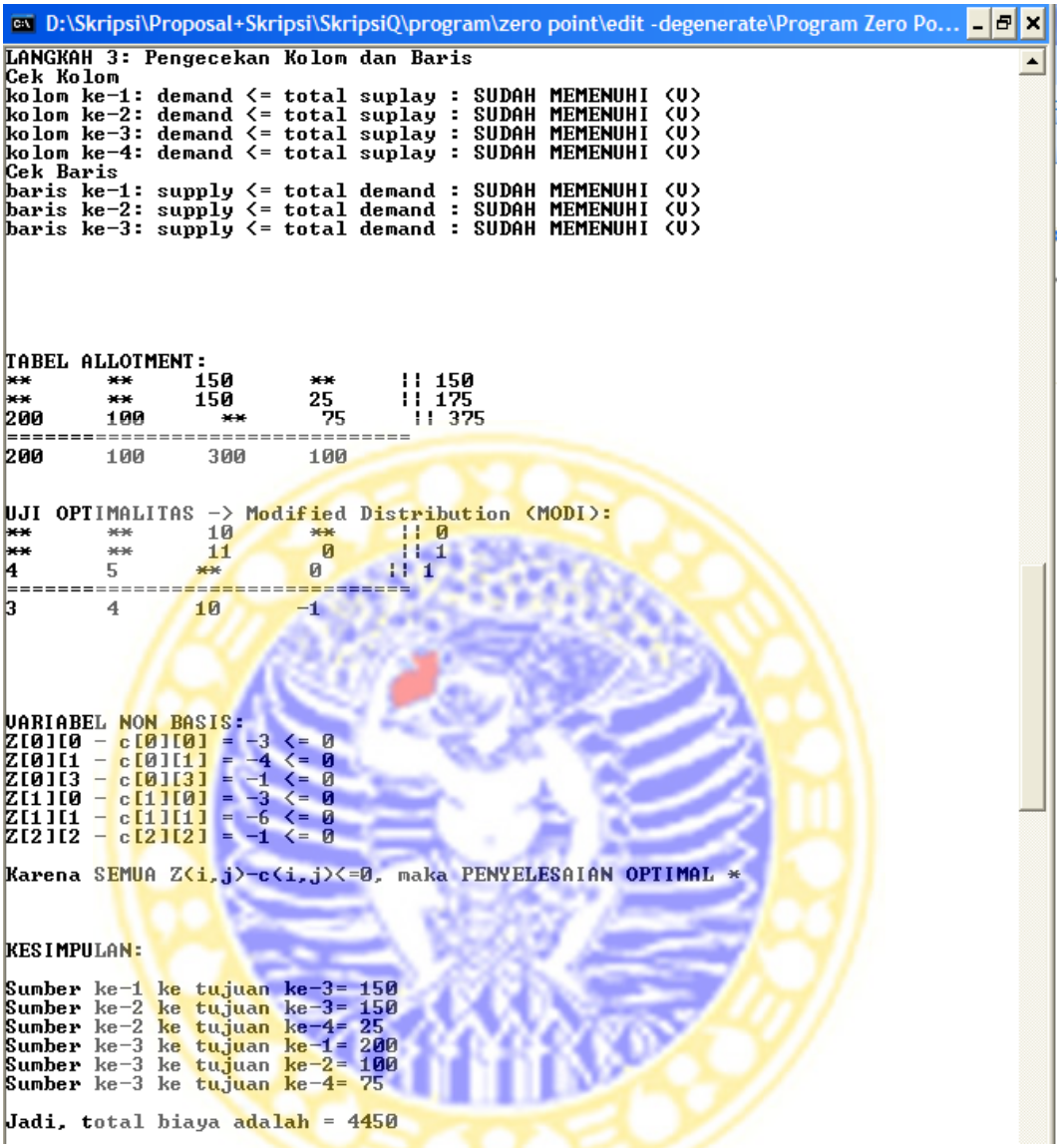

D:\Skripsi\Proposal+Skripsi\SkripsiQ\program\zero point\edit -degenerate\Program Zero Po...
LANGKAH 3: Pengecekan Kolom dan Baris
Cek Kolom
kolom ke-1: demand <= total suplay : SUDAH MEMENUHI (V)
kolom ke-2: demand <= total suplay : SUDAH MEMENUHI (V)
kolom ke-3: demand > total suplay : tidak memenuhi
kolom ke-4: demand <= total suplay : SUDAH MEMENUHI (V)
Cek Baris
baris ke-1: supply <= total demand : SUDAH MEMENUHI (V)
baris ke-2: supply > total demand : tidak memenuhi
baris ke-3: supply <= total demand : SUDAH MEMENUHI (V)
LANGKAH 4: Memberi garis pada setiap elemen nol
LANGKAH 5: Merevisi tabel transportasi
TABEL TRANSPORTASI:
4 1 0 0 || 7
0 1 0 3 || 17
5 0 1 0 || 16
10 2 13 15
LANGKAH 3: Pengecekan Kolom dan Baris
Cek Kolom
kolom ke-1: demand <= total suplay : SUDAH MEMENUHI (V)
kolom ke-2: demand <= total suplay : SUDAH MEMENUHI (V)
kolom ke-3: demand <= total suplay : SUDAH MEMENUHI (V)
