Laporan Tugas Kecil 3 IF2211 Strategi Algoritma
Semester 2 Tahun 2020/2021

Implementasi Algoritma A* untuk Menentukan Lintasan Terpendek

Disusun oleh :

Rahmah Khoirussyifa' Nurdini (13519013)

Clarisa Natalia Edelin (13519213)

SEKOLAH TEKNIK ELEKTRO DAN INFORMATIKA

INSTITUT TEKNOLOGI BANDUNG

2021

A. Source Code

File Simpul.cs

```
using System;

namespace Tucil3Stima
{
    public class Simpul
    {
        private string namaSimpul;
        private double coorX;
        private double coorY;

        public Simpul(string nama, double x, double y)
        {
            this.namaSimpul = nama;
            this.coorX = x;
            this.coorY = y;
        }

        public string getNama() { return namaSimpul; }

        public double getX() { return coorX; }

        public double getY() { return coorY; }

        /*Mendapatkan jarak lurus dari this ke S*/
        public double getStraightDistance(Simpul S)
        {
            return Math.Sqrt(Math.Pow(this.getX() - S.getX(), 2) +
Math.Pow(this.getY() - S.getY(), 2));
        }
    }
}
```

File Element.cs

```
using System;
using System.Collections.Generic;
using System.IO;

namespace Tucil3Stima
{
    public class Elemen
    {
        private Simpul simpul;
        private List<Simpul> path;
```

```
        public Elemen(Simpul s) // Membuat Elemen dengan path dirinya
sendiri saja
        {
            this.simpul = s;
            this.path = new List<Simpul>();
            this.path.Add(s);
        }

        public Elemen(List<Simpul> path, Simpul s) // Membuat Elemen
dengan path e ditambah path dirinya sendiri
        {
            this.simpul = s;
            this.path = new List<Simpul>();
            this.path.AddRange(path);
            this.path.Add(s);
        }

        /*Mendapatkan gedung dari Elemen*/
        public Simpul getSimpul() { return this.simpul; }

        /*Mendapatkan path dari Elemen*/
        public List<Simpul> getPath() { return this.path; }

        /*Mendapatkan jarak sejauh path*/
        public double getPathDistance()
        {
            // Inisialisasi dist yang akan menyimpan jarak sejauh path
            double dist = 0;

            // Lakukan perulangan untuk menghitung jarak pathnya
            for (int i = 0; i < path.Count - 1; i++)
            {
                dist += path[i].getStraightDistance(path[i + 1]);
            }

            return dist;
        }

        /*Menghitung f(n)*/
        public double countFn(Simpul tujuan)
        {
            /*Menghitung g(n) + h(n)*/
            return getPathDistance() +
simpul.getStraightDistance(tujuan);
        }

        /*Print out path*/
        public string getPathInString()
        {
```

```
            if (this.path == null)
            {
                return "Tidak ditemukan jalur";
            }
            else
            {
                string temp = "";
                for (int i = 0; i < path.Count; i++)
                {
                    temp += path[i].getNama();

                    if (i != path.Count - 1)
                    {
                        temp += "->";
                    }
                }
                return temp;
            }
        }
    }
}
```

File Graph.cs

```
using System;
using System.Collections.Generic;
using System.IO;

namespace Tucil3Stima
{
      public class Graph
      {
            private int nbSimpul;
            private List<Simpul> listSimpul;
            private bool[,] matAdj;

            public Graph(int n, List<Simpul> list, bool[,] mat)
            {
                  this.nbSimpul = n;
                  this.listSimpul = list;
                  this.matAdj = mat;
            }

            public int getNbElmt() { return this.nbSimpul; }

            public Simpul getSimpulAtIdx(int idx) { return
this.listSimpul[idx]; }
```

```
            public int getIdxOfSimpul(Simpul s) { return
listSimpul.IndexOf(s); }

            public List<Simpul> getAllAdjSimpul(Simpul s)
            {
                  List<Simpul> list = new List<Simpul>();

                  for (int i = 0; i < getNbElmt(); i++)
                  {
                        if (matAdj[getIdxOfSimpul(s), i])
                        {
                              if (!list.Contains(getSimpulAtIdx(i)))
                              {
                                    list.Add(getSimpulAtIdx(i));
                              }
                        }
                  }
                  return list;
            }

            /*Melakukan sort list dari f(n) terkecil sampai terbesar*/
            public void sortList(List<Elemen> list, Simpul tujuan)
            {
                  for (var i = 0; i < list.Count; i++)
                  {
                        var min = i;
                        for (var j = i + 1; j < list.Count; j++)
                        {
                              if (list[min].countFn(tujuan) >
list[j].countFn(tujuan))
                              {
                                    min = j;
                              }
                        }

                        if (i != min)
                        {
                              var valRendah = list[min];
                              list[min] = list[i];
                              list[i] = valRendah;
                        }
                  }
            }

            public Elemen getAStar(Simpul asal, Simpul tujuan,
List<Elemen> list)
            {
                  if (list.Count == 0) // Sudah tidak ada simpul yang
ingin dikunjungi
```

```
                  {
                        return null;
                  }
                  else if (list[0].getSimpul().Equals(tujuan)) // Sudah
ketemu jarak terpendek
                  {
                        return list[0];
                  }
                  else
                  {
                        // Mendapatkan semua simpul yang bertetangga
                        List<Simpul> listAdjSimpul =
getAllAdjSimpul(list[0].getSimpul());

                        for (int i = 0; i < listAdjSimpul.Count; i++)
                        {
                              List<Simpul> newPath = list[0].getPath();
                              Elemen E = new Elemen(newPath,
listAdjSimpul[i]);

                              // Insert simpul yang bertetangga ke list,
kemudian disort
                              list.Add(E);
                              sortList(list, tujuan);
                        }

                        list.Remove(list[0]);

                        // Jika balik lagi ke simpul asal
                        if (list[0].getSimpul().Equals(asal))
                {
                              list.Remove(list[0]);
                }

                        return getAStar(asal, tujuan, list);
                  }
            }
      }
}
```

