```
/*SOFTWARE PELATIHAN JARINGAN*/
#include <conio.h>
#include <stdlib.h>
#include <math.h>
#include <string.h>
#include <fstream.h>

const char Dl[5]={0,1,3,2,1};
const float do_bar[5]={0,4.0,4.0,2.5,1.0};
const float do_doang[5]={0,0.9,0.8,0.7,1.0};
const char dimensi_US[5]={0,9,10,12,4};
const char plus_US[5]={0,1,1,2,4};
const char KSl[5]={0,12,80,97,47};
float gamma[5]={0,0.9,0.9,0.9,0.8};
float r4=1.0;
const char Al[5]={0,1,2,2,2};
const char KCl[5]={1,8,33,64,35};
const char dimensi_UC[5]={9,10,12,12,0};
const char plus_UC[5]={1,1,2,4,1};
char jL[4][64];
char U0[19][19];
const char n1[12]={1,1,1,1,1,1,1,1,1,1,1,1};
const char n2[80]={4,4,4,4,4,2,4,3,4,2,
                   3,4,4,2,2,4,3,4,3,3,
                   4,3,3,4,4,3,3,3,3,3,
                   4,3,2,5,4,3,3,3,3,3,
                   3,4,3,3,3,3,3,3,2,3,
                   3,4,3,3,3,3,3,3,3,3,
                   3,3,4,3,3,3,3,3,3,2,
                   3,2,3,2,4,4,2,2,2,2};
const char n3[97]={2,3,2,2,2,2,2,2,2,3,2,2,2,1,1,2,3,1,2,2,4,4,2,2,2,
                   2,2,2,2,2,3,2,2,2,2,2,3,2,1,2,3,2,1,3,2,2,2,3,2,2,
                   2,2,3,3,2,2,2,2,2,2,2,2,2,2,2,2,2,3,3,3,3,3,2,1,2,2,
                   2,2,2,2,2,1,2,2,2,2,2,2,2,1,2,2,2,2,2,2,2,2,2,2,2};
const char n4[47]={2,2,2,2,2,3,4,1,1,2,2,2,2,2,2,1,2,2,1,2,2,1,2,2,
                   1,2,2,2,2,2,2,2,1,2,2,1,2,2,2,2,2,2,1,2,2,2,2};
const char seed_x_US3[97]={5,5,5,5,5,6,4,4,4,4,4,6,5,5,5,4,6,6,6,5,6,6,4,4,4,
                           4,4,6,4,4,4,4,4,4,6,6,4,6,6,4,4,6,6,6,5,5,6,6,5,5,4,
                           4,6,6,6,5,4,4,6,5,5,5,5,5,5,6,5,6,6,6,6,6,6,6,4,5,
                           4,5,5,5,5,5,6,4,4,6,5,5,5,5,4,4,4,6,6,5,4,5};
const char seed_y_US3[97]={4,4,6,4,6,5,5,4,4,6,6,6,5,5,5,5,5,5,5,4,6,6,4,4,5,
                           5,5,4,6,6,5,5,5,4,4,4,4,4,5,5,5,4,4,4,4,5,5,6,6,5,
                           6,6,5,5,6,4,6,4,4,6,4,5,5,6,5,5,6,6,6,6,6,6,5,6,
                           6,5,5,5,5,5,5,4,4,5,5,5,5,6,6,4,6,6,6,6,6,4,5};
float US1[12][19][19];
float UC1[8][21][21];
float US2[80][21][21];
float UC2[33][13][13];
float US3[97][13][13];
float UC3[64][7][7];
float US4[47][3][3];
float UC4[35][1][1];
float UV1[19][19];
float UV2[21][21];
```

```
float UV3[13][13];
float UV4[3][3];
char tot[19][19];
float PC[9][8];
float p_weight[35][9][8];
float a1[12][3][3];
float a2[80][8][5][5];
float a3[97][33][5][5];
float a4[47][64][5][5];
float b1[12],b2[80],b3[97],b4[47];

typedef struct{
    char x;
    char y;
} koord;

typedef struct{
    char banyak;
    char x;
    char y;
} tetangga;

void inisialisasi_a_weight()
{
    ifstream input;
    char file[10];
    char index[3];
    char i,j,k,l;

    for(i=0;i<KSl[1];i++) {
        strcpy(file,"A");
        itoa(1,index,10);
        strcat(file,index);
        strcat(file,"-");
        itoa(i+1,index,10);
        strcat(file,index);
        strcat(file,".wgt");
        input.open(file,ios::binary);
        for(l=0;l<3;l++) {
            for(k=0;k<3;k++) {
                input.read((char *) &a1[i][k][l],sizeof(float));
            }
        }
        input.close();
    }

    for(i=0;i<80;i++) {
        strcpy(file,"A");
        itoa(2,index,10);
        strcat(file,index);
        strcat(file,"-");
        itoa(i+1,index,10);
        strcat(file,index);
        strcat(file,".wgt");
```

```
    input.open(file,ios::binary);
    for(j=0;j<8;j++) {
        for(l=0;l<5;l++) {
            for(k=0;k<5;k++) {
                input.read((char *) &a2[i][j][k][l],sizeof(float));
            }
        }
    }
    input.close();
}

    for(i=0;i<KSl[3];i++) {
    strcpy(file,"A");
    itoa(3,index,10);
    strcat(file,index);
    strcat(file,"-");
    itoa(i+1,index,10);
    strcat(file,index);
    strcat(file,".wgt");
    input.open(file,ios::binary);
    for(j=0;j<33;j++) {
        for(l=0;l<5;l++) {
            for(k=0;k<5;k++) {
                input.read((char *) &a3[i][j][k][l],sizeof(float));
            }
        }
    }
    input.close();

}

for(i=0;i<KSl[4];i++) {
    strcpy(file,"A");
    itoa(4,index,10);
    strcat(file,index);
    strcat(file,"-");
    itoa(i+1,index,10);
    strcat(file,index);
    strcat(file,".wgt");
    input.open(file,ios::binary);
    for(j=0;j<64;j++) {
        for(l=0;l<5;l++) {
            for(k=0;k<5;k++) {
                input.read((char *) &a4[i][j][k][l],sizeof(float));
            }
        }
    }
    input.close();
}
for(i=0;i<35;i++) {
    strcpy(file,"P");
    itoa(i,index,10);
    strcat(file,index);
    strcat(file,".wgt");
```

```
        input.open(file);
        input.read((char *) p_weight[i],sizeof(p_weight[i]));
        input.close();
    }
}

void inisialisasi_b_weight()
{
    ifstream input;

    input.open("b1.wgt",ios::binary);
    input.read((char *) b1,sizeof(b1));
    input.close();
    input.open("b2.wgt",ios::binary);
    input.read((char *) b2,sizeof(b2));
    input.close();
    input.open("b3.wgt",ios::binary);
    input.read((char *) b3,sizeof(b3));
    input.close();
    input.open("b4.wgt",ios::binary);
    input.read((char *) b4,sizeof(b4));
    input.close();
}

void inisialisasi_jL()
{
    char idx1;
    char tmp1[7][21]= {{0,2,4,6},
                       {1,3,5,7},
                       {4,8,9,14,26,27,30},
                       {6,10,18,19},
                       {0,6,7,13,14,17,19,21,22,24,25,26,27,31,39,44,45,46,55,59,60},
                       {9,11,15,18},
                       {1,4,5,6,9,12,16,19,20,26}};

    for(idx1=0;idx1<64;idx1++) jL[0][idx1]=0;
    for(idx1=0;idx1<4;idx1++) jL[0][tmp1[0][idx1]]=1;
    for(idx1=0;idx1<4;idx1++) jL[0][tmp1[1][idx1]]=2;
    for(idx1=0;idx1<64;idx1++) jL[1][idx1]=0;
    for(idx1=0;idx1<33;idx1++) jL[1][idx1]=1;
    for(idx1=0;idx1<7;idx1++) jL[1][tmp1[2][idx1]]=2;
    jL[1][7]=jL[1][24]=jL[1][25]=3;
    for(idx1=0;idx1<4;idx1++) jL[1][tmp1[3][idx1]]=4;
    jL[1][5]=jL[1][11]=5;
    jL[1][16]=jL[1][17]=8;
    for(idx1=0;idx1<64;idx1++) jL[2][idx1]=1;
    for(idx1=0;idx1<21;idx1++) jL[2][tmp1[4][idx1]]=2;
    for(idx1=0;idx1<4;idx1++) jL[2][tmp1[5][idx1]]=3;
    jL[2][50]=5;
    for(idx1=0;idx1<64;idx1++) jL[3][idx1]=0;
    for(idx1=0;idx1<35;idx1++) jL[3][idx1]=1;
    for(idx1=0;idx1<10;idx1++) jL[3][tmp1[6][idx1]]=2;
    jL[3][25]=3;
```

```
}

char hitung_jL(char layer,char kS,char kC)
{
    char tmp2=0;
    char tmp1,idx1;

    for(idx1=0;idx1<=kC;idx1++) tmp2+=jL[layer-1][idx1];
    if( (kS+1>(tmp2-jL[layer-1][kC])) && (kS+1<=tmp2) )tmp1=1;
    else tmp1=0;

    return tmp1;
}

koord seed_cell(char layer,char k_seed)
{
    koord tmp1;

    if (layer==1) tmp1.x=tmp1.y=0;
    if (layer==2) tmp1.x=tmp1.y=0;
    if (layer==3) {
        tmp1.x=(seed_x_US3[k_seed]-5)*2;
        tmp1.y=(seed_y_US3[k_seed]-5)*2;
    }
    if (layer==4) tmp1.x=tmp1.y=0;

    return tmp1;
}

float a_weight(char layer,koord v,char kC,char kS)
{
    float tmp1;
    koord tmp2;

    tmp2.x=v.x+Al[layer];
    tmp2.y=v.y+Al[layer];

    if(layer==1) tmp1=a1[kS][tmp2.x][tmp2.y];
    if(layer==2) tmp1=a2[kS][kC][tmp2.x][tmp2.y];
    if(layer==3) tmp1=a3[kS][kC][tmp2.x][tmp2.y];
    if(layer==4) tmp1=a4[kS][kC][tmp2.x][tmp2.y];

    return tmp1;
}

void isi_U0(char layer,char k,char n)
{
    char file[10],index[3];
    ifstream input;
    char tengah[5]={0,8,5,0,0};
    char lebar[5]={19,3,9,19,19};

    itoa(layer,file,10);
    strcat(file,"-");
```

```
    itoa(k+1,index,10);
    strcat(file,index);
    strcat(file,".ptt");
    input.open(file);
    input.seekg(n*sizeof(U0),ios::beg);
    input.read((char *) U0,sizeof(U0));
    input.close();
}


float b_weight(char layer,char k)
{
    float tmp1;

    if(layer==1) tmp1=b1[k];
    if(layer==2) tmp1=b2[k];
    if(layer==3) tmp1=b3[k];
    if(layer==4) tmp1=b4[k];

    return tmp1;
}

char hitung_U0(koord n)
{
    return U0[n.x+dimensi_UC[0]][n.y+dimensi_UC[0]];
}


float rL(char layer,char k)
{
    float tmp1;

    if(layer==1) tmp1=1.7;
    if(layer==2) {
        if(k==39||k==41||k==40||k==47||k==48||k==46) tmp1=3.8;
        else tmp1=4.0;
    }
    if(layer==3) tmp1=1.5;
    if(layer==4) tmp1=r4;

    return tmp1;
}

float UC(char layer,koord n,char k)
{
    float tmp1;

    if(layer==0) tmp1=hitung_U0(n);
    if(layer==1) tmp1=UC1[k][n.x+dimensi_UC[layer]][n.y+dimensi_UC[layer]];
    if(layer==2) tmp1=UC2[k][(n.x+dimensi_UC[layer])/plus_UC[layer]]
                     [(n.y+dimensi_UC[layer])/plus_UC[layer]];