kolom ke-4: demand <= total suplay : SUDAH MEMENUHI (V)
Cek Baris
baris ke-1: supply <= total demand : SUDAH MEMENUHI (V)
baris ke-2: supply <= total demand : SUDAH MEMENUHI (V)
baris ke-3: supply <= total demand : SUDAH MEMENUHI (V)
TABEL ALLOTMENT:
** ** 6 1 || 7
10 ** 7 ** || 17
** 2 ** 14 || 16
10 2 13 15
UJI OPTIMALITAS -> Modified Distribution (MODI):
** ** 2 1 || 0
4 ** 7 ** || 5
** 1 ** 1 || 0
-1 1 2 1


D:\Skripsi\Proposal+Skripsi\SkripsiQ\program\zero point\edit -degenerate\Program Zero Po...
VARIABEL NON BASIS:
Z[0][0 - c[0][0] = -4 <= 0
Z[0][1 - c[0][1] = -1 <= 0
Z[1][1 - c[1][1] = -1 <= 0
Z[1][3 - c[1][3] = -3 <= 0
Z[2][0 - c[2][0] = -5 <= 0
Z[2][2 - c[2][2] = -1 <= 0
Karena SEMUA Z(i,j)-c(i,j)<=0, maka PENYELESAIAN OPTIMAL *
KESIMPULAN:
Sumber ke-1 ke tujuan ke-3= 6
Sumber ke-1 ke tujuan ke-4= 1
Sumber ke-2 ke tujuan ke-1= 10
Sumber ke-2 ke tujuan ke-3= 7
Sumber ke-3 ke tujuan ke-2= 2
Sumber ke-3 ke tujuan ke-4= 14
Jadi, total biaya adalah = 118