File Form1.cs

```
using System;
using System.Collections.Generic;
using System.ComponentModel;
using System.Data;
using System.Drawing;
using System.Linq;
```

```
using System.Text;
using System.Threading.Tasks;
using System.Windows.Forms;

namespace Tucil3Stima
{
    public partial class Form1 : Form
    {
        private int nbSimpulForm;
        private List<Simpul> listSimpulForm;
        private bool[,] matAdjForm;
        private string asal;
        private string tujuan;

        public Form1()
        {
            InitializeComponent();
        }

        OpenFileDialog ofd = new OpenFileDialog();
        OpenFileDialog ofd2 = new OpenFileDialog();

        private void button1_Click(object sender, EventArgs e)
        {
            ofd.Filter = "Text Documents|*.txt";
            if (ofd.ShowDialog() == DialogResult.OK)
            {
                label1.Text = ofd.FileName;
                string readfile =
System.IO.File.ReadAllText(ofd.FileName);
                string[] cobasekiankali = readfile.Split(new[] { "\r\n",
"\r", "\n", "," }, StringSplitOptions.None);

                int banyakgedung = Int16.Parse(cobasekiankali[0]);
//insialisasi banyak gedung
                this.nbSimpulForm = banyakgedung;

                //bagian bikin list nama gedung dan list koordinat gedung
                List<string> listgedung = new List<string>();
                List <Tuple<double,double>> listkoordinatgedung = new
List<Tuple<double,double>> ();
                int isekarang = 0;

                for (int i = 1; i < banyakgedung*3.00; i++)
                {
                    listgedung.Add(cobasekiankali[i]);
                    double koordx = double.Parse(cobasekiankali[i = i +
1], System.Globalization.CultureInfo.InvariantCulture);
                    double koordy = double.Parse(cobasekiankali[i = i +
1], System.Globalization.CultureInfo.InvariantCulture);
```

```
                    listkoordinatgedung.Add(Tuple.Create(koordx,
koordy));
                    isekarang = i;
                }

                //list node yang terhubung, buat bikin grafnya
                int banyakelemenbacafile = cobasekiankali.Length;
                List<Tuple<string, string>> nodegraf = new
List<Tuple<string, string>>();

                for (int mulai = isekarang+1; mulai <
banyakelemenbacafile-1; mulai++)
                {
                    nodegraf.Add(Tuple.Create(cobasekiankali[mulai],
cobasekiankali[mulai = mulai + 1]));
                }

                //menampilkan graph
                Microsoft.Msagl.Drawing.Graph graph = new
Microsoft.Msagl.Drawing.Graph("graph");

                for (int qwe = 0; qwe < nodegraf.Count; qwe++)
                {
                    graph.AddEdge(nodegraf[qwe].Item1,
nodegraf[qwe].Item2);
                }

                foreach (var edge in graph.Edges)
                {
                    edge.Attr.ArrowheadAtTarget =
Microsoft.Msagl.Drawing.ArrowStyle.None;
                }

                List<string> distinctelement = new List<string>();
                int banyakelemenunik = 0;
                for (int i = isekarang+1; i < banyakelemenbacafile; i++)
                {
                    bool isDuplicate = false;
                    for (int j = isekarang + 1; j < i; j++)
                    {
                        if (cobasekiankali[i] == cobasekiankali[j])
                        {
                            isDuplicate = true;
                            break;
                        }
                    }

                    if (!isDuplicate)
                    {
                        Console.WriteLine(cobasekiankali[i]);
```

```
                        distinctelement.Add(cobasekiankali[i]);
                        banyakelemenunik++;

                    }
                }

                distinctelement.ToArray();

                for (int ter = 0; ter < banyakelemenunik; ter++)
                {
                    Microsoft.Msagl.Drawing.Node nodeehe =
graph.FindNode(distinctelement[ter]);
                    nodeehe.Attr.Shape =
Microsoft.Msagl.Drawing.Shape.Circle;
                }

                //tampilin graf di formnya
                gViewer1.Graph = graph;

                /*Membuat listSimpul dari Graph*/
                List<Simpul> listSimpul = new List<Simpul>();

                for (int i=0; i<banyakgedung; i++)
                {
                    Simpul tempSimpul = new Simpul(listgedung[i],
listkoordinatgedung[i].Item1, listkoordinatgedung[i].Item2);
                    listSimpul.Add(tempSimpul);
                }

                this.listSimpulForm = listSimpul;

                //input pilihan ke combo box
                for(int combos = 0; combos<banyakelemenunik; combos++)
                {
                    comboBox1.Items.Add(distinctelement[combos]);
                    comboBox2.Items.Add(distinctelement[combos]);
                }
            }
        }

        private void button2_Click(object sender, EventArgs e)
        {
            ofd2.Filter = "Text Documents|*.txt";

            if (ofd2.ShowDialog() == DialogResult.OK)
            {

                label7.Text = ofd2.FileName;
                string readfile2 =
System.IO.File.ReadAllText(ofd2.FileName);
```

```
                string[] cobasekiankali2 = readfile2.Split(new[] {
"\r\n", "\r", "\n", " "}, StringSplitOptions.None);

                double ukuranmatriksdalamdouble =
Math.Sqrt(cobasekiankali2.Length);
                string ukuranmatriksstring =
ukuranmatriksdalamdouble.ToString();
                int ukuranmatriksint = Int16.Parse(ukuranmatriksstring);

                //membuat matriks boolean
                bool[,] matriksketerhubungan = new
bool[ukuranmatriksint,ukuranmatriksint];

                int matriksbacafile = 0;
                for(int r = 0; r<ukuranmatriksint; r++)
                {
                    for(int co = 0; co<ukuranmatriksint; co++)
                    {
                        if(cobasekiankali2[matriksbacafile] == "1")
                        {
                            matriksketerhubungan[r, (matriksbacafile %
ukuranmatriksint)] = true;
                        }

                        matriksbacafile++;
                    }
                }

                this.matAdjForm = matriksketerhubungan;

                string tester = matriksketerhubungan[0,1].ToString();
            }
        }

        private void button3_Click(object sender, EventArgs e)
        {
            this.asal = comboBox1.Text;
            this.tujuan = comboBox2.Text;

            /*Mencari simpul dengan nama asal*/
            Simpul simpulAsal = new Simpul("X", 0, 0);
            for (int i = 0; i < this.nbSimpulForm; i++)
            {
                if (this.listSimpulForm[i].getNama().Equals(asal))
                {
                    simpulAsal = listSimpulForm[i];
                    break;
                }
            }
```

```
            /*Mencari simpul dengan nama tujuan*/
            Simpul simpulTujuan = new Simpul("X",0,0);
            for (int i=0; i<this.nbSimpulForm; i++)
            {
                if (this.listSimpulForm[i].getNama().Equals(tujuan))
                {
                    simpulTujuan = listSimpulForm[i];
                    break;
                }
            }

            /*Membuat objek Graph*/
            Graph G = new Graph(this.nbSimpulForm, this.listSimpulForm,
this.matAdjForm);

            List<Elemen> listElemen = new List<Elemen>();
            listElemen.Add(new Elemen(simpulAsal));

            /*Mendapatkan jalur terpendek dengan algortima A star*/
            Elemen E = G.getAStar(simpulAsal, simpulTujuan, listElemen);

            if (E == null)
            {
                label8.Text = "Tidak ada jalur yang tersedia";
            }
            else
            {
                label8.Text = "Jarak yang ditempuh: " +
E.getPathDistance().ToString() + " km";
            }


            string readfile = System.IO.File.ReadAllText(ofd.FileName);
            string[] cobasekiankali = readfile.Split(new[] { "\r\n",
"\r", "\n", "," }, StringSplitOptions.None);

            int banyakgedung = Int16.Parse(cobasekiankali[0]);
//insialisasi banyak gedung
            this.nbSimpulForm = banyakgedung;

            //bagian bikin list nama gedung dan list koordinat gedung
            List<string> listgedung = new List<string>();
            List<Tuple<double, double>> listkoordinatgedung = new
List<Tuple<double, double>>();
            int isekarang = 0;

            for (int i = 1; i < banyakgedung * 3.00; i++)
            {
                listgedung.Add(cobasekiankali[i]);
                double koordx = double.Parse(cobasekiankali[i = i + 1],
```

```
System.Globalization.CultureInfo.InvariantCulture);
                double koordy = double.Parse(cobasekiankali[i = i + 1],
System.Globalization.CultureInfo.InvariantCulture);
                listkoordinatgedung.Add(Tuple.Create(koordx, koordy));
                isekarang = i;
            }

            //list node yang terhubung, buat bikin grafnya
            int banyakelemenbacafile = cobasekiankali.Length;
            List<Tuple<string, string>> nodegraf = new List<Tuple<string,
string>>();

            for (int mulai = isekarang + 1; mulai < banyakelemenbacafile
- 1; mulai++)
            {
                nodegraf.Add(Tuple.Create(cobasekiankali[mulai],
cobasekiankali[mulai = mulai + 1]));
            }

            //membuat graf
            Microsoft.Msagl.Drawing.Graph graph = new
Microsoft.Msagl.Drawing.Graph("graph");

            for (int qwe = 0; qwe < nodegraf.Count; qwe++)
            {
                var edge = graph.AddEdge(nodegraf[qwe].Item1,
nodegraf[qwe].Item2);

                if (E != null)
                {
                    for (int i = 0; i < E.getPath().Count - 1; i++)
                    {
                        bool conditional1 =
(nodegraf[qwe].Item1.Equals(E.getPath()[i].getNama()) &&
nodegraf[qwe].Item2.Equals(E.getPath()[i + 1].getNama()));
                        bool conditional2 =
(nodegraf[qwe].Item2.Equals(E.getPath()[i].getNama()) &&
nodegraf[qwe].Item1.Equals(E.getPath()[i + 1].getNama()));
                        if (conditional1 || conditional2)
                        {
                            edge.Attr.Color =
Microsoft.Msagl.Drawing.Color.ForestGreen;
                        }
                    }
                }
            }

            foreach (var edge in graph.Edges)
            {
                edge.Attr.ArrowheadAtTarget =
```

```
Microsoft.Msagl.Drawing.ArrowStyle.None;
            }

            List<string> distinctelement = new List<string>();
            int banyakelemenunik = 0;
            for (int i = isekarang + 1; i < banyakelemenbacafile; i++)
            {
                bool isDuplicate = false;
                for (int j = isekarang + 1; j < i; j++)
                {
                    if (cobasekiankali[i] == cobasekiankali[j])
                    {
                        isDuplicate = true;
                        break;
                    }
                }

                if (!isDuplicate)
                {
                    Console.WriteLine(cobasekiankali[i]);
                    distinctelement.Add(cobasekiankali[i]);
                    banyakelemenunik++;

                }
            }

            distinctelement.ToArray();

            for (int ter = 0; ter < banyakelemenunik; ter++)
            {
                Microsoft.Msagl.Drawing.Node nodeehe =
graph.FindNode(distinctelement[ter]);
                nodeehe.Attr.Shape =
Microsoft.Msagl.Drawing.Shape.Circle;
            }

            //tampilin graf di formnya
            gViewer1.Graph = graph;

            if (E != null)
            {
                for (int k = 0; k < this.nbSimpulForm; k++)
                {
                    for (int j = 0; j < E.getPath().Count; j++)
                    {
                        if
(listSimpulForm[k].getNama().Equals(E.getPath()[j].getNama()))
                        {

graph.FindNode(listSimpulForm[k].getNama()).Attr.FillColor =
```

```
Microsoft.Msagl.Drawing.Color.LimeGreen;
                        }
                    }
                }
            }
        }

        private void label8_Click(object sender, EventArgs e)
        {

        }
    }
}
```