    if(layer==3) tmp1=UC3[k][(n.x+dimensi_UC[layer])/plus_UC[layer]]
                     [(n.y+dimensi_UC[layer])/plus_UC[layer]];
    if(layer==4) tmp1=UC4[k][n.x][n.y];
```

```
    return tmp1;
}

float UV(char layer,koord n)
{
    float tmp1;

    if(layer==1) tmp1=UV1[n.x+dimensi_US[layer]][n.y+dimensi_US[layer]];
    if(layer==2) tmp1=UV2[n.x+dimensi_US[layer]][n.y+dimensi_US[layer]];
    if(layer==3) tmp1=UV3[(n.x+dimensi_US[layer])/plus_US[layer]]
                    [(n.y+dimensi_US[layer])/plus_US[layer]];
    if(layer==4) tmp1=UV4[(n.x+dimensi_US[layer])/plus_US[layer]]
                    [(n.y+dimensi_US[layer])/plus_US[layer]];

    return tmp1;
}

float US(char layer,koord n,char k)
{
    float tmp1;

    if(layer==1) tmp1=US1[k][n.x+dimensi_US[layer]][n.y+dimensi_US[layer]];
    if(layer==2) tmp1=US2[k][n.x+dimensi_US[layer]][n.y+dimensi_US[layer]];
    if(layer==3) tmp1=US3[k][(n.x+dimensi_US[layer])/plus_US[layer]]
                    [(n.y+dimensi_US[layer])/plus_US[layer]];
    if(layer==4) tmp1=US4[k][(n.x+dimensi_US[layer])/plus_US[layer]]
                    [(n.y+dimensi_US[layer])/plus_US[layer]];

    return tmp1;
}

float nilai_UC(char layer,koord n,char k);

float nilai_UV(char layer,koord n)
{
    char idx1;
    float tmp1,pangkat,c_weight;
    koord tmp2,tmp3;

    tmp1=0.0;
    for(idx1=0;idx1<KCl[layer-1];idx1++) {
        for(tmp2.x=(0-Al[layer]);tmp2.x<=Al[layer];(tmp2.x)++) {
            for(tmp2.y=(0-Al[layer]);tmp2.y<=Al[layer];tmp2.y++) {
                pangkat=sqrt((tmp2.x*tmp2.x)+(tmp2.y*tmp2.y));
                c_weight=pow(gamma[layer],pangkat);
                tmp3.x=n.x+(tmp2.x*plus_UC[layer-1]);
                tmp3.y=n.y+(tmp2.y*plus_UC[layer-1]);
                if((abs(tmp3.x)>dimensi_UC[layer-1])||(abs(tmp3.y)>dimensi_UC[layer-1])) tmp1+=0;
                else tmp1+=c_weight*UC(layer-1,tmp3,idx1)*UC(layer-1,tmp3,idx1);
            }
        }
    }
```

```
    tmp1=sqrt(tmp1);
    return tmp1;
}

float nilai_US(char layer,koord n,char k)
{
    float tmp1,pembilang=0.0,penyebut;
    char idx1;
    koord tmp2,tmp3;

    for(idx1=0;idx1<KCl[layer-1];idx1++) {
        for(tmp2.x=(0-Al[layer]);tmp2.x<=Al[layer];(tmp2.x)++) {
            for(tmp2.y=(0-Al[layer]);tmp2.y<=Al[layer];tmp2.y++) {
                tmp3.x=n.x+(tmp2.x*plus_UC[layer-1]);
                tmp3.y=n.y+(tmp2.y*plus_UC[layer-1]);
                if((abs(tmp3.x)>dimensi_UC[layer-1])||(abs(tmp3.y)>dimensi_UC[layer-1]))
                    pembilang+=0;
                else pembilang+=a_weight(layer,tmp2,idx1,k)*UC(layer-1,tmp3,idx1);
            }
        }
    }
    pembilang+=1;
    penyebut=1+( rL(layer,k) * (1/(1+rL(layer,k))) *b_weight(layer,k) *UV(layer,n) );
    tmp1=(pembilang/penyebut)-1;
    if(tmp1<0) tmp1=0;
    tmp1*=rL(layer,k);

    return tmp1;
}

float nilai_UC(char layer,koord n, char k)
{
    float tmp1,tmp3,d_weight,pangkat;
    char tmp2;
    koord tmp4,tmp5;
    char idx1;

    if(layer==0) tmp1=hitung_U0(n);
    else {
        tmp1=0.0;
        for(idx1=0;idx1<KSl[layer];idx1++) {
            tmp2=hitung_jL(layer,idx1,k);
            tmp3=0.0;
            for(tmp4.x=(0-Dl[layer]);tmp4.x<=Dl[layer];(tmp4.x)++) {
                for(tmp4.y=(0-Dl[layer]);tmp4.y<=Dl[layer];tmp4.y++) {
                    pangkat=sqrt((tmp4.x*tmp4.x)+(tmp4.y*tmp4.y));
                    d_weight=do_bar[layer]*pow(do_doang[layer],pangkat);
                    tmp5.x=n.x+(tmp4.x*plus_US[layer]);
                    tmp5.y=n.y+(tmp4.y*plus_US[layer]);
                    if((abs(tmp5.x)>dimensi_US[layer])||(abs(tmp5.y)>dimensi_US[layer])) tmp3+=0;
                    else tmp3+=d_weight*US(layer,tmp5,idx1);
                }
            }
            tmp1+=tmp2*tmp3;
```

```
        }
        tmp1=tmp1/(1.0+tmp1);
    }
    return tmp1;

}

void hitung_UV(char layer)
{
    koord n;
    if(layer==1) {
        for(n.x=(0-dimensi_US[layer]);n.x<=dimensi_US[layer];n.x++) {
            for(n.y=(0-dimensi_US[layer]);n.y<=dimensi_US[layer];n.y++) {
                UV1[n.x+dimensi_US[layer]][n.y+dimensi_US[layer]]=nilai_UV(layer,n);
            }
        }
    }
    if(layer==2) {
        for(n.x=(0-dimensi_US[layer]);n.x<=dimensi_US[layer];n.x++) {
            for(n.y=(0-dimensi_US[layer]);n.y<=dimensi_US[layer];n.y++) {
                UV2[n.x+dimensi_US[layer]][n.y+dimensi_US[layer]]=nilai_UV(layer,n);
            }
        }
    }
    if(layer==3) {
        for(n.x=(0-dimensi_US[layer]);n.x<=dimensi_US[layer];n.x+=plus_US[layer]) {
            for(n.y=(0-dimensi_US[layer]);n.y<=dimensi_US[layer];n.y+=plus_US[layer]) {
                UV3[(n.x+dimensi_US[layer])/plus_US[layer]]
                    [(n.y+dimensi_US[layer])/plus_US[layer]]=nilai_UV(layer,n);
            }
        }
    }
    if(layer==4) {
        for(n.x=(0-dimensi_US[layer]);n.x<=dimensi_US[layer];n.x+=plus_US[layer]) {
            for(n.y=(0-dimensi_US[layer]);n.y<=dimensi_US[layer];n.y+=plus_US[layer]) {
                UV4[(n.x+dimensi_US[layer])/plus_US[layer]]
                    [(n.y+dimensi_US[layer])/plus_US[layer]]=nilai_UV(layer,n);
            }
        }
    }
}

void hitung_US(char layer)
{
    koord n;
    char k;

    if(layer==1) {
        for(k=0;k<12;k++) {
            for(n.x=(0-dimensi_US[layer]);n.x<=dimensi_US[layer];n.x++) {
                for(n.y=(0-dimensi_US[layer]);n.y<=dimensi_US[layer];n.y++) {
                    US1[k][(n.x+dimensi_US[layer])][(n.y+dimensi_US[layer])]=nilai_US(layer,n,k);
                }
            }
```

```
        }
    }
    if(layer==2) {
        for(k=0;k<80;k++) {
            for(n.x=(0-dimensi_US[layer]);n.x<=dimensi_US[layer];n.x++) {
                for(n.y=(0-dimensi_US[layer]);n.y<=dimensi_US[layer];n.y++) {
                    US2[k][n.x+dimensi_US[layer]][n.y+dimensi_US[layer]]=nilai_US(layer,n,k);
                }
            }
        }
    }
    if(layer==3) {
        for(k=0;k<97;k++) {
            for(n.x=(0-dimensi_US[layer]);n.x<=dimensi_US[layer];n.x+=plus_US[layer]) {
                for(n.y=(0-dimensi_US[layer]);n.y<=dimensi_US[layer];n.y+=plus_US[layer]) {
                    US3[k][(n.x+dimensi_US[layer])/plus_US[layer]]
                        [(n.y+dimensi_US[layer])/plus_US[layer]]=nilai_US(layer,n,k);
                }
            }
        }
    }
    if(layer==4) {
        for(k=0;k<47;k++) {
            for(n.x=(0-dimensi_US[layer]);n.x<=dimensi_US[layer];n.x+=plus_US[layer]) {
                for(n.y=(0-dimensi_US[layer]);n.y<=dimensi_US[layer];n.y+=plus_US[layer]) {
                    US4[k][(n.x+dimensi_US[layer])/plus_US[layer]]
                        [(n.y+dimensi_US[layer])/plus_US[layer]]=nilai_US(layer,n,k);
                }
            }
        }
    }

}

void hitung_UC(char layer)
{
    koord n;
    char k;
    float a,b;


    if(layer==1) {
        for(k=0;k<KCl[layer];k++) {
            for(n.x=(0-dimensi_UC[layer]);n.x<=dimensi_UC[layer];n.x++) {
                for(n.y=(0-dimensi_UC[layer]);n.y<=dimensi_UC[layer];n.y++) {
                    UC1[k][(n.x+dimensi_UC[layer])/plus_UC[layer]]
                        [(n.y+dimensi_UC[layer])/plus_UC[layer]]=nilai_UC(layer,n,k);
                }
            }
        }
    }

    if(layer==2) {
        for(k=0;k<KCl[layer];k++) {
```

```
            for(n.x=(0-dimensi_UC[layer]);n.x<=dimensi_UC[layer];n.x+=plus_UC[layer]) {
                for(n.y=(0-dimensi_UC[layer]);n.y<=dimensi_UC[layer];n.y+=plus_UC[layer]) {
                    UC2[k][(n.x+dimensi_UC[layer])/plus_UC[layer]]
                        [(n.y+dimensi_UC[layer])/plus_UC[layer]]=nilai_UC(layer,n,k);
                }
            }
        }
    }
    if(layer==3) {
        for(k=0;k<KCl[layer];k++) {
            for(n.x=(0-dimensi_UC[layer]);n.x<=dimensi_UC[layer];n.x+=plus_UC[layer]) {
                for(n.y=(0-dimensi_UC[layer]);n.y<=dimensi_UC[layer];n.y+=plus_UC[layer]) {
                    UC3[k][(n.x+dimensi_UC[layer])/plus_UC[layer]]
                        [(n.y+dimensi_UC[layer])/plus_UC[layer]]=nilai_UC(layer,n,k);
                }
            }
        }
    }
    if(layer==4) {
        for(k=0;k<KCl[layer];k++) {
            n.x=0;
            n.y=0;
            UC4[k][0][0]=nilai_UC(layer,n,k);
        }
    }
}

float delta_a_weight(char layer,koord v,char k,char k_seed)
{
    float tmp1;
    koord tmp2;
    float gamma[5]={0,0.9,0.9,0.9,0.8};