Contoh 2

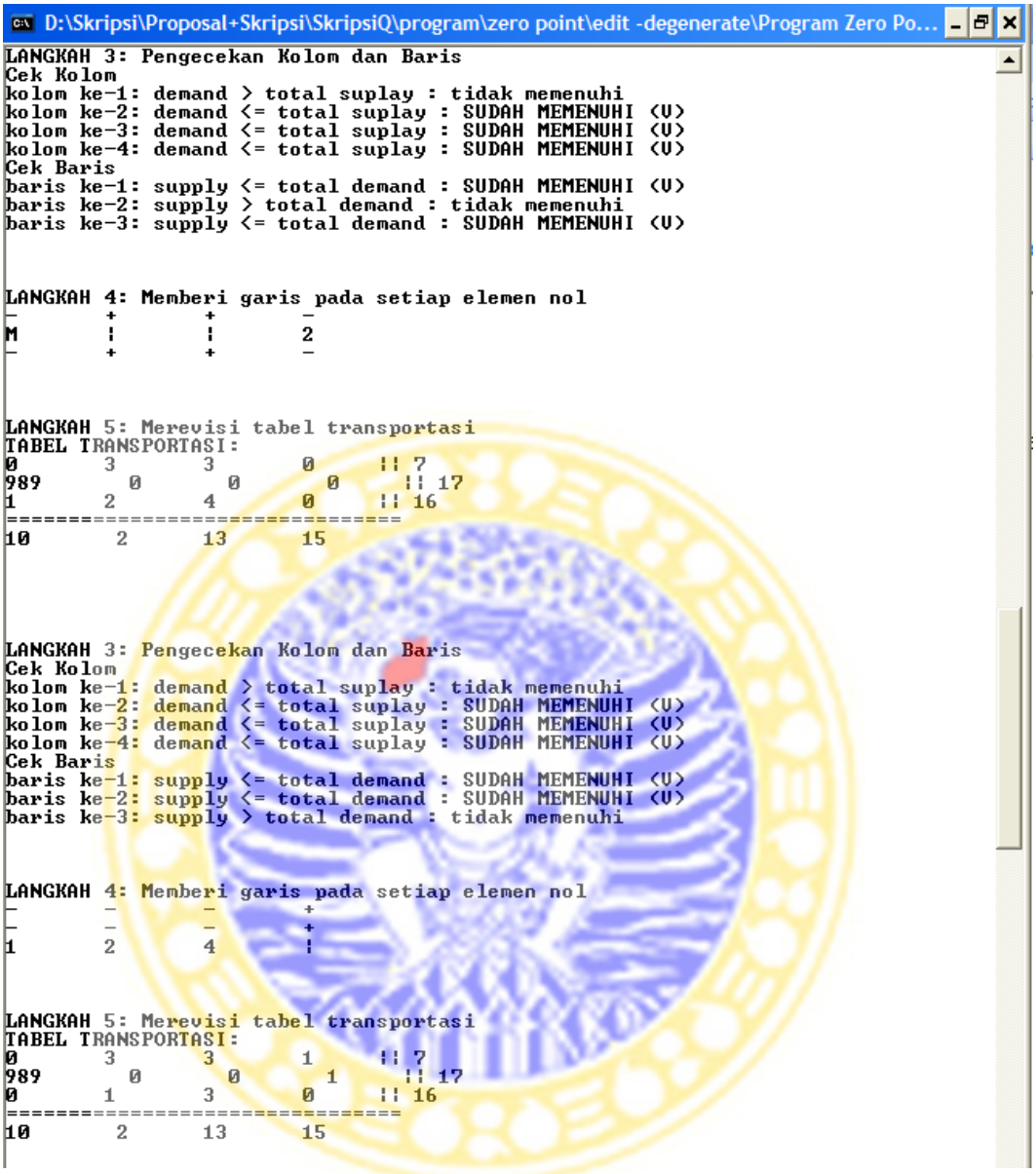

D:\Skripsi\Proposal+Skripsi\SkripsiQ\program\zero point\edit -degenerate\Program Zero Po...
LANGKAH 3: Pengecekan Kolom dan Baris
Cek Kolom
kolom ke-1: demand <= total suplay : SUDAH MEMENUHI (V)
kolom ke-2: demand <= total suplay : SUDAH MEMENUHI (V)
kolom ke-3: demand > total suplay : tidak memenuhi
kolom ke-4: demand <= total suplay : SUDAH MEMENUHI (V)
Cek Baris
baris ke-1: supply <= total demand : SUDAH MEMENUHI (V)
baris ke-2: supply > total demand : tidak memenuhi
baris ke-3: supply <= total demand : SUDAH MEMENUHI (V)
LANGKAH 4: Memberi garis pada setiap elemen nol
3 6 1
LANGKAH 5: Merevisi tabel transportasi
TABEL TRANSPORTASI:
2 3 0 1 || 150
2 5 0 0 || 175
0 0 2 1 || 375
200 100 300 100
LANGKAH 3: Pengecekan Kolom dan Baris
Cek Kolom
kolom ke-1: demand <= total suplay : SUDAH MEMENUHI (V)
kolom ke-2: demand <= total suplay : SUDAH MEMENUHI (V)
kolom ke-3: demand <= total suplay : SUDAH MEMENUHI (V)
kolom ke-4: demand <= total suplay : SUDAH MEMENUHI (V)
Cek Baris
baris ke-1: supply <= total demand : SUDAH MEMENUHI (V)
baris ke-2: supply <= total demand : SUDAH MEMENUHI (V)
baris ke-3: supply > total demand : tidak memenuhi
LANGKAH 4: Memberi garis pada setiap elemen nol
2 1
LANGKAH 5: Merevisi tabel transportasi
TABEL TRANSPORTASI:
3 4 0 1 || 150
3 6 0 0 || 175
0 0 1 0 || 375
200 100 300 100