B. Testing

1. Test case 1 - ITB

File ITB File 1.txt

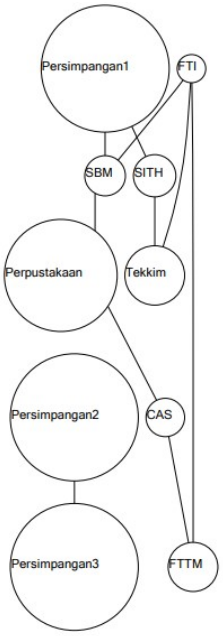
```
10
Persimpangan1, -6.887987512115787, 107.6082750834938
SBM, -6.888000381490534, 107.60926311704202
Perpustakaan, -6.887935932461746, 107.61074121055448
CAS, -6.88786535169282, 107.61138020568835
Persimpangan2, -6.88820167037999, 107.61153778648033
FTTM, -6.887935932461746, 107.61074121055448
Persimpangan3, -6.888748083808535, 107.61157870310456
FTI, -6.888633516174999, 107.60917225312909
Tekkim, -6.88911599416145, 107.60905096216837
SITH, -6.889503879360927, 107.60903684025672
Persimpangan1,SBM
SBM,Perpustakaan
Perpustakaan,CAS
CAS,FTTM
FTI,SBM
FTI,Tekkim
SITH,Tekkim
FTI,FTTM
Persimpangan2,Persimpangan3
Persimpangan1,SITH
```