    float pangkat,c_weight;

    tmp2=seed_cell(layer,k_seed);
    pangkat=sqrt((v.x*v.x)+(v.y*v.y));
    c_weight=pow(gamma[layer],pangkat);
    v.x*=plus_UC[layer-1];
    v.y*=plus_UC[layer-1];
    tmp2.x+=v.x;
    tmp2.y+=v.y;
    tmp1=10000*c_weight*UC(layer-1,tmp2,k);

    return tmp1;
}

float delta_b_weight(char layer,char k_seed)
{
    float tmp1;
    koord tmp2;

    tmp2=seed_cell(layer,k_seed);
    tmp1=10000*UV(layer,tmp2);
```

```
    return tmp1;
}

void main()
{
    char i,j,k,l;
//  float a[3][3],b[12];  // untuk pelatihan lapisan pertama
//  float a[8][5][5],b[80]; //untuk pelatihan lapisan kedua
//  float a[33][5][5],b[97]; //untuk pelatihan lapisan ketiga
//  float a[64][5][5],b[47];
    ofstream outb,outa;
    char file[10];
    char index[3];
    koord n;
/*
//  pelatihan pertama mulai
    strcpy(file,"B");
    itoa(1,index,10);
    strcat(file,index);
    strcat(file,".wgt");
    outb.open(file,ios::binary);
    for(k=0;k<12;k++) {
        strcpy(file,"A");
        itoa(1,index,10);
        strcat(file,index);
        strcat(file,"-");
        itoa(k+1,index,10);
        strcat(file,index);
        strcat(file,".wgt");
        outa.open(file,ios::binary);
        isi_U0(1,k,0);
        hitung_UV(1);
        b[k]=0.0;
        for(n.y=-1;n.y<=1;n.y++) {
            for(n.x=-1;n.x<=1;n.x++) {
                a[n.x+1][n.y+1]=delta_a_weight(1,n,0,k);
                outa.write((char *) &a[n.x+1][n.y+1],sizeof(float));
            }
        }
        b[k]=delta_b_weight(1,k);
        outa.close();
    }
    outb.write((char *) b,sizeof(b));
    outb.close();
//  pelatihan pertama selesai */

/*  //pelatihan kedua mulai
    strcpy(file,"B");
    itoa(2,index,10);
    strcat(file,index);
    strcat(file,".wgt");
    outb.open(file,ios::binary);
    strcpy(file,"A");
    itoa(2,index,10);
```

```
        strcat(file,index);
        strcat(file,"-");
        itoa(k+1,index,10);
        strcat(file,index);
        strcat(file,".wgt");
        outa.open(file,ios::binary);
        b[k]=0.0;
        for(l=0;l<8;l++) {
            for(n.y=-2;n.y<=2;n.y++) {
                for(n.x=-2;n.x<=2;n.x++) {
                    a[l][n.x+2][n.y+2]=0.0;
                }
            }
        }
        for(i=0;i<n2[k];i++) {
            isi_U0(2,k,i);
            hitung_UV(1);
            hitung_US(1);
            hitung_UC(1);
            hitung_UV(2);
            for(l=0;l<8;l++) {
                for(n.y=-2;n.y<=2;n.y++) {
                    for(n.x=-2;n.x<=2;n.x++) {
                        a[l][n.x+2][n.y+2]+=delta_a_weight(2,n,l,k);
                    }
                }
            }
            b[k]+=delta_b_weight(2,k);
        }
        for(l=0;l<8;l++) {
            for(n.y=-2;n.y<=2;n.y++) {
                for(n.x=-2;n.x<=2;n.x++) {
                    outa.write((char *) &a[l][n.x+2][n.y+2],sizeof(float));
                    }
                }

            }
            outa.close();
        }
        outb.write((char *) b,sizeof(b));
        outb.close();
    //pelatihan kedua selesai */

    /*  //pelatihan ketiga mulai
        strcpy(file,"B");
        itoa(3,index,10);
        strcat(file,index);
        strcat(file,".wgt");
        outb.open(file,ios::binary);
        for(k=0;k<97;k++) {
            strcpy(file,"A");
            itoa(3,index,10);
            strcat(file,index);
            strcat(file,"-");
```

```
            itoa(k+1,index,10);
            strcat(file,index);
            strcat(file,".wgt");
            outa.open(file,ios::binary);
            for(i=0;i<33;i++) {
                for(j=0;j<5;j++) {
                    for(l=0;l<5;l++) a[i][j][l]=0.0;
                }
            }
            b[k]=0.0;
            for(m=0;m<n3[k];m++) {
                isi_U0(3,k,m);
                hitung_UV(1);
                hitung_US(1);
                hitung_UC(1);
                hitung_UV(2);
                hitung_US(2);
                hitung_UC(2);
                hitung_UV(3);
                for(l=0;l<33;l++) {
                    for(n.y=-2;n.y<=2;n.y++) {
                        for(n.x=-2;n.x<=2;n.x++) {
                            a[l][n.x+2][n.y+2]+=delta_a_weight(3,n,l,k);
                        }
                    }
                }
                b[k]+=delta_b_weight(3,k);
            }
            for(l=0;l<33;l++) {
                for(n.y=-2;n.y<=2;n.y++) {
                    for(n.x=-2;n.x<=2;n.x++) {
                        outa.write((char *) &a[l][n.x+2][n.y+2],sizeof(float));
                    }
                }

            }
            outa.close();
        }
        outb.write((char *) b,sizeof(b));
        outb.close();
    //pelatihan ketiga selesai */

    /*  //pelatihan lapisan keempat mulai
        strcpy(file,"B");
        itoa(4,index,10);
        strcat(file,index);
        strcat(file,".wgt");
        outb.open(file,ios::binary);
        for(k=0;k<47;k++) {
            strcpy(file,"A");
            itoa(4,index,10);
            strcat(file,index);
            strcat(file,"-");
            itoa(k+1,index,10);
```

```
        strcat(file,index);
        strcat(file,".wgt");
        outa.open(file,ios::binary);
        for(i=0;i<64;i++) {
            for(j=0;j<5;j++) {
                for(l=0;l<5;l++) a[i][j][l]=0.0;
            }
        }
        b[k]=0.0;
        for(m=0;m<n4[k];m++) {
            isi_U0(4,k,m);
            hitung_UV(1);
            hitung_US(1);
            hitung_UC(1);
            hitung_UV(2);
            hitung_US(2);
            hitung_UC(2);
            hitung_UV(3);
            hitung_US(3);
            hitung_UC(3);
            hitung_UV(4);
            for(l=0;l<64;l++) {
                for(n.y=-2;n.y<=2;n.y++) {
                    for(n.x=-2;n.x<=2;n.x++) {
                        a[l][n.x+2][n.y+2]+=delta_a_weight(4,n,l,k);
                    }
                }
            }
            b[k]+=delta_b_weight(4,k);
        }
        for(l=0;l<64;l++) {
            for(n.y=-2;n.y<=2;n.y++) {
                for(n.x=-2;n.x<=2;n.x++) {
                    outa.write((char *) &a[l][n.x+2][n.y+2],sizeof(float));
                }
            }
        }
        outa.close();
    }
    outb.write((char *) b,sizeof(b));
    outb.close();
//pelatihan keempat selesai*/
}

-----------------------------------------------------------------------------------------------------------------------
-
/*SOFTWARE PENGENALAN*/
-----------------------------------------------------------------------------------------------------------------------
-

// ta1.h : main header file for the TA1 application

#if !defined(AFX_TA1_H__F4ACEA81_93F0_49D6_8B82_EE98EEBC4256__INCLUDED_)
#define AFX_TA1_H__F4ACEA81_93F0_49D6_8B82_EE98EEBC4256__INCLUDED_
```

```
#if _MSC_VER > 1000
#pragma once
#endif // _MSC_VER > 1000

#ifndef __AFXWIN_H__
    #error include 'stdafx.h' before including this file for PCH
#endif

#include "resource.h"          // main symbols

class CTa1App : public CWinApp
{
public:
    CTa1App();

// Overrides
    // ClassWizard generated virtual function overrides
    //{{AFX_VIRTUAL(CTa1App)
    public:
    virtual BOOL InitInstance();
    //}}AFX_VIRTUAL

// Implementation

    //{{AFX_MSG(CTa1App)
        // NOTE - the ClassWizard will add and remove member functions here.
        //    DO NOT EDIT what you see in these blocks of generated code !
    //}}AFX_MSG
    DECLARE_MESSAGE_MAP()
};


/////////////////////////////////////////////////////////////////////////////

//{{AFX_INSERT_LOCATION}}
// Microsoft Visual C++ will insert additional declarations immediately before the previous line.

#endif // !defined(AFX_TA1_H__F4ACEA81_93F0_49D6_8B82_EE98EEBC4256__INCLUDED_)

-----------------------------------------------------------------------------------------------------------------------
-
// ta1.cpp : Defines the class behaviors for the application.
//

#include "stdafx.h"
#include "ta1.h"
#include "ta1Dlg.h"

#ifdef _DEBUG
#define new DEBUG_NEW
#undef THIS_FILE
static char THIS_FILE[] = __FILE__;
#endif
```

```
/////////////////////////////////////////////////////////////////////////////
// CTa1App

BEGIN_MESSAGE_MAP(CTa1App, CWinApp)
    //{{AFX_MSG_MAP(CTa1App)
        // NOTE - the ClassWizard will add and remove mapping macros here.
        //    DO NOT EDIT what you see in these blocks of generated code!