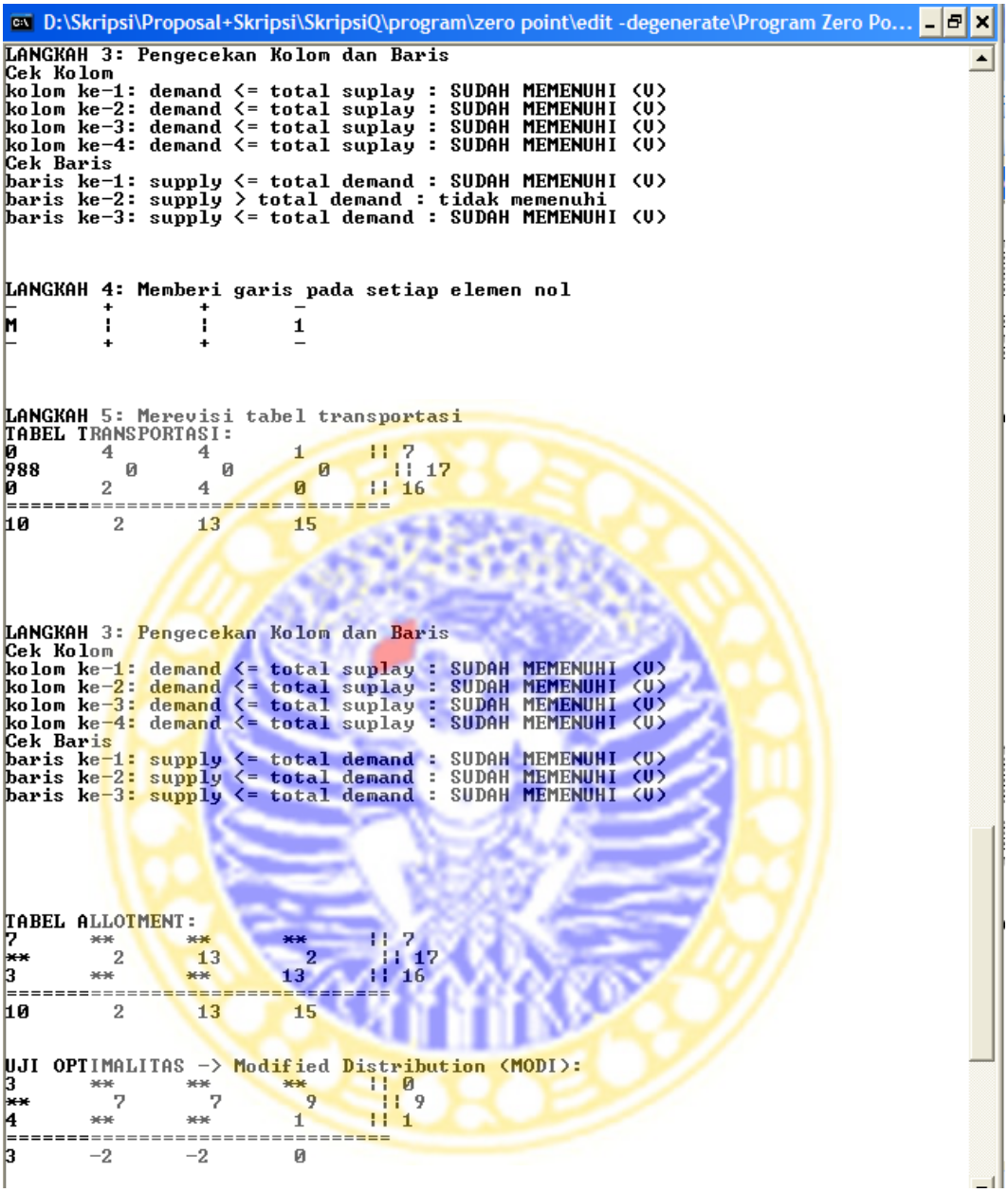

D:\Skripsi\Proposal+Skripsi\SkripsiQ\program\zero point\edit -degenerate\Program Zero Po...
LANGKAH 3: Pengecekan Kolom dan Baris
Cek Kolom
kolom ke-1: demand <= total suplay : SUDAH MEMENUHI (V)
kolom ke-2: demand <= total suplay : SUDAH MEMENUHI (V)
kolom ke-3: demand > total suplay : tidak memenuhi
kolom ke-4: demand <= total suplay : SUDAH MEMENUHI (V)
Cek Baris
baris ke-1: supply <= total demand : SUDAH MEMENUHI (V)
baris ke-2: supply > total demand : tidak memenuhi
baris ke-3: supply <= total demand : SUDAH MEMENUHI (V)
LANGKAH 4: Memberi garis pada setiap elemen nol
3 6 1
LANGKAH 5: Merevisi tabel transportasi
TABEL TRANSPORTASI:
2 3 0 1 || 150
2 5 0 0 || 175
0 0 2 1 || 375
200 100 300 100
LANGKAH 3: Pengecekan Kolom dan Baris
Cek Kolom
kolom ke-1: demand <= total suplay : SUDAH MEMENUHI (V)
kolom ke-2: demand <= total suplay : SUDAH MEMENUHI (V)
kolom ke-3: demand <= total suplay : SUDAH MEMENUHI (V)
kolom ke-4: demand <= total suplay : SUDAH MEMENUHI (V)
Cek Baris
baris ke-1: supply <= total demand : SUDAH MEMENUHI (V)
baris ke-2: supply <= total demand : SUDAH MEMENUHI (V)
baris ke-3: supply > total demand : tidak memenuhi
LANGKAH 4: Memberi garis pada setiap elemen nol
2 1
LANGKAH 5: Merevisi tabel transportasi
TABEL TRANSPORTASI:
3 4 0 1 || 150
3 6 0 0 || 175
0 0 1 0 || 375
200 100 300 100