File ITB Matriks.txt

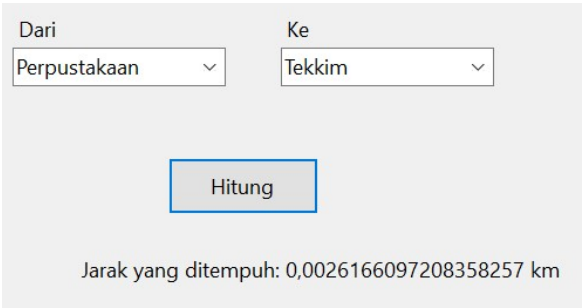
```
0 1 0 0 0 0 0 0 0 1
1 0 1 0 0 0 0 1 0 0
0 1 0 1 0 0 0 0 0 0
```

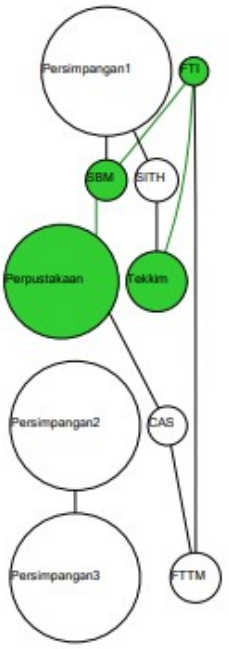
```
0 0 1 0 0 1 0 0 0 0
0 0 0 0 0 0 1 0 0 0
0 0 0 1 0 0 0 1 0 0
0 0 0 0 1 0 0 0 0 0
0 1 0 0 0 1 0 0 1 0
0 0 0 0 0 0 0 1 0 0
1 0 0 0 0 0 0 0 0 0
```

Graph yang dihasilkan adalah sebagai berikut.

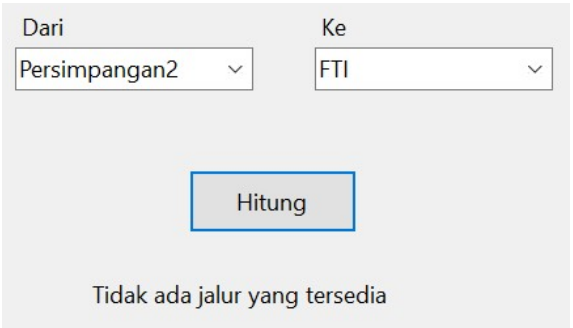


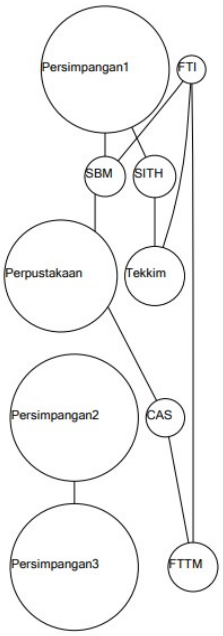
Ketika mencari jarak terpendek dari Perpustakaan ke Tekkim, akan ditampilkan visualisasi sebagai berikut, dengan jarak terpendek adalah 0,0026 km.





Ketika mencari jarak terpendek dari Persimpangan2 ke FTI, akan ditampilkan visualisasi sebagai berikut, dengan tidak ada jalur yang tersedia karena FTI dan Persimpangan2 berada pada subgraf yang berbeda.





2. Test case 2 - Alun-alun

File AlunAlun File 1.txt

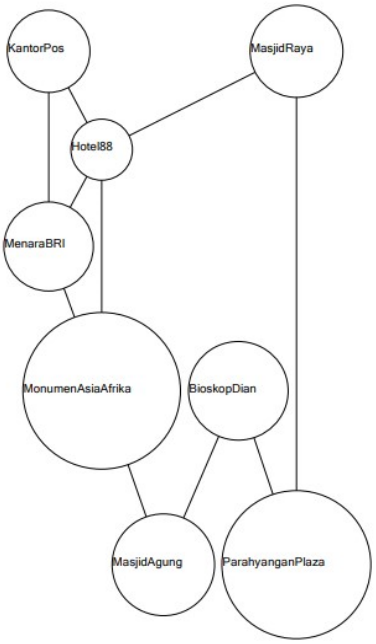
```
8
MasjidRaya, -6.92165984786416, 107.60622135616822
Hotel88, -6.921100859637293, 107.60622976052328
MonumenAsiaAfrika, -6.9212802365285, 107.60773834225725
MasjidAgung, -6.921417897817364, 107.60762068128635
BioskopDian, -6.9225150151389805, 107.6074820094278
ParahyanganPlaza, -6.922360667989351, 107.60627178229859
MenaraBRI, -6.921027250329429, 107.60732708692336
KantorPos, -6.920777990887927, 107.60618170826935
KantorPos,Hotel88
Hotel88,MenaraBRI
Hotel88,MonumenAsiaAfrika
MenaraBRI,MonumenAsiaAfrika
MonumenAsiaAfrika,MasjidAgung
BioskopDian,MasjidAgung
BioskopDian,ParahyanganPlaza
MasjidRaya,ParahyanganPlaza
MasjidRaya,Hotel88
KantorPos,MenaraBRI
```