    //}}AFX_MSG
    ON_COMMAND(ID_HELP, CWinApp::OnHelp)
END_MESSAGE_MAP()

/////////////////////////////////////////////////////////////////////////////
// CTa1App construction

CTa1App::CTa1App()
{
    // TODO: add construction code here,
    // Place all significant initialization in InitInstance
}

/////////////////////////////////////////////////////////////////////////////
// The one and only CTa1App object

CTa1App theApp;

/////////////////////////////////////////////////////////////////////////////
// CTa1App initialization

BOOL CTa1App::InitInstance()
{
    AfxEnableControlContainer();

    // Standard initialization
    // If you are not using these features and wish to reduce the size
    //  of your final executable, you should remove from the following
    //  the specific initialization routines you do not need.

#ifdef _AFXDLL
    Enable3dControls();            // Call this when using MFC in a shared DLL
#else
    Enable3dControlsStatic();    // Call this when linking to MFC statically
#endif

    CTa1Dlg dlg;
    m_pMainWnd = &dlg;
    int nResponse = dlg.DoModal();
    if (nResponse == IDOK)
    {
    }
    else if (nResponse == IDCANCEL)
    {
    }
```

```
    return FALSE;
}
-------------------------------------------------------------------------------------------------------------------------
-

// ta1Dlg.h : header file
//

#if !defined(AFX_TA1DLG_H__036827B0_F772_420D_9418_B8B4C89A8A5E__INCLUDED_)
#define AFX_TA1DLG_H__036827B0_F772_420D_9418_B8B4C89A8A5E__INCLUDED_

#if _MSC_VER > 1000
#pragma once
#endif // _MSC_VER > 1000
#include "CanvasDlg.h"
#include "StatDlg.h"

/////////////////////////////////////////////////////////////////////////////
// CTa1Dlg dialog

typedef struct {
        char x;
        char y;
} koord;

typedef struct{
        char banyak;
        char x;
        char y;
} tetangga;

class CTa1Dlg : public CDialog
{
// Construction
private:
    void hapus_noise();

    void inisialisasi_a_weight();
    void inisialisasi_b_weight();
    void inisialisasi_jL();
    char hitung_jL(char layer,char kS,char kC);
    koord seed_cell(char layer,char k_seed);
    float a_weight(char layer,koord v,char kC,char kS);
    float b_weight(char layer,char k);
    char hitung_U0(koord n);
    float rL(char layer,char k);
    float UC(char layer,koord n,char k);
    float UV(char layer,koord n);
    float US(char layer,koord n,char k);
    float nilai_UC(char layer,koord n,char k);
    float nilai_UV(char layer,koord n);
    float nilai_US(char layer,koord n,char k);
    float nilai_PC(char kuadran,char k);
    void hitung_PC();
```

```
    void hitung_UV(char layer);
    void hitung_US(char layer);
    void hitung_UC(char layer);
    float delta_a_weight(char layer,koord v,char k,char k_seed);
    float delta_b_weight(char layer,char k_seed);
    int bulat(double angka);
    char sebelah(koord n,char input[19][19]);
    tetangga hitung_tetangga(koord n,char input[19][19],char jauh,char urut);
    koord sambung(koord n,char x,char y);
    int pilih(int n,int m);
    void resample();

    char jL[4][64];
    char U0[19][19];

    float US1[12][19][19];
    float UC1[8][21][21];
    float US2[80][21][21];
    float UC2[33][13][13];
    float US3[97][13][13];
    float UC3[64][7][7];
    float US4[47][3][3];
    float UC4[35][1][1];
    float UV1[19][19];
    float UV2[21][21];
    float UV3[13][13];
    float UV4[3][3];

    char tot[19][19];
    float PC[9][8];
    float p_weight[35][9][8];

    float a1[12][3][3];
    float a2[80][8][5][5];
    float a3[97][33][5][5];
    float a4[47][64][5][5];
    float b1[12],b2[80],b3[97],b4[47];

    CCanvasDlg m_dlgCanvas;
    CStatDlg m_dlgStat;


public:
    CTa1Dlg(CWnd* pParent = NULL);    // standard constructor

// Dialog Data
    //{{AFX_DATA(CTa1Dlg)
    enum { IDD = IDD_TA1_DIALOG };
    CString    m_hasil;
    CString    m_sblm;
    //}}AFX_DATA

    // ClassWizard generated virtual function overrides
    //{{AFX_VIRTUAL(CTa1Dlg)
```

```
    protected:
    virtual void DoDataExchange(CDataExchange* pDX);   // DDX/DDV support
    //}}AFX_VIRTUAL

// Implementation
protected:
    HICON m_hIcon;

    // Generated message map functions
    //{{AFX_MSG(CTa1Dlg)
    virtual BOOL OnInitDialog();
    afx_msg void OnSysCommand(UINT nID, LPARAM lParam);
    afx_msg void OnPaint();
    afx_msg HCURSOR OnQueryDragIcon();
    afx_msg void OnHapusButton();
    afx_msg void OnKenaliButton();
    //}}AFX_MSG
    DECLARE_MESSAGE_MAP()
};

//{{AFX_INSERT_LOCATION}}
// Microsoft Visual C++ will insert additional declarations immediately before the previous line.

#endif // !defined
(AFX_TA1DLG_H__036827B0_F772_420D_9418_B8B4C89A8A5E__INCLUDED_)
-------------------------------------------------------------------------------------------------------------------------
-

// ta1Dlg.cpp : implementation file
//

#include "stdafx.h"
#include "ta1.h"
#include "ta1Dlg.h"
#include "CanvasDlg.h"
#include <conio.h>
#include <stdlib.h>
#include <math.h>
#include <string.h>
#include <fstream.h>


#ifdef _DEBUG
#define new DEBUG_NEW
#undef THIS_FILE
static char THIS_FILE[] = __FILE__;
#endif

/////////////////////////////////////////////////////////////////////////////
// CAboutDlg dialog used for App About

const char Dl[5]={0,1,3,2,1};
const float do_bar[5]={0,4.0,4.0,2.5,1.0};
const float do_doang[5]={0,0.9,0.8,0.7,1.0};
```

```
const char dimensi_US[5]={0,9,10,12,4};
const char plus_US[5]={0,1,1,2,4};
const char KSl[5]={0,12,80,97,47};

float gamma[5]={0,0.9,0.9,0.9,0.8};
float r4=1.0;
const char Al[5]={0,1,2,2,2};
const char KCl[5]={1,8,33,64,35};
const char dimensi_UC[5]={9,10,12,12,0};
const char plus_UC[5]={1,1,2,4,1};

const char n1[12]={1,1,1,1,1,1,1,1,1,1,1,1};
const char n2[80]=
{4,4,4,4,4,2,4,3,4,2,3,4,4,2,2,4,3,4,3,3,4,3,3,4,4,3,3,3,3,3,4,3,2,5,4,3,3,3,3,3,4,3,3,3,3,3,3,2,3,
        3,4,3,3,3,3,3,3,3,3,3,3,4,3,3,3,3,3,3,2,3,2,3,2,4,4,2,2,2,2};
const char n3[97]={2,3,2,2,2,2,2,2,2,3,2,2,2,1,1,2,3,1,2,2,4,4,2,2,2,
                   2,2,2,2,2,3,2,2,2,2,2,3,2,1,2,3,2,1,3,2,2,2,3,2,2,
                      2,2,3,3,2,2,2,2,2,2,2,2,2,2,2,2,2,3,3,3,3,2,1,2,2,
                      2,2,2,2,2,1,2,2,2,2,2,2,1,2,2,2,2,2,2,2,2,2};
const char n4[47]={2,2,2,2,2,3,4,1,1,2,2,2,2,2,2,1,2,2,1,2,2,1,2,2,
                         1,2,2,2,2,2,2,2,1,2,2,1,2,2,2,2,2,2,2,1,2,2,2,2};
const char seed_x_US3[97]={5,5,5,5,5,6,4,4,4,4,4,6,5,5,5,4,6,6,6,5,6,6,4,4,4,
                           4,4,6,4,4,4,4,4,4,6,6,4,6,6,4,4,6,6,6,5,5,6,6,5,5,4,
                           4,6,6,6,5,4,4,6,5,5,5,5,5,5,6,5,6,6,6,6,6,6,6,4,5,
                           4,5,5,5,5,5,6,4,4,6,5,5,5,5,4,4,4,6,6,5,4,5};
const char seed_y_US3[97]={4,4,6,4,6,5,5,4,4,6,6,6,5,5,5,5,5,5,5,4,6,6,4,4,5,
                           5,5,4,6,6,5,5,5,4,4,4,4,4,5,5,5,4,4,4,4,5,5,6,6,5,
                           6,6,5,5,6,4,6,4,4,6,4,5,5,6,5,5,6,6,6,6,6,6,6,5,6,
                           6,5,5,5,5,5,4,4,5,5,5,5,6,6,4,6,6,6,6,6,4,5};

class CAboutDlg : public CDialog
{
public:
    CAboutDlg();

// Dialog Data
    //{{AFX_DATA(CAboutDlg)
    enum { IDD = IDD_ABOUTBOX };
    //}}AFX_DATA

    // ClassWizard generated virtual function overrides
    //{{AFX_VIRTUAL(CAboutDlg)
    protected:
    virtual void DoDataExchange(CDataExchange* pDX);    // DDX/DDV support
    //}}AFX_VIRTUAL

// Implementation
protected:
    //{{AFX_MSG(CAboutDlg)
    //}}AFX_MSG
    DECLARE_MESSAGE_MAP()
};

CAboutDlg::CAboutDlg() : CDialog(CAboutDlg::IDD)
```

```
{
    //{{AFX_DATA_INIT(CAboutDlg)
    //}}AFX_DATA_INIT
}

void CAboutDlg::DoDataExchange(CDataExchange* pDX)
{
    CDialog::DoDataExchange(pDX);
    //{{AFX_DATA_MAP(CAboutDlg)
    //}}AFX_DATA_MAP
}

BEGIN_MESSAGE_MAP(CAboutDlg, CDialog)
    //{{AFX_MSG_MAP(CAboutDlg)
        // No message handlers
    //}}AFX_MSG_MAP
END_MESSAGE_MAP()

/////////////////////////////////////////////////////////////////////////////
// CTa1Dlg dialog

CTa1Dlg::CTa1Dlg(CWnd* pParent /*=NULL*/)
    : CDialog(CTa1Dlg::IDD, pParent)
{
    //{{AFX_DATA_INIT(CTa1Dlg)
    m_hasil = _T("");
    m_sblm = _T("");
    //}}AFX_DATA_INIT
    // Note that LoadIcon does not require a subsequent DestroyIcon in Win32

    m_hIcon = AfxGetApp()->LoadIcon(IDR_MAINFRAME);
}

void CTa1Dlg::DoDataExchange(CDataExchange* pDX)
{
    CDialog::DoDataExchange(pDX);
    //{{AFX_DATA_MAP(CTa1Dlg)
    DDX_Text(pDX, IDC_HASIL, m_hasil);
    DDX_Text(pDX, IDC_SBLM, m_sblm);
    //}}AFX_DATA_MAP
}

BEGIN_MESSAGE_MAP(CTa1Dlg, CDialog)
    //{{AFX_MSG_MAP(CTa1Dlg)
    ON_WM_SYSCOMMAND()
    ON_WM_PAINT()
    ON_WM_QUERYDRAGICON()
    ON_BN_CLICKED(IDC_BUTTON2, OnHapusButton)
    ON_BN_CLICKED(IDC_BUTTON1, OnKenaliButton)
    //}}AFX_MSG_MAP
END_MESSAGE_MAP()

/////////////////////////////////////////////////////////////////////////////
// CTa1Dlg message handlers
```

```
BOOL CTa1Dlg::OnInitDialog()
{
    CDialog::OnInitDialog();


    // Add "About..." menu item to system menu.