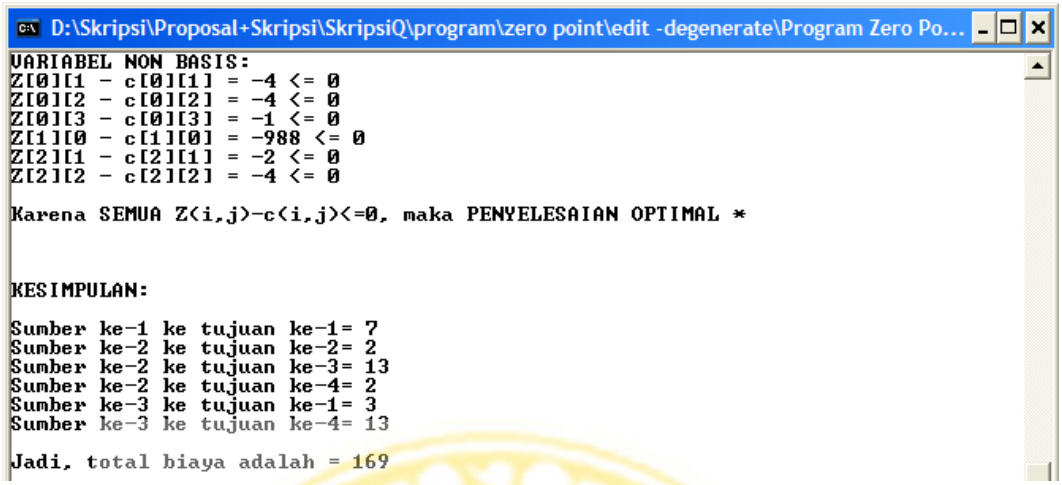

D:\Skripsi\Proposal+Skripsi\SkripsiQ\program\zero point\edit -degenerate\Program Zero Po...
LANGKAH 3: Pengecekan Kolom dan Baris
Cek Kolom
kolom ke-1: demand <= total suplay : SUDAH MEMENUHI (V)
kolom ke-2: demand <= total suplay : SUDAH MEMENUHI (V)
kolom ke-3: demand <= total suplay : SUDAH MEMENUHI (V)
kolom ke-4: demand <= total suplay : SUDAH MEMENUHI (V)
Cek Baris
baris ke-1: supply <= total demand : SUDAH MEMENUHI (V)
baris ke-2: supply <= total demand : SUDAH MEMENUHI (V)
baris ke-3: supply <= total demand : SUDAH MEMENUHI (V)
TABEL ALLOTMENT:
** ** 150 ** || 150
** ** 150 25 || 175
200 100 ** 75 || 375
==========
200 100 300 100
UJI OPTIMALITAS -> Modified Distribution (MODI):
** ** 10 ** || 0
** ** 11 0 || 1
4 5 ** 0 || 1
==========
3 4 10 -1
VARIABEL NON BASIS:
Z[0][0 - c[0][0] = -3 <= 0
Z[0][1 - c[0][1] = -4 <= 0
Z[0][3 - c[0][3] = -1 <= 0
Z[1][0 - c[1][0] = -3 <= 0
Z[1][1 - c[1][1] = -6 <= 0
Z[2][2 - c[2][2] = -1 <= 0
Karena SEMUA Z(i,j)-c(i,j)<=0, maka PENYELESAIAN OPTIMAL *
KESIMPULAN:
Sumber ke-1 ke tujuan ke-3= 150
Sumber ke-2 ke tujuan ke-3= 150
Sumber ke-2 ke tujuan ke-4= 25
Sumber ke-3 ke tujuan ke-1= 200
Sumber ke-3 ke tujuan ke-2= 100
Sumber ke-3 ke tujuan ke-4= 75
Jadi, total biaya adalah = 4450

Contoh 3


D:\Skripsi\Proposal+Skripsi\SkripsiQ\program\zero point\edit -degenerate\Program Zero Po...
Banyaknya suplay : 3
Banyaknya demand : 4
PILIHAN TUJUAN (Z) :
1. Memaksimumkan Z
2. Meminimumkan Z
Pilihan Tujuan = 2
LANGKAH 1: Membuat tabel transportasi
Tabel biaya (matrix C)
c[1][1]= 3
c[1][2]= 2
c[1][3]= 2
c[1][4]= 1
c[2][1]= 1000
c[2][2]= 7
c[2][3]= 7
c[2][4]= 9
c[3][1]= 4
c[3][2]= 1
c[3][3]= 3
c[3][4]= 1
Tabel supply (matrix S)
Supply1: 7
Supply2: 17
Supply3: 16
Tabel demand (matrix D)
Demand1: 10
Demand2: 2
Demand3: 13
Demand4: 15
TABEL TRANSPORTASI:
3 2 2 1 || 7
1000 7 7 9 || 17
4 1 3 1 || 16
==========
10 2 13 15
LANGKAH 2: Mereduksi tabel transportasi dengan
elemen terkecil di tiap baris dan kolom
TABEL TRANSPORTASI:
0 1 1 0 || 7
991 0 0 2 || 17
1 0 2 0 || 16
==========
10 2 13 15