File AlunAlun Matriks.txt

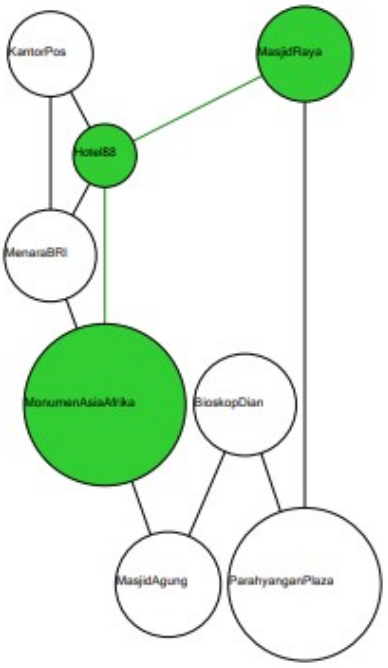
```
0 1 0 0 0 1 0 0
1 0 1 0 0 0 1 1
```

```
0 1 0 1 0 0 1 0
0 0 1 0 1 0 0 0
0 0 0 1 0 1 0 0
1 0 0 0 1 0 0 0
0 1 1 0 0 0 0 1
0 1 0 0 0 0 1 0
```

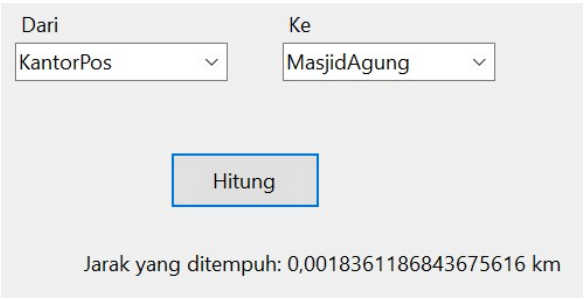
Graph yang dihasilkan adalah sebagai berikut.

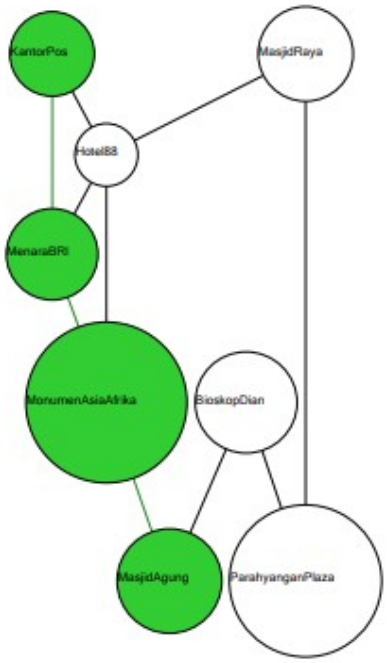


Ketika mencari jarak terpendek dari MonumenAsiaAfrika ke MasjidRaya, akan ditampilkan visualisasi sebagai berikut, dengan jarak terpendek adalah 0,002078 km.



Ketika mencari jarak terpendek dari KantorPos ke MasjidAgung, akan ditampilkan visualisasi sebagai berikut, dengan jarak terpendek adalah 0.0018 km.





3. **Test case 3 - Buah Batu**

File BuahBatu File 1.txt

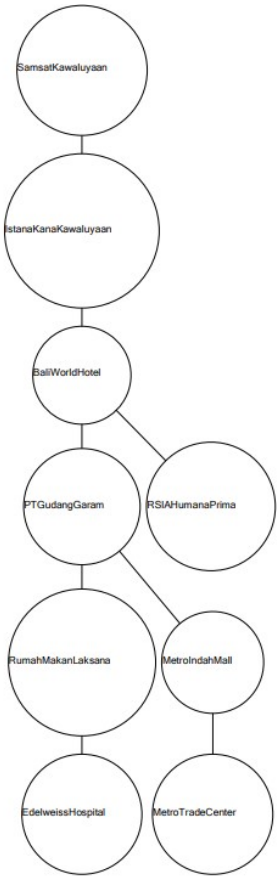
```
9
SamsatKawaluyaan, -6.9331, 107.66292
IstanaKanaKawaluyaan, -6.93644, 107.66303
PTGudangGaram, -6.93862, 107.66058
MetroIndahMall, -6.94179, 107.65875
EdelweissHospital, -6.94331, 107.64964
MetroTradeCenter, -6.94292, 107.65895
RumahMakanLaksana, -6.94177, 107.65170
BaliWorldHotel, -6.93806, 107.66319
RSIAHumanaPrima, -6.94052, 107.66401
SamsatKawaluyaan,IstanaKanaKawaluyaan
IstanaKanaKawaluyaan,BaliWorldHotel
BaliWorldHotel,RSIAHumanaPrima
BaliWorldHotel,PTGudangGaram
PTGudangGaram,MetroIndahMall
PTGudangGaram,RumahMakanLaksana
MetroIndahMall,MetroTradeCenter
RumahMakanLaksana,EdelweissHospital
```

File AlunAlun Matriks.txt

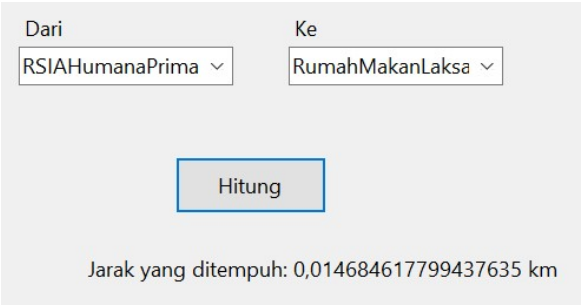
```
0 1 0 0 0 0 0 0 0
1 0 0 0 0 0 0 1 0
0 0 0 1 0 0 1 1 0
```

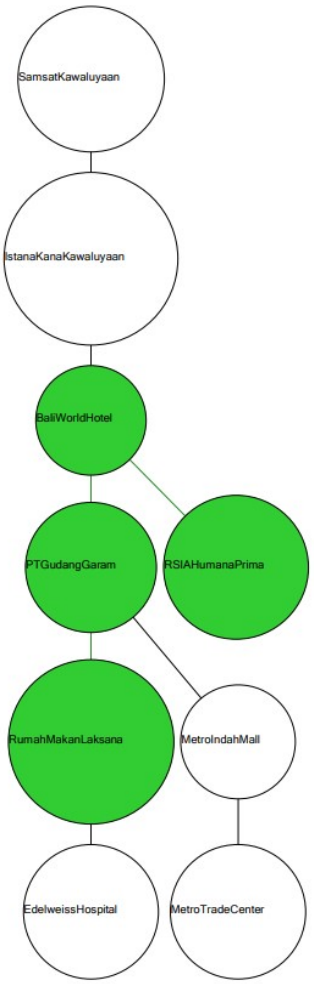
```
0 0 1 0 0 1 0 0 0
0 0 0 0 0 0 1 0 0
0 0 0 1 0 0 0 0 0
0 0 1 0 1 0 0 0 0
0 1 1 0 0 0 0 0 1
0 0 0 0 0 0 0 1 0
```

Graph yang dihasilkan adalah sebagai berikut.