    // IDM_ABOUTBOX must be in the system command range.
    ASSERT((IDM_ABOUTBOX & 0xFFF0) == IDM_ABOUTBOX);
    ASSERT(IDM_ABOUTBOX < 0xF000);

    CMenu* pSysMenu = GetSystemMenu(FALSE);
    if (pSysMenu != NULL)
    {
        CString strAboutMenu;
        strAboutMenu.LoadString(IDS_ABOUTBOX);
        if (!strAboutMenu.IsEmpty())
        {
            pSysMenu->AppendMenu(MF_SEPARATOR);
            pSysMenu->AppendMenu(MF_STRING, IDM_ABOUTBOX, strAboutMenu);
        }
    }

    // Set the icon for this dialog.  The framework does this automatically
    //  when the application's main window is not a dialog
    SetIcon(m_hIcon, TRUE);             // Set big icon
    SetIcon(m_hIcon, FALSE);       // Set small icon

    // TODO: Add extra initialization here

    m_dlgCanvas.Create(IDD_CANVAS_DLG,this);
    m_dlgCanvas.ShowWindow(SW_SHOW);

    m_dlgStat.Create(IDD_STAT_DLG,this);

    inisialisasi_jL();
    inisialisasi_a_weight();
    inisialisasi_b_weight();


    return TRUE;  // return TRUE  unless you set the focus to a control
}

void CTa1Dlg::OnSysCommand(UINT nID, LPARAM lParam)
{
    if ((nID & 0xFFF0) == IDM_ABOUTBOX)
    {
        CAboutDlg dlgAbout;
        dlgAbout.DoModal();
    }
    else
    {
        CDialog::OnSysCommand(nID, lParam);
```

```
    }
}

// If you add a minimize button to your dialog, you will need the code below
//  to draw the icon.  For MFC applications using the document/view model,
//  this is automatically done for you by the framework.

void CTa1Dlg::OnPaint()
{
    if (IsIconic())
    {
        CPaintDC dc(this); // device context for painting

        SendMessage(WM_ICONERASEBKGND, (WPARAM) dc.GetSafeHdc(), 0);

        // Center icon in client rectangle
        int cxIcon = GetSystemMetrics(SM_CXICON);
        int cyIcon = GetSystemMetrics(SM_CYICON);
        CRect rect;
        GetClientRect(&rect);
        int x = (rect.Width() - cxIcon + 1) / 2;
        int y = (rect.Height() - cyIcon + 1) / 2;

        // Draw the icon
        dc.DrawIcon(x, y, m_hIcon);
    }
    else
    {
        CDialog::OnPaint();
    }
}

// The system calls this to obtain the cursor to display while the user drags
//  the minimized window.
HCURSOR CTa1Dlg::OnQueryDragIcon()
{
    return (HCURSOR) m_hIcon;
}

void CTa1Dlg::OnHapusButton()
{
    m_dlgCanvas.Invalidate();
}

void CTa1Dlg::inisialisasi_a_weight()
{
    ifstream input;
    char file[10];
    char index[3];
    char i,j,k,l;

    for(i=0;i<KSl[1];i++) {
        strcpy(file,"A");
        itoa(1,index,10);
```

```
    strcat(file,index);
    strcat(file,"-");
    itoa(i+1,index,10);
    strcat(file,index);
    strcat(file,".wgt");
    input.open(file,ios::binary);

    for(l=0;l<3;l++) {
        for(k=0;k<3;k++) {
            input.read((char *) &a1[i][k][l],sizeof(float));
        }
    }

    input.close();
}

for(i=0;i<80;i++) {
    strcpy(file,"A");
    itoa(2,index,10);
    strcat(file,index);
    strcat(file,"-");
    itoa(i+1,index,10);
    strcat(file,index);
    strcat(file,".wgt");
    input.open(file,ios::binary);

    for(j=0;j<8;j++) {
        for(l=0;l<5;l++) {
            for(k=0;k<5;k++) {
                input.read((char *) &a2[i][j][k][l],sizeof(float));
            }
        }
    }

    input.close();

}

    for(i=0;i<KSl[3];i++) {
    strcpy(file,"A");
    itoa(3,index,10);
    strcat(file,index);
    strcat(file,"-");
    itoa(i+1,index,10);
    strcat(file,index);
    strcat(file,".wgt");
    input.open(file,ios::binary);

    for(j=0;j<33;j++) {
        for(l=0;l<5;l++) {
            for(k=0;k<5;k++) {
                input.read((char *) &a3[i][j][k][l],sizeof(float));
            }
        }
```

```
        }

        input.close();

    }

    for(i=0;i<KSl[4];i++) {
        strcpy(file,"A");
        itoa(4,index,10);
        strcat(file,index);
        strcat(file,"-");
        itoa(i+1,index,10);
        strcat(file,index);
        strcat(file,".wgt");
        input.open(file,ios::binary);

        for(j=0;j<64;j++) {
            for(l=0;l<5;l++) {
                for(k=0;k<5;k++) {
                    input.read((char *) &a4[i][j][k][l],sizeof(float));
                }
            }
        }

        input.close();

    }

    for(i=0;i<35;i++) {
        strcpy(file,"P");
        itoa(i,index,10);
        strcat(file,index);
        strcat(file,".wgt");
        input.open(file);
        input.read((char *) p_weight[i],sizeof(p_weight[i]));
        input.close();
    }
}

void CTa1Dlg::inisialisasi_b_weight()
{
    ifstream input;

    input.open("b1.wgt",ios::binary);
    input.read((char *) b1,sizeof(b1));
    input.close();

    input.open("b2.wgt",ios::binary);
    input.read((char *) b2,sizeof(b2));
    input.close();

    input.open("b3.wgt",ios::binary);
    input.read((char *) b3,sizeof(b3));
    input.close();
```

```
    input.open("b4.wgt",ios::binary);
    input.read((char *) b4,sizeof(b4));
    input.close();

}



void CTa1Dlg::inisialisasi_jL()
{
    char idx1;
    char tmp1[7][21]= {{0,2,4,6},
                       {1,3,5,7},
                       {4,8,9,14,26,27,30},
                       {6,10,18,19},
                       {0,6,7,13,14,17,19,21,22,24,25,26,27,31,39,44,45,46,55,59,60},
                       {9,11,15,18},
                       {1,4,5,6,9,12,16,19,20,26}};

    for(idx1=0;idx1<64;idx1++) jL[0][idx1]=0;
    for(idx1=0;idx1<4;idx1++) jL[0][tmp1[0][idx1]]=1;
    for(idx1=0;idx1<4;idx1++) jL[0][tmp1[1][idx1]]=2;

    for(idx1=0;idx1<64;idx1++) jL[1][idx1]=0;
    for(idx1=0;idx1<33;idx1++) jL[1][idx1]=1;
    for(idx1=0;idx1<7;idx1++) jL[1][tmp1[2][idx1]]=2;
    jL[1][7]=jL[1][24]=jL[1][25]=3;
    for(idx1=0;idx1<4;idx1++) jL[1][tmp1[3][idx1]]=4;
    jL[1][5]=jL[1][11]=5;
    jL[1][16]=jL[1][17]=8;

    for(idx1=0;idx1<64;idx1++) jL[2][idx1]=1;
    for(idx1=0;idx1<21;idx1++) jL[2][tmp1[4][idx1]]=2;
    for(idx1=0;idx1<4;idx1++) jL[2][tmp1[5][idx1]]=3;
    jL[2][50]=5;

    for(idx1=0;idx1<64;idx1++) jL[3][idx1]=0;
    for(idx1=0;idx1<35;idx1++) jL[3][idx1]=1;
    for(idx1=0;idx1<10;idx1++) jL[3][tmp1[6][idx1]]=2;
    jL[3][25]=3;

}

char CTa1Dlg::hitung_jL(char layer,char kS,char kC)
{
    char tmp2=0;
    char tmp1,idx1;

    for(idx1=0;idx1<=kC;idx1++) tmp2+=jL[layer-1][idx1];
    if( (kS+1>(tmp2-jL[layer-1][kC])) && (kS+1<=tmp2) )tmp1=1;
    else tmp1=0;
```

```
    return tmp1;
}

koord CTa1Dlg::seed_cell(char layer,char k_seed)
{
    koord tmp1;

    if (layer==1) tmp1.x=tmp1.y=0;
    if (layer==2) tmp1.x=tmp1.y=0;
    if (layer==3) {
        tmp1.x=(seed_x_US3[k_seed]-5)*2;
        tmp1.y=(seed_y_US3[k_seed]-5)*2;
    }
    if (layer==4) tmp1.x=tmp1.y=0;

    return tmp1;
}

float CTa1Dlg::a_weight(char layer,koord v,char kC,char kS)
{
    float tmp1;
    koord tmp2;

    tmp2.x=v.x+Al[layer];
    tmp2.y=v.y+Al[layer];

    if(layer==1) tmp1=a1[kS][tmp2.x][tmp2.y];
    if(layer==2) tmp1=a2[kS][kC][tmp2.x][tmp2.y];
    if(layer==3) tmp1=a3[kS][kC][tmp2.x][tmp2.y];
    if(layer==4) tmp1=a4[kS][kC][tmp2.x][tmp2.y];

    return tmp1;

}

float CTa1Dlg::b_weight(char layer,char k)
{
    float tmp1;

    if(layer==1) tmp1=b1[k];
    if(layer==2) tmp1=b2[k];
    if(layer==3) tmp1=b3[k];
    if(layer==4) tmp1=b4[k];

    return tmp1;
}

char CTa1Dlg::hitung_U0(koord n)
{
    return U0[n.x+dimensi_UC[0]][n.y+dimensi_UC[0]];
}
```

```
float CTa1Dlg::rL(char layer,char k)
{
    float tmp1;

    if(layer==1) tmp1=1.7;
    if(layer==2) {
        if(k==39||k==41||k==40||k==47||k==48||k==46) tmp1=3.8;
        else tmp1=4.0;
    }
    if(layer==3) tmp1=1.5;
    if(layer==4) tmp1=r4;

    return tmp1;
}

float CTa1Dlg::UC(char layer,koord n,char k)
{
    float tmp1;

    if(layer==0) tmp1=hitung_U0(n);

    if(layer==1) tmp1=UC1[k][n.x+dimensi_UC[layer]][n.y+dimensi_UC[layer]];
    if(layer==2) tmp1=UC2[k][(n.x+dimensi_UC[layer])/plus_UC[layer]]
                    [(n.y+dimensi_UC[layer])/plus_UC[layer]];
    if(layer==3) tmp1=UC3[k][(n.x+dimensi_UC[layer])/plus_UC[layer]]
                    [(n.y+dimensi_UC[layer])/plus_UC[layer]];
    if(layer==4) tmp1=UC4[k][n.x][n.y];

    return tmp1;
}

float CTa1Dlg::UV(char layer,koord n)
