LAPORAN TUGAS KECIL 3

IF2211 – STRATEGI ALGORITMA

Penyelesaian Pencarian Rute dengan Algoritma A*

Oleh:

Alifah Rahmatika Basyasya 13519053

PROGRAM STUDI TEKNIK INFORMATIKA

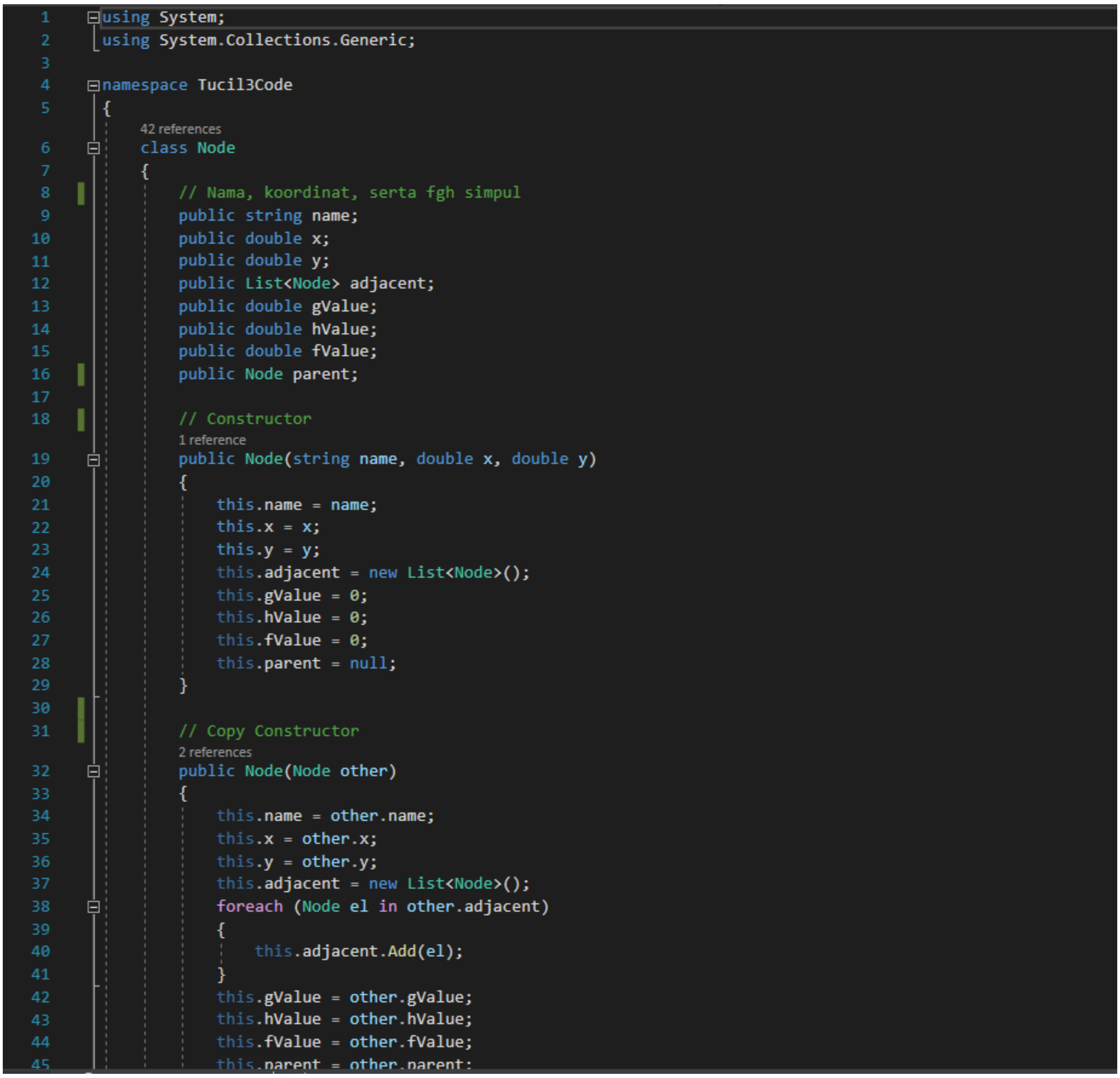
SEKOLAH TEKNIK ELEKTRO DAN INFORMATIKA