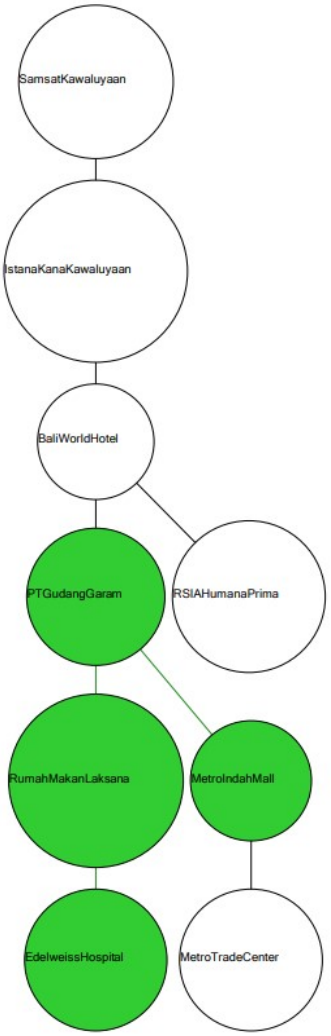


Ketika mencari jarak terpendek dari RSIAHumanaPrima ke RumahMakanLaksana, akan ditampilkan visualisasi sebagai berikut, dengan jarak terpendek adalah 0,01468 km.





Ketika mencari jarak terpendek dari EdelweissHospital ke MetroIndahMall, akan ditampilkan visualisasi sebagai berikut, dengan jarak terpendek adalah 0,0156 km.



4. **Test case 4 - Kota Tua**

   File KotaTua File 1.txt

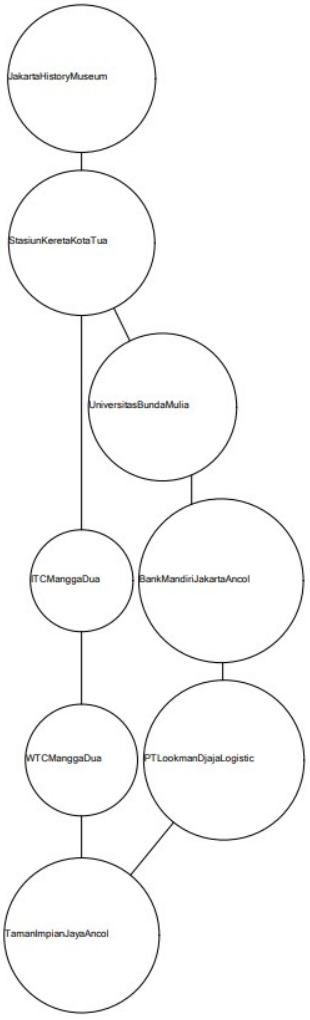
```
8
JakartaHistoryMuseum, -6.13490, 106.81327
StasiunKeretaKotaTua, -6.13398, 106.81863
UniversitasBundaMulia, -6.1299, 106.81841
ITCManggaDua, -6.13540, 106.82462
WTCManggaDua, -6.13439, 106.83097
BankMandiriJakartaAncol, -6.12839, 106.82233
PTLookmanDjajaLogistic, -6.12820, 106.82753
TamanImpianJayaAncol, -6.12896, 106.83338
JakartaHistoryMuseum,StasiunKeretaKotaTua
StasiunKeretaKotaTua,UniversitasBundaMulia
StasiunKeretaKotaTua,ITCManggaDua
ITCManggaDua,WTCManggaDua
WTCManggaDua,TamanImpianJayaAncol
UniversitasBundaMulia,BankMandiriJakartaAncol
```

```
BankMandiriJakartaAncol,PTLookmanDjajaLogistic
PTLookmanDjajaLogistic,TamanImpianJayaAncol
```

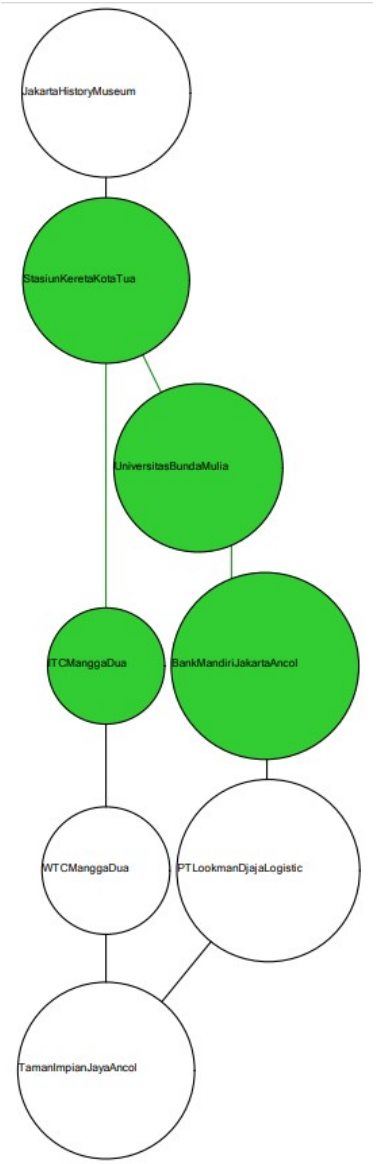
File KotaTua Matriks.txt

```
0 1 0 0 0 0 0 0
1 0 1 1 0 0 0 0
0 1 0 0 0 1 0 0
0 1 0 0 1 0 0 0
0 0 0 1 0 0 0 1
0 0 1 0 0 0 1 0
0 0 0 0 0 1 0 1
0 0 0 0 1 0 1 0
```

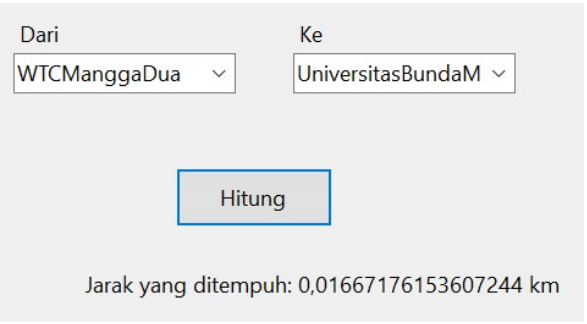
Graph yang dihasilkan adalah sebagai berikut.