{
    float tmp1;

    if(layer==1) tmp1=UV1[n.x+dimensi_US[layer]][n.y+dimensi_US[layer]];
    if(layer==2) tmp1=UV2[n.x+dimensi_US[layer]][n.y+dimensi_US[layer]];
    if(layer==3) tmp1=UV3[(n.x+dimensi_US[layer])/plus_US[layer]]
                    [(n.y+dimensi_US[layer])/plus_US[layer]];
    if(layer==4) tmp1=UV4[(n.x+dimensi_US[layer])/plus_US[layer]]
                    [(n.y+dimensi_US[layer])/plus_US[layer]];

    return tmp1;
}

float CTa1Dlg::US(char layer,koord n,char k)
{
    float tmp1;

    if(layer==1) tmp1=US1[k][n.x+dimensi_US[layer]][n.y+dimensi_US[layer]];
    if(layer==2) tmp1=US2[k][n.x+dimensi_US[layer]][n.y+dimensi_US[layer]];
    if(layer==3) tmp1=US3[k][(n.x+dimensi_US[layer])/plus_US[layer]]
                    [(n.y+dimensi_US[layer])/plus_US[layer]];
    if(layer==4) tmp1=US4[k][(n.x+dimensi_US[layer])/plus_US[layer]]
```

```
                    [(n.y+dimensi_US[layer])/plus_US[layer]];

    return tmp1;
}


float CTa1Dlg::nilai_UV(char layer,koord n)
{
    char idx1;
    float tmp1,pangkat,c_weight;
    koord tmp2,tmp3;

    tmp1=0.0;
    for(idx1=0;idx1<KCl[layer-1];idx1++) {
        for(tmp2.x=(0-Al[layer]);tmp2.x<=Al[layer];(tmp2.x)++) {
            for(tmp2.y=(0-Al[layer]);tmp2.y<=Al[layer];tmp2.y++) {
                pangkat=sqrt((tmp2.x*tmp2.x)+(tmp2.y*tmp2.y));
                c_weight=pow(gamma[layer],pangkat);
                tmp3.x=n.x+(tmp2.x*plus_UC[layer-1]);
                tmp3.y=n.y+(tmp2.y*plus_UC[layer-1]);
                if((abs(tmp3.x)>dimensi_UC[layer-1])||(abs(tmp3.y)>dimensi_UC[layer-1])) tmp1+=0;
                else tmp1+=c_weight*UC(layer-1,tmp3,idx1)*UC(layer-1,tmp3,idx1);
            }
        }
    }

    tmp1=sqrt(tmp1);
    return tmp1;
}

float CTa1Dlg::nilai_US(char layer,koord n,char k)
{
    float tmp1,pembilang=0.0,penyebut;
    char idx1;
    koord tmp2,tmp3;

    for(idx1=0;idx1<KCl[layer-1];idx1++) {
        for(tmp2.x=(0-Al[layer]);tmp2.x<=Al[layer];(tmp2.x)++) {
            for(tmp2.y=(0-Al[layer]);tmp2.y<=Al[layer];tmp2.y++) {
                tmp3.x=n.x+(tmp2.x*plus_UC[layer-1]);
                tmp3.y=n.y+(tmp2.y*plus_UC[layer-1]);
                if((abs(tmp3.x)>dimensi_UC[layer-1])||(abs(tmp3.y)>dimensi_UC[layer-1]))
                    pembilang+=0;
                else pembilang+=a_weight(layer,tmp2,idx1,k)*UC(layer-1,tmp3,idx1);
            }
        }
    }
    pembilang+=1;
    penyebut=1+( rL(layer,k) * (1/(1+rL(layer,k))) *b_weight(layer,k) *UV(layer,n) );
    tmp1=(pembilang/penyebut)-1;

    if(tmp1<0) tmp1=0;

    tmp1*=rL(layer,k);
```

```
    return tmp1;

}

float CTa1Dlg::nilai_UC(char layer,koord n, char k)
{
    float tmp1,tmp3,d_weight,pangkat;
    char tmp2;
    koord tmp4,tmp5;
    char idx1;

    if(layer==0) tmp1=hitung_U0(n);
    else {
        tmp1=0.0;
        for(idx1=0;idx1<KSl[layer];idx1++) {
            tmp2=hitung_jL(layer,idx1,k);
            tmp3=0.0;
            for(tmp4.x=(0-Dl[layer]);tmp4.x<=Dl[layer];(tmp4.x)++) {
                for(tmp4.y=(0-Dl[layer]);tmp4.y<=Dl[layer];tmp4.y++) {
                    pangkat=sqrt((tmp4.x*tmp4.x)+(tmp4.y*tmp4.y));
                    d_weight=do_bar[layer]*pow(do_doang[layer],pangkat);
                    tmp5.x=n.x+(tmp4.x*plus_US[layer]);
                    tmp5.y=n.y+(tmp4.y*plus_US[layer]);
                    if((abs(tmp5.x)>dimensi_US[layer])||(abs(tmp5.y)>dimensi_US[layer])) tmp3+=0;
                    else tmp3+=d_weight*US(layer,tmp5,idx1);
                }
            }
            tmp1+=tmp2*tmp3;
        }
        tmp1=tmp1/(1.0+tmp1);
    }
    return tmp1;

}

float CTa1Dlg::nilai_PC(char kuadran,char k)
{
    char idx1,tmp2;
    float tmp1=0.0,tmp3,pangkat,d_weight;
    char x,y,x_atas,x_bawah,y_atas,y_bawah;

    x_bawah=(kuadran%3)*6;
    x_atas=x_bawah+6;
    y_bawah=(kuadran/3)*6;
    y_atas=y_bawah+6;

    for(idx1=0;idx1<12;idx1++) {
        tmp2=hitung_jL(1,idx1,k);
        tmp3=0.0;
        for(y=y_bawah;y<=y_atas;y++) {
            for(x=x_bawah;x<=x_atas;x++) {
                pangkat=sqrt((((x%6)-3)*((x%6)-3))+(((y%6)-3)*((y%6)-3)));
                d_weight=1*pow(1,pangkat);
```

```
                tmp3+=d_weight*US1[idx1][x][y];
            }
        }
        tmp1+=tmp2*tmp3;
    }
    tmp1=tmp1/(1.0+tmp1);

    return tmp1;
}


void CTa1Dlg::hitung_PC()
{
    char i,j;
    for(i=0;i<9;i++) {
        for(j=0;j<8;j++) {
            PC[i][j]=nilai_PC(i,j);
        }
    }
}

void CTa1Dlg::hitung_UV(char layer)
{
    koord n;
    if(layer==1) {
        for(n.x=(0-dimensi_US[layer]);n.x<=dimensi_US[layer];n.x++) {
            for(n.y=(0-dimensi_US[layer]);n.y<=dimensi_US[layer];n.y++) {
                UV1[n.x+dimensi_US[layer]][n.y+dimensi_US[layer]]=nilai_UV(layer,n);
            }
        }
    }
    if(layer==2) {
        for(n.x=(0-dimensi_US[layer]);n.x<=dimensi_US[layer];n.x++) {
            for(n.y=(0-dimensi_US[layer]);n.y<=dimensi_US[layer];n.y++) {
                UV2[n.x+dimensi_US[layer]][n.y+dimensi_US[layer]]=nilai_UV(layer,n);
            }
        }
    }
    if(layer==3) {
        for(n.x=(0-dimensi_US[layer]);n.x<=dimensi_US[layer];n.x+=plus_US[layer]) {
            for(n.y=(0-dimensi_US[layer]);n.y<=dimensi_US[layer];n.y+=plus_US[layer]) {
                UV3[(n.x+dimensi_US[layer])/plus_US[layer]][(n.y+dimensi_US[layer])/plus_US
[layer]]=nilai_UV(layer,n);
            }
        }
    }
    if(layer==4) {
        for(n.x=(0-dimensi_US[layer]);n.x<=dimensi_US[layer];n.x+=plus_US[layer]) {
            for(n.y=(0-dimensi_US[layer]);n.y<=dimensi_US[layer];n.y+=plus_US[layer]) {
                UV4[(n.x+dimensi_US[layer])/plus_US[layer]][(n.y+dimensi_US[layer])/plus_US
[layer]]=nilai_UV(layer,n);
            }
        }
    }
```

```
}

void CTa1Dlg::hitung_US(char layer)
{
    koord n;
    char k;

    if(layer==1) {
        for(k=0;k<12;k++) {
            for(n.x=(0-dimensi_US[layer]);n.x<=dimensi_US[layer];n.x++) {
                for(n.y=(0-dimensi_US[layer]);n.y<=dimensi_US[layer];n.y++) {
                    US1[k][(n.x+dimensi_US[layer])][(n.y+dimensi_US[layer])]=nilai_US(layer,n,k);
                }
            }
        }
    }
    if(layer==2) {
        for(k=0;k<80;k++) {
            for(n.x=(0-dimensi_US[layer]);n.x<=dimensi_US[layer];n.x++) {
                for(n.y=(0-dimensi_US[layer]);n.y<=dimensi_US[layer];n.y++) {
                    US2[k][n.x+dimensi_US[layer]][n.y+dimensi_US[layer]]=nilai_US(layer,n,k);
                }
            }
        }
    }
    if(layer==3) {
        for(k=0;k<97;k++) {
            for(n.x=(0-dimensi_US[layer]);n.x<=dimensi_US[layer];n.x+=plus_US[layer]) {
                for(n.y=(0-dimensi_US[layer]);n.y<=dimensi_US[layer];n.y+=plus_US[layer]) {
                    US3[k][(n.x+dimensi_US[layer])/plus_US[layer]][(n.y+dimensi_US[layer])/
plus_US[layer]]=nilai_US(layer,n,k);
                }
            }
        }
    }
    if(layer==4) {
        for(k=0;k<47;k++) {
            for(n.x=(0-dimensi_US[layer]);n.x<=dimensi_US[layer];n.x+=plus_US[layer]) {
                for(n.y=(0-dimensi_US[layer]);n.y<=dimensi_US[layer];n.y+=plus_US[layer]) {
                    US4[k][(n.x+dimensi_US[layer])/plus_US[layer]][(n.y+dimensi_US[layer])/
plus_US[layer]]=nilai_US(layer,n,k);
                }
            }
        }
    }

}

void CTa1Dlg::hitung_UC(char layer)
{
    koord n;
    char k;
    float a,b;
```

```
    if(layer==1) {
        for(k=0;k<KCl[layer];k++) {
            for(n.x=(0-dimensi_UC[layer]);n.x<=dimensi_UC[layer];n.x++) {
                for(n.y=(0-dimensi_UC[layer]);n.y<=dimensi_UC[layer];n.y++) {
                    UC1[k][(n.x+dimensi_UC[layer])/plus_UC[layer]][(n.y+dimensi_UC[layer])/
plus_UC[layer]]=nilai_UC(layer,n,k);
                }
            }
        }
    }

    if(layer==2) {
        for(k=0;k<KCl[layer];k++) {
            for(n.x=(0-dimensi_UC[layer]);n.x<=dimensi_UC[layer];n.x+=plus_UC[layer]) {
                for(n.y=(0-dimensi_UC[layer]);n.y<=dimensi_UC[layer];n.y+=plus_UC[layer]) {
                    UC2[k][(n.x+dimensi_UC[layer])/plus_UC[layer]][(n.y+dimensi_UC[layer])/