D:\Skripsi\Proposal+Skripsi\SkripsiQ\program\zero point\edit -degenerate\Program Zero Po...
LANGKAH 3: Pengecekan Kolom dan Baris
Cek Kolom
kolom ke-1: demand > total suplay : tidak memenuhi
kolom ke-2: demand <= total suplay : SUDAH MEMENUHI (V)
kolom ke-3: demand <= total suplay : SUDAH MEMENUHI (V)
kolom ke-4: demand <= total suplay : SUDAH MEMENUHI (V)
Cek Baris
baris ke-1: supply <= total demand : SUDAH MEMENUHI (V)
baris ke-2: supply > total demand : tidak memenuhi
baris ke-3: supply <= total demand : SUDAH MEMENUHI (V)
LANGKAH 4: Memberi garis pada setiap elemen nol
M
2
LANGKAH 5: Merevisi tabel transportasi
TABEL TRANSPORTASI:
0 3 3 0 || 7
989 0 0 0 || 17
1 2 4 0 || 16
10 2 13 15
LANGKAH 3: Pengecekan Kolom dan Baris
Cek Kolom
kolom ke-1: demand > total suplay : tidak memenuhi
kolom ke-2: demand <= total suplay : SUDAH MEMENUHI (V)
kolom ke-3: demand <= total suplay : SUDAH MEMENUHI (V)
kolom ke-4: demand <= total suplay : SUDAH MEMENUHI (V)
Cek Baris
baris ke-1: supply <= total demand : SUDAH MEMENUHI (V)
baris ke-2: supply <= total demand : SUDAH MEMENUHI (V)
baris ke-3: supply > total demand : tidak memenuhi
LANGKAH 4: Memberi garis pada setiap elemen nol
1 2 4
LANGKAH 5: Merevisi tabel transportasi
TABEL TRANSPORTASI:
0 3 3 1 || 7
989 0 0 1 || 17
0 1 3 0 || 16
10 2 13 15


D:\Skripsi\Proposal+Skripsi\SkripsiQ\program\zero point\edit -degenerate\Program Zero Po...
LANGKAH 3: Pengecekan Kolom dan Baris
Cek Kolom
kolom ke-1: demand <= total suplay : SUDAH MEMENUHI (V)
kolom ke-2: demand <= total suplay : SUDAH MEMENUHI (V)
kolom ke-3: demand <= total suplay : SUDAH MEMENUHI (V)
kolom ke-4: demand <= total suplay : SUDAH MEMENUHI (V)
Cek Baris
baris ke-1: supply <= total demand : SUDAH MEMENUHI (V)
baris ke-2: supply > total demand : tidak memenuhi
baris ke-3: supply <= total demand : SUDAH MEMENUHI (V)
LANGKAH 4: Memberi garis pada setiap elemen nol
LANGKAH 5: Merevisi tabel transportasi
TABEL TRANSPORTASI:
LANGKAH 3: Pengecekan Kolom dan Baris
Cek Kolom
kolom ke-1: demand <= total suplay : SUDAH MEMENUHI (V)
kolom ke-2: demand <= total suplay : SUDAH MEMENUHI (V)
kolom ke-3: demand <= total suplay : SUDAH MEMENUHI (V)
kolom ke-4: demand <= total suplay : SUDAH MEMENUHI (V)
Cek Baris
baris ke-1: supply <= total demand : SUDAH MEMENUHI (V)
baris ke-2: supply <= total demand : SUDAH MEMENUHI (V)
baris ke-3: supply <= total demand : SUDAH MEMENUHI (V)
TABEL ALLOTMENT:
UJI OPTIMALITAS -> Modified Distribution (MODI):


D:\Skripsi\Proposal+Skripsi\SkripsiQ\program\zero point\edit -degenerate\Program Zero Po...
VARIABEL NON BASIS:
Z[0][1 - c[0][1] = -4 <= 0
Z[0][2 - c[0][2] = -4 <= 0
Z[0][3 - c[0][3] = -1 <= 0
Z[1][0 - c[1][0] = -988 <= 0
Z[2][1 - c[2][1] = -2 <= 0
Z[2][2 - c[2][2] = -4 <= 0
Karena SEMUA Z(i,j)-c(i,j)<=0, maka PENYELESAIAN OPTIMAL *
KESIMPULAN:
Sumber ke-1 ke tujuan ke-1= 7
Sumber ke-2 ke tujuan ke-2= 2
Sumber ke-2 ke tujuan ke-3= 13
Sumber ke-2 ke tujuan ke-4= 2
Sumber ke-3 ke tujuan ke-1= 3
Sumber ke-3 ke tujuan ke-4= 13
Jadi, total biaya adalah = 169