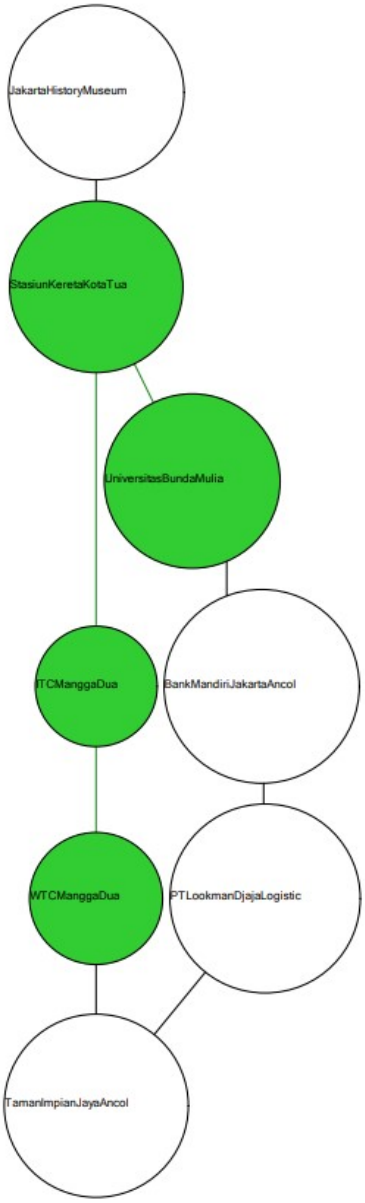


Ketika mencari jarak terpendek dari BankMandiriJakarta ke ITCManggaDua, akan ditampilkan visualisasi sebagai berikut, dengan jarak terpendek adalah 0,0144 km.



Ketika mencari jarak terpendek dari WTCManggaDua ke UniversitasBundaMulia, akan ditampilkan visualisasi sebagai berikut, dengan jarak terpendek adalah 0,0166 km.





5. **Test case 5**

   File test case 5 File 1.txt

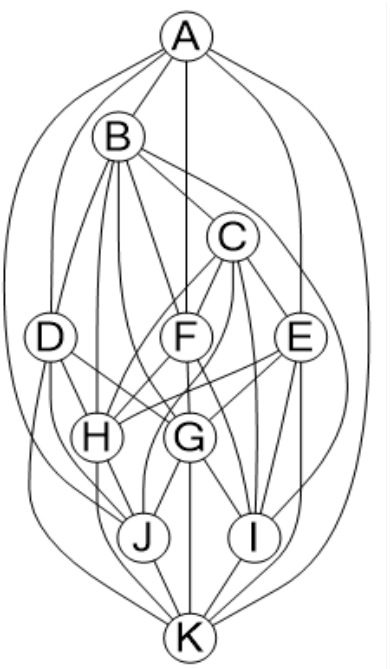
11

```
A, -5.20345, 10.29483
B, -4.30495, 19.23094
C, -5.12984, 16.12394
D, -6.10293, 18.12934
E, -4.10293, 17.19283
F, -5.19283, 16.19203
G, -7.19283, 14.12376
H, -5.19863, 17.09812
I, -6.12399, 18.12938
J, -6.19234, 19.12399
K, -7.19238, 17.19283
A,B
A,D
A,E
A,F
A,J
A,K
B,C
B,D
B,F
B,G
B,H
B,I
C,E
C,F
C,H
C,I
C,J
D,G
D,H
D,J
D,K
E,G
E,H
E,I
E,K
F,G
F,H
F,I
G,I
G,J
G,K
H,J
H,K
I,K
J,K
```

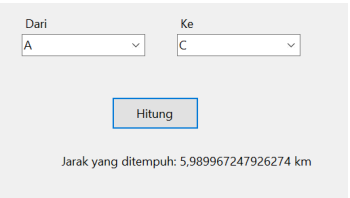
File test case 5 Matriks.txt

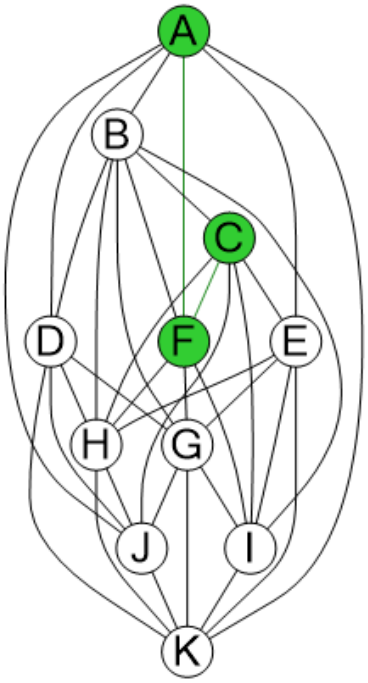
```
0 1 0 1 1 1 0 0 0 1 1
1 0 1 1 0 0 1 1 1 0 0
0 1 0 0 1 1 0 1 1 1 0
1 1 0 0 0 0 1 1 0 1 1
1 0 1 0 0 0 1 1 1 0 1
1 0 1 0 0 0 1 1 1 0 0
0 1 0 1 1 1 0 0 1 1 1
0 1 1 1 1 1 0 0 0 1 1
0 1 1 0 1 1 1 0 0 0 1
1 0 1 1 0 0 1 1 0 0 1
1 0 0 1 1 0 1 1 1 1 0
```

Graph yang dihasilkan adalah sebagai berikut.