plus_UC[layer]]=nilai_UC(layer,n,k);
                }
            }
        }
    }
    if(layer==3) {
        for(k=0;k<KCl[layer];k++) {
            for(n.x=(0-dimensi_UC[layer]);n.x<=dimensi_UC[layer];n.x+=plus_UC[layer]) {
                for(n.y=(0-dimensi_UC[layer]);n.y<=dimensi_UC[layer];n.y+=plus_UC[layer]) {
                    UC3[k][(n.x+dimensi_UC[layer])/plus_UC[layer]][(n.y+dimensi_UC[layer])/
plus_UC[layer]]=nilai_UC(layer,n,k);
                }
            }
        }
    }
    if(layer==4) {
        for(k=0;k<KCl[layer];k++) {
            n.x=0;
            n.y=0;
            UC4[k][0][0]=nilai_UC(layer,n,k);
        }
    }
}

float CTa1Dlg::delta_a_weight(char layer,koord v,char k,char k_seed)
{
    float tmp1;
    koord tmp2;
    float gamma[5]={0,0.9,0.9,0.9,0.8};
    float pangkat,c_weight;

    tmp2=seed_cell(layer,k_seed);

    pangkat=sqrt((v.x*v.x)+(v.y*v.y));
    c_weight=pow(gamma[layer],pangkat);

    v.x*=plus_UC[layer-1];
```

```
    v.y*=plus_UC[layer-1];

    tmp2.x+=v.x;
    tmp2.y+=v.y;

    tmp1=10000*c_weight*UC(layer-1,tmp2,k);

    return tmp1;
}

float CTa1Dlg::delta_b_weight(char layer,char k_seed)
{
    float tmp1;
    koord tmp2;

    tmp2=seed_cell(layer,k_seed);
    tmp1=10000*UV(layer,tmp2);

    return tmp1;
}

int CTa1Dlg::bulat(double angka)
{
    double depan,belakang;
    int hasil;

    belakang=modf(angka,&depan);
    if(belakang<0.5) hasil=depan;
    else hasil=depan+1;

    return hasil;
}

char CTa1Dlg::sebelah(koord n,char input[19][19])
{
    koord m;
    char hsl=0;
    for(m.x=n.x-1;m.x<=n.x+1;m.x++) {
        for(m.y=n.y-1;m.y<=n.y+1;m.y++) {
            if((!((m.x<0)||(m.x>18)||(m.y<0)||(m.y>18)))&&(input[m.x][m.y]==1)) hsl++;
        }
    }
    return hsl;
}


tetangga CTa1Dlg::hitung_tetangga(koord n,char input[19][19],char jauh,char urut=0)
{
    koord k;
    tetangga hasil;
    koord tmp1[16];
    char nb;
    hasil.banyak=0;
```

```
    for(k.x=n.x-jauh;k.x<=n.x+jauh;k.x+=2*jauh)
        for(k.y=n.y-jauh;k.y<=n.y+jauh;k.y++)
            if(!((k.x<0)||(k.x>18)||(k.y<0)||(k.y>18))) {
                if(input[k.x][k.y]==1) {
                    nb=sebelah(k,input);
                    if (nb<3) {
                        tmp1[hasil.banyak].x=k.x;
                        tmp1[hasil.banyak].y=k.y;
                        hasil.banyak++;
                    }
                }
            }
    for(k.y=n.y-jauh;k.y<=n.y+jauh;k.y+=2*jauh)
        for(k.x=n.x-jauh-1;k.x<=n.x+jauh-1;k.x++)
            if(!((k.x<0)||(k.x>18)||(k.y<0)||(k.y>18)))
                if(input[k.x][k.y]==1) {
                    nb=sebelah(k,input);
                    if(nb<3) {
                        tmp1[hasil.banyak].x=k.x;
                        tmp1[hasil.banyak].y=k.y;
                        hasil.banyak++;
                    }
                }

    hasil.x=tmp1[urut-1].x;
    hasil.y=tmp1[urut-1].y;

    return hasil;

}

koord CTa1Dlg::sambung(koord n,char x,char y)
{
    koord tmp;

    tmp.x=x-n.x;
    tmp.y=y-n.y;

    tmp.x/=2;
    tmp.y/=2;

    tmp.y+=n.y;
    tmp.x+=n.x;


    if(!((tmp.x<0)||(tmp.x>18)||(tmp.y<0)||(tmp.y>18))) return tmp;
    else return n;

}

int CTa1Dlg::pilih(int n,int m)
{
    float a;
    if(m==0) a=0.00001; else a=m;
```

```
    if((n/a)<0.5) return m;
    else return n;
}


void CTa1Dlg::resample()
{
    char i,j,vert,hort,aa;
    koord n,o,p,q;
    float c,d;
    char e,f;
    tetangga nb;
    char jauh;

    for(n.y=-9;n.y<=9;n.y++) {
        for(n.x=-9;n.x<=9;n.x++) {
            tot[n.x+9][n.y+9]=0;
        }

    }
    p.x=p.y=-10;
    o.x=o.y=10;
    for(n.x=-9;n.x<10;n.x++) {
        for(n.y=-9;n.y<10;n.y++) {
            if(U0[n.x+9][n.y+9]==1) {
                if(o.x>n.x) o.x=n.x;
                if(o.y>n.y) o.y=n.y;
                if(p.x<n.x) p.x=n.x;
                if(p.y<n.y) p.y=n.y;
            }
        }
    }

    if(((p.x-o.x)%2)!=0) o.x--;
    if(((p.y-o.y)%2)!=0) o.y--;

    hort=p.x-o.x;
    vert=p.y-o.y;
    cout << hort+0 << " " << vert+0 << endl;
    i=o.x+(hort/2);
    j=o.y+(vert/2);


    int h0,v0;
    if(hort==0) h0=999999; else h0=19/hort;
    if(vert==0) v0=999999; else v0=19/vert;

    if((h0==3)||(v0==3)) jauh=3; else jauh=2;
    for(n.x=o.x;n.x<=p.x;n.x++) {
        for(n.y=o.y;n.y<=p.y;n.y++) {
            if(U0[n.x+9][n.y+9]==1) {

                if(hort!=0) c=(n.x-i)*9/pilih(i-o.x,j-o.y);
                else c=i;
```

```
                if(vert!=0) d=(n.y-j)*9/pilih(j-o.y,i-o.x);
                else d=j;
                e=9+bulat(c);
                f=9+bulat(d);

                tot[e][f]=1;
            }
        }
    }


    for(n.y=0;n.y<19;n.y++) {
        for(n.x=0;n.x<19;n.x++) {
            U0[n.x][n.y]=tot[n.x][n.y];
        }
    }


    for(n.y=0;n.y<19;n.y++) {
        for(n.x=0;n.x<19;n.x++) {
            if(U0[n.x][n.y]==1) {
                nb=hitung_tetangga(n,tot,jauh);
                for(aa=0;aa<nb.banyak;aa++) {
                    nb=hitung_tetangga(n,tot,jauh,aa+1);
                    q=sambung(n,nb.x,nb.y);
                    tot[q.x][q.y]=1;
                }
            }
        }
    }

    for(n.y=0;n.y<19;n.y++) {
        for(n.x=0;n.x<19;n.x++) {
            U0[n.x][n.y]=tot[n.x][n.y];
        }
    }

}



void CTa1Dlg::OnKenaliButton()
{
    char x,y;
    char aa,nsatu,ndua,nsatu1,ndua1,banyak,banyak1;
    koord n;
    float pengali,temp,satu,dua,satu1,dua1;
    CString t;
    char sementara[19][19];
    float tes[35];



    m_dlgStat.ShowWindow(1);
```

```
RedrawWindow();
m_dlgStat.RedrawWindow();


UpdateData(TRUE);
for(y=0;y<19;y++) {
    for(x=0;x<19;x++) {
        sementara[x][y]=m_dlgCanvas.input_mouse[x][y];
    }
}

for(y=0;y<19;y++) {
    for(x=0;x<19;x++) {
        U0[x][y]=sementara[x][y];
    }
}



hapus_noise();


hitung_UV(1);
hitung_US(1);
hitung_UC(1);
hitung_UV(2);
hitung_US(2);
hitung_UC(2);
hitung_UV(3);
hitung_US(3);
hitung_UC(3);

r4=1.0;
do {

    hitung_UV(4);
    hitung_US(4);
    hitung_UC(4);

    for(aa=1;aa<35;aa++) {
        tes[aa]=UC4[aa][0][0];
    }

    satu=UC4[0][0][0];
    nsatu=0;
    dua=0.0;
    if(satu > 0) banyak=1; else banyak=0;
    for(aa=1;aa<35;aa++) {
        if(UC4[aa][0][0]>dua) {
            if(UC4[aa][0][0]>satu) {
                dua=satu;
                ndua=nsatu;
                satu=UC4[aa][0][0];
                nsatu=aa;
```

```
            }
            else {
                dua=UC4[aa][0][0];
                ndua=aa;
            }

        }
        if(UC4[aa][0][0]>0) banyak++;
    }

    if(banyak>0) break;
    else r4-=0.1;

} while(r4>0);

char dat=nsatu;

if(nsatu<10) nsatu+=48;
else if(nsatu>23) nsatu+=56;
else nsatu+=55;

if(banyak<1) {
    t.Format("Tidak dikenali");
    m_sblm=t;
}
else {
    t.Format("%c",nsatu);
    m_sblm=t;
}
nsatu=dat;
UpdateData(FALSE);


r4=1.0;
resample();


hitung_UV(1);
hitung_US(1);
hitung_PC();


for(aa=0;aa<35;aa++) {
    if(UC4[aa][0][0]>0) {
        pengali=1.0;
        for(n.x=0;n.x<9;n.x++) {
            for(n.y=0;n.y<8;n.y++) {
                if(p_weight[aa][n.x][n.y]==1) {
                    temp=p_weight[aa][n.x][n.y]*PC[n.x][n.y];
                    if(temp>0.5) {
                        pengali*=1.0;
                    }
                    else pengali*=0.0;
                }
```

```
                if(p_weight[aa][n.x][n.y]==-1) {
                    temp=p_weight[aa][n.x][n.y]*PC[n.x][n.y];
                    if(temp<-0.5) pengali*=0.0;
                    else {
                        pengali*=1.0;
                    }
                }
            }
        }
        UC4[aa][0][0]*=pengali;
    }
}

satu1=UC4[0][0][0];
nsatu1=0;
dua1=0.0;
if(satu1 > 0) banyak1=1; else banyak1=0;
for(aa=1;aa<35;aa++) {
    if(UC4[aa][0][0]>dua1) {
        if(UC4[aa][0][0]>satu1) {
            dua1=satu1;
            ndua1=nsatu1;
            satu1=UC4[aa][0][0];
            nsatu1=aa;
        }
        else {
            dua1=UC4[aa][0][0];
            ndua1=aa;
        }

    }
    if(UC4[aa][0][0]>0) banyak1++;
}


if(nsatu<10) nsatu+=48;
else if(nsatu>23) nsatu+=56;
else nsatu+=55;