INSTITUT TEKNOLOGI BANDUNG

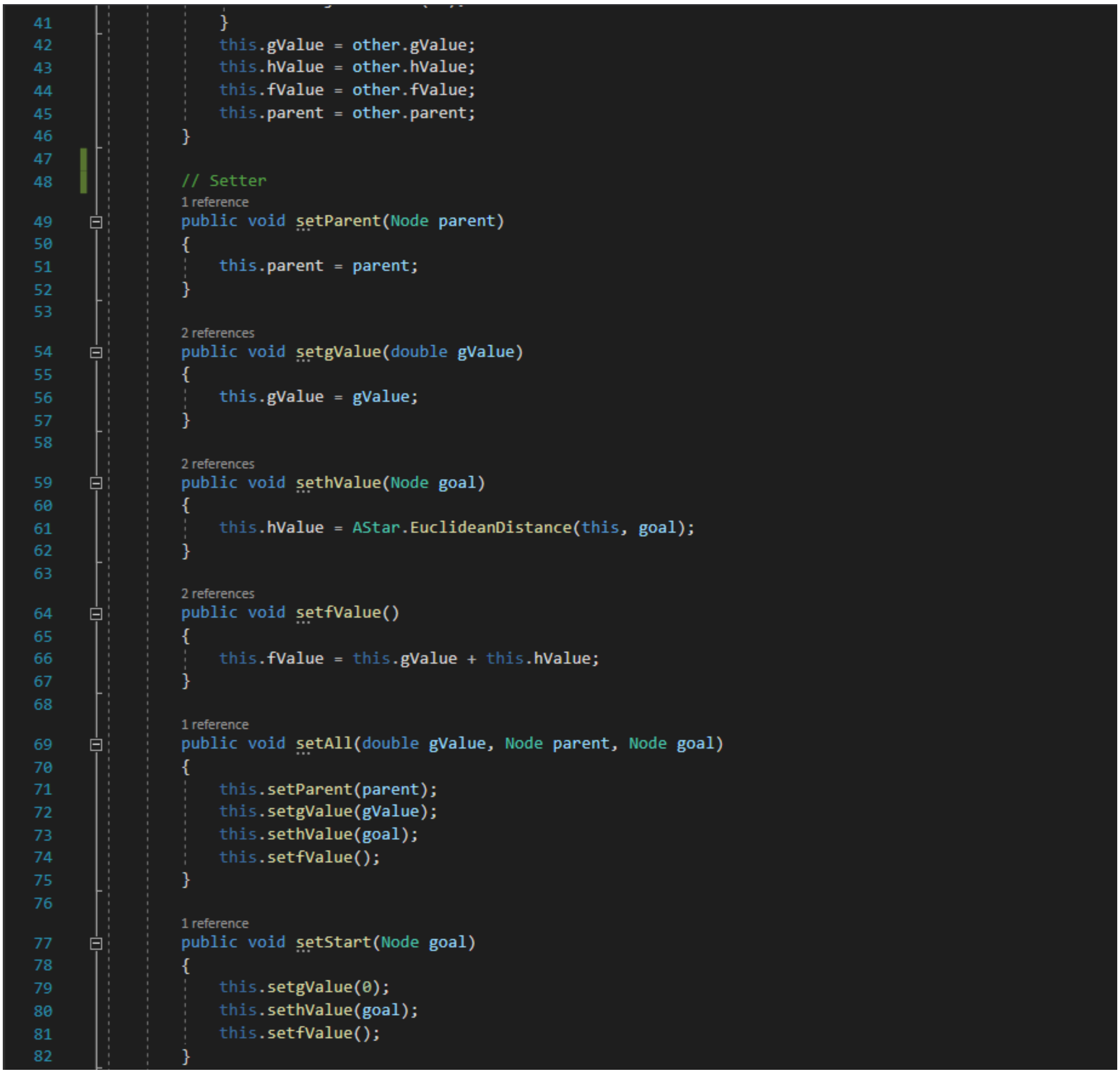
2021

Kode Program

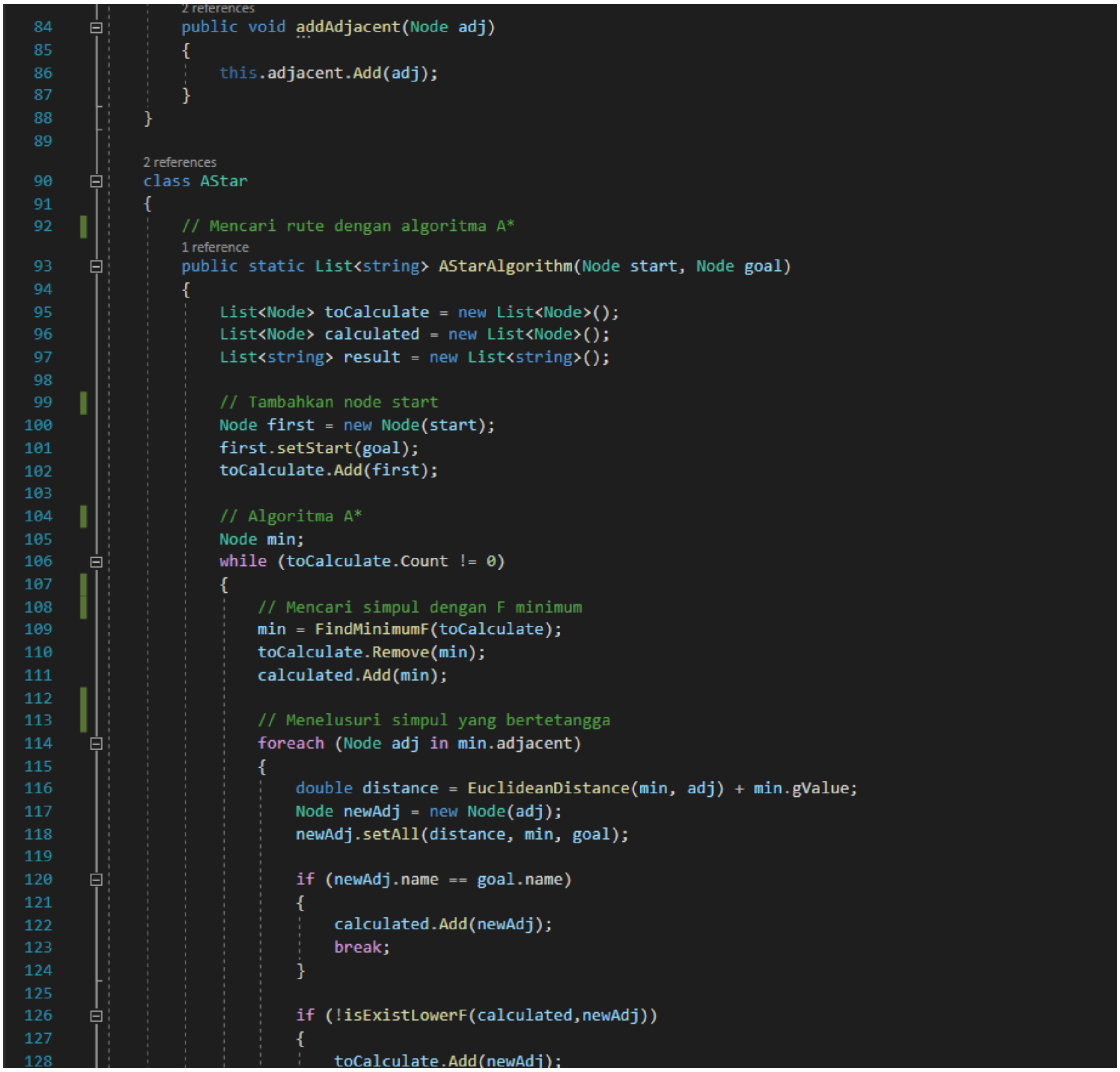
Algoritma