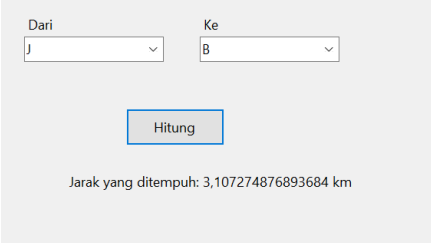


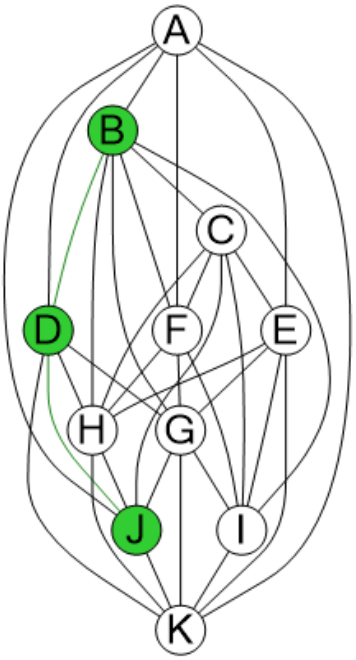
Ketika mencari jarak terpendek dari A ke C , akan ditampilkan visualisasi sebagai berikut, dengan jarak terpendek adalah 5,989967247926274 km.





Ketika mencari jarak terpendek dari J ke B , akan ditampilkan visualisasi sebagai berikut, dengan jarak terpendek adalah 3,107274876893684 km.





6. Test case 6

File test case 6 File 1.txt

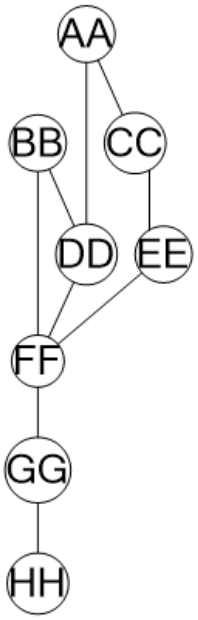
```
8
AA, -5.35723, 10.23422
BB, -6.23475, 15.23468
CC, -7.52290, 12.23869
DD, -6.11446, 13.24676
EE, -9.14432, 11.23754
FF, -6.14677, 12.11299
GG, -6.89070, 11.23795
HH, -7.34596, 16.12768
AA,CC
AA,DD
BB,DD
BB,FF
CC,EE
DD,FF
EE,FF
FF,GG
GG,HH
```

File test case 6 Matriks.txt

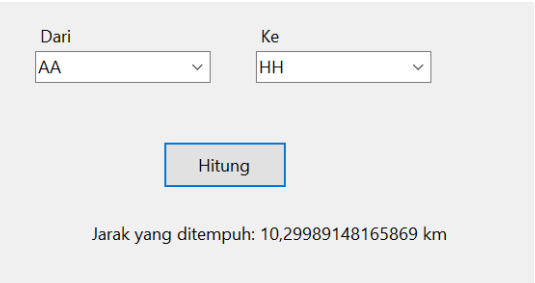
```
0 0 1 1 0 0 0 0
0 0 0 1 0 1 0 0
1 0 0 0 1 0 0 0
1 1 0 0 0 1 0 0
```

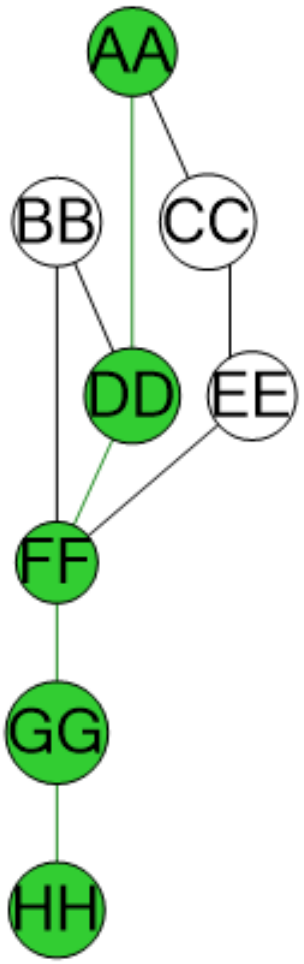
```
0 0 1 0 0 1 0 0
0 1 0 1 1 0 1 0
0 0 0 0 0 1 0 1
0 0 0 0 0 0 1 0
```

Graph yang dihasilkan adalah sebagai berikut.

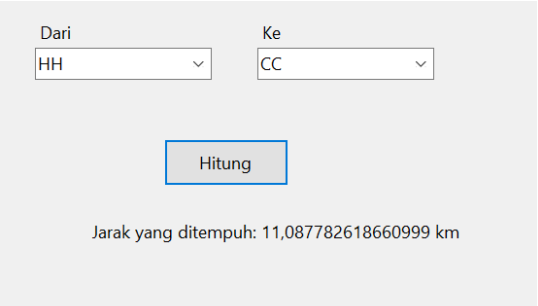


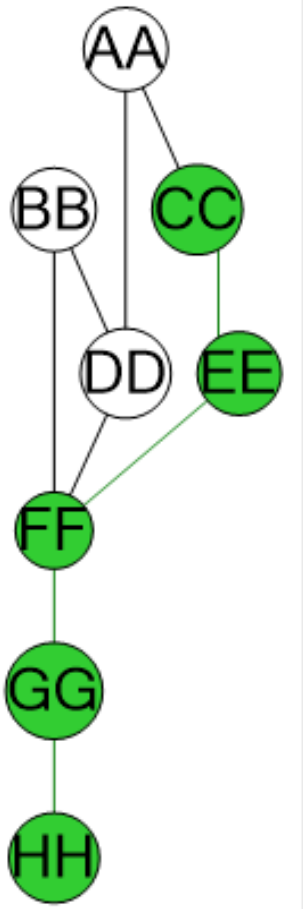
Ketika mencari jarak terpendek dari AA ke HH , akan ditampilkan visualisasi sebagai berikut, dengan jarak terpendek adalah 10,29989148165869 km.





Ketika mencari jarak terpendek dari HH ke CC , akan ditampilkan visualisasi sebagai berikut, dengan jarak terpendek adalah 11,087782618660999 km.





C. Alamat Drive Tugas

File-file deliverable tugas ini dapat diakses melalui: https://github.com/ClarisaNatalia/Tucil3-Stima

D. Checklist

1	Program dapat menerima input graf	√
2	Program dapat menghitung lintasan terpendek	√
3	Program dapat menampilkan lintasan terpendek serta jaraknya	√
4	Bonus: Program dapat menerima input peta dengan Google Map API dan menampilkan peta	