if(nsatu1<10) nsatu1+=48;
else if(nsatu1>23) nsatu1+=56;
else nsatu1+=55;

m_dlgStat.ShowWindow(0);

if(banyak1<1) {
    if(banyak>0) {
        t.Format("%c",nsatu);
        m_hasil=t;
    }
    else {
        t.Format("Tidak dikenali");
        m_hasil=t;
    }
}
```

```
    else {
        t.Format("%c",nsatu1);
        m_hasil=t;
    }
    UpdateData(FALSE);

}


void CTa1Dlg::hapus_noise()
{
    koord n;
    char nb;
    for(n.x=0;n.x<19;n.x++) {
        for(n.y=0;n.y<19;n.y++) {
            nb=sebelah(n,U0);
            if(nb<2) U0[n.x][n.y]=0;
        }
    }

}
-----------------------------------------------------------------------------------------------------------------------
-

#if !defined
(AFX_CANVASDLG_H__68DBBA84_5EDA_45E9_9C2E_425A574B74AA__INCLUDED_)
#define AFX_CANVASDLG_H__68DBBA84_5EDA_45E9_9C2E_425A574B74AA__INCLUDED_

#if _MSC_VER > 1000
#pragma once
#endif // _MSC_VER > 1000
// CanvasDlg.h : header file
//

/////////////////////////////////////////////////////////////////////////////
// CCanvasDlg dialog

class CCanvasDlg : public CDialog
{
// Construction
public:
    char input_mouse[19][19];

    CCanvasDlg(CWnd* pParent = NULL);   // standard constructor

// Dialog Data
    //{{AFX_DATA(CCanvasDlg)
    enum { IDD = IDD_CANVAS_DLG };
        // NOTE: the ClassWizard will add data members here
    //}}AFX_DATA


// Overrides
    // ClassWizard generated virtual function overrides
```

```
    //{{AFX_VIRTUAL(CCanvasDlg)
    protected:
    virtual void DoDataExchange(CDataExchange* pDX);    // DDX/DDV support
    //}}AFX_VIRTUAL

// Implementation
protected:

    // Generated message map functions
    //{{AFX_MSG(CCanvasDlg)
    afx_msg void OnMouseMove(UINT nFlags, CPoint point);
    afx_msg void OnLButtonDown(UINT nFlags, CPoint point);
    virtual BOOL OnInitDialog();
    afx_msg void OnLButtonUp(UINT nFlags, CPoint point);
    afx_msg void OnRButtonDown(UINT nFlags, CPoint point);
    //}}AFX_MSG
    DECLARE_MESSAGE_MAP()
private:
    int m_iPrevY;
    int m_iPrevX;
};

//{{AFX_INSERT_LOCATION}}
// Microsoft Visual C++ will insert additional declarations immediately before the previous line.

#endif // !defined
(AFX_CANVASDLG_H__68DBBA84_5EDA_45E9_9C2E_425A574B74AA__INCLUDED_)

-----------------------------------------------------------------------------------------------------------------------
-

// CanvasDlg.cpp : implementation file
//

#include "stdafx.h"
#include "ta1.h"
#include "CanvasDlg.h"
#include "ta1Dlg.h"
#include <stdlib.h>

#ifdef _DEBUG
#define new DEBUG_NEW
#undef THIS_FILE
static char THIS_FILE[] = __FILE__;
#endif

/////////////////////////////////////////////////////////////////////////////
// CCanvasDlg dialog


CCanvasDlg::CCanvasDlg(CWnd* pParent /*=NULL*/)
    : CDialog(CCanvasDlg::IDD, pParent)
{
    //{{AFX_DATA_INIT(CCanvasDlg)
```

```
        // NOTE: the ClassWizard will add member initialization here
    //}}AFX_DATA_INIT
}


void CCanvasDlg::DoDataExchange(CDataExchange* pDX)
{
    CDialog::DoDataExchange(pDX);
    //{{AFX_DATA_MAP(CCanvasDlg)
        // NOTE: the ClassWizard will add DDX and DDV calls here
    //}}AFX_DATA_MAP
}


BEGIN_MESSAGE_MAP(CCanvasDlg, CDialog)
    //{{AFX_MSG_MAP(CCanvasDlg)
    ON_WM_MOUSEMOVE()
    ON_WM_LBUTTONDOWN()
    ON_WM_LBUTTONUP()
    ON_WM_RBUTTONDOWN()
    //}}AFX_MSG_MAP
END_MESSAGE_MAP()

/////////////////////////////////////////////////////////////////////////////
// CCanvasDlg message handlers

void CCanvasDlg::OnMouseMove(UINT nFlags, CPoint point)
{
    // TODO: Add your message handler code here and/or call default
    if((nFlags & MK_LBUTTON) == MK_LBUTTON)
    {
        CClientDC dc(this);

        dc.MoveTo(m_iPrevX,m_iPrevY);
        dc.LineTo(point.x,point.y);

        m_iPrevX=point.x;
        m_iPrevY=point.y;
        dc.SaveDC();
    }



    CDialog::OnMouseMove(nFlags, point);
}

void CCanvasDlg::OnLButtonDown(UINT nFlags, CPoint point)
{
    // TODO: Add your message handler code here and/or call default

    m_iPrevX=point.x;
    m_iPrevY=point.y;

    CDialog::OnLButtonDown(nFlags, point);
```

```
}



BOOL CCanvasDlg::OnInitDialog()
{
    CDialog::OnInitDialog();

/*  CString tes;
    CRect tes1;
    GetClientRect(tes1);
    tes1.NormalizeRect();

    tes.Format("%d %d",tes1.right,tes1.bottom);
    MessageBox(tes);
*/

    return TRUE;  // return TRUE unless you set the focus to a control
                  // EXCEPTION: OCX Property Pages should return FALSE
}

void CCanvasDlg::OnLButtonUp(UINT nFlags, CPoint point)
{

    char x,y,jml,xx,yy;
    CString tes;
//  unsigned long piksel[95][95];
//  unsigned long piksel[190][190];

    CClientDC dc(this);
/*  for(x=0;x<190;x++) {
        for(y=0;y<190;y++) {
            piksel[x][y]=dc.GetPixel(x,y);
        }
    }
*/
    for(y=0;y<19;y++) {
        for(x=0;x<19;x++) {
            jml=0;
            for(xx=x*5;xx<(x+1)*5;xx++){
                for(yy=y*5;yy<(y+1)*5;yy++){
                    if(dc.GetPixel(xx,yy)==0) jml++;
                }
            }
            if(jml>2) {
                input_mouse[x][y]=1;
/*              for(xx=x*5;xx<(x+1)*5;xx++){
                    for(yy=y*5;yy<(y+1)*5;yy++){
                        dc.SetPixel(xx,yy,0);
                    }
                }
*/          }
            else input_mouse[x][y]=0;
        }
```

```
    }

    UpdateData(FALSE);

    CDialog::OnLButtonUp(nFlags, point);
}

void CCanvasDlg::OnRButtonDown(UINT nFlags, CPoint point)
{

    CClientDC dc(this);
    CString tes;
    tes.Format("%d",dc.GetPixel(point));
    MessageBox(tes);
    CDialog::OnRButtonDown(nFlags, point);
}
-----------------------------------------------------------------------------------------------------------------------
-

#if !defined(AFX_STATDLG_H__C6CBFC75_6A42_4783_8AF7_9A8D40296A6D__INCLUDED_)
#define AFX_STATDLG_H__C6CBFC75_6A42_4783_8AF7_9A8D40296A6D__INCLUDED_

#if _MSC_VER > 1000
#pragma once
#endif // _MSC_VER > 1000
// StatDlg.h : header file
//

/////////////////////////////////////////////////////////////////////////////
// CStatDlg dialog

class CStatDlg : public CDialog
{
// Construction
public:
    CStatDlg(CWnd* pParent = NULL);   // standard constructor

// Dialog Data
    //{{AFX_DATA(CStatDlg)
    enum { IDD = IDD_STAT_DLG };
        // NOTE: the ClassWizard will add data members here
    //}}AFX_DATA


// Overrides
    // ClassWizard generated virtual function overrides
    //{{AFX_VIRTUAL(CStatDlg)
    protected:
    virtual void DoDataExchange(CDataExchange* pDX);    // DDX/DDV support
    //}}AFX_VIRTUAL

// Implementation
protected:
```

```
    // Generated message map functions
    //{{AFX_MSG(CStatDlg)
        // NOTE: the ClassWizard will add member functions here
    //}}AFX_MSG
    DECLARE_MESSAGE_MAP()
};

//{{AFX_INSERT_LOCATION}}
// Microsoft Visual C++ will insert additional declarations immediately before the previous line.

#endif // !defined
(AFX_STATDLG_H__C6CBFC75_6A42_4783_8AF7_9A8D40296A6D__INCLUDED_)
-----------------------------------------------------------------------------------------------------------------------
-

// StatDlg.cpp : implementation file
//

#include "stdafx.h"
#include "ta1.h"
#include "StatDlg.h"

#ifdef _DEBUG
#define new DEBUG_NEW
#undef THIS_FILE
static char THIS_FILE[] = __FILE__;
#endif

/////////////////////////////////////////////////////////////////////////////
// CStatDlg dialog


CStatDlg::CStatDlg(CWnd* pParent /*=NULL*/)
    : CDialog(CStatDlg::IDD, pParent)
{
    //{{AFX_DATA_INIT(CStatDlg)
        // NOTE: the ClassWizard will add member initialization here
    //}}AFX_DATA_INIT
}

void CStatDlg::DoDataExchange(CDataExchange* pDX)
{
    CDialog::DoDataExchange(pDX);
    //{{AFX_DATA_MAP(CStatDlg)
        // NOTE: the ClassWizard will add DDX and DDV calls here
    //}}AFX_DATA_MAP
}

BEGIN_MESSAGE_MAP(CStatDlg, CDialog)
    //{{AFX_MSG_MAP(CStatDlg)
        // NOTE: the ClassWizard will add message map macros here
    //}}AFX_MSG_MAP
END_MESSAGE_MAP()
```

```
////////////////////////////////////////////////////////////////////////////
// CStatDlg message handlers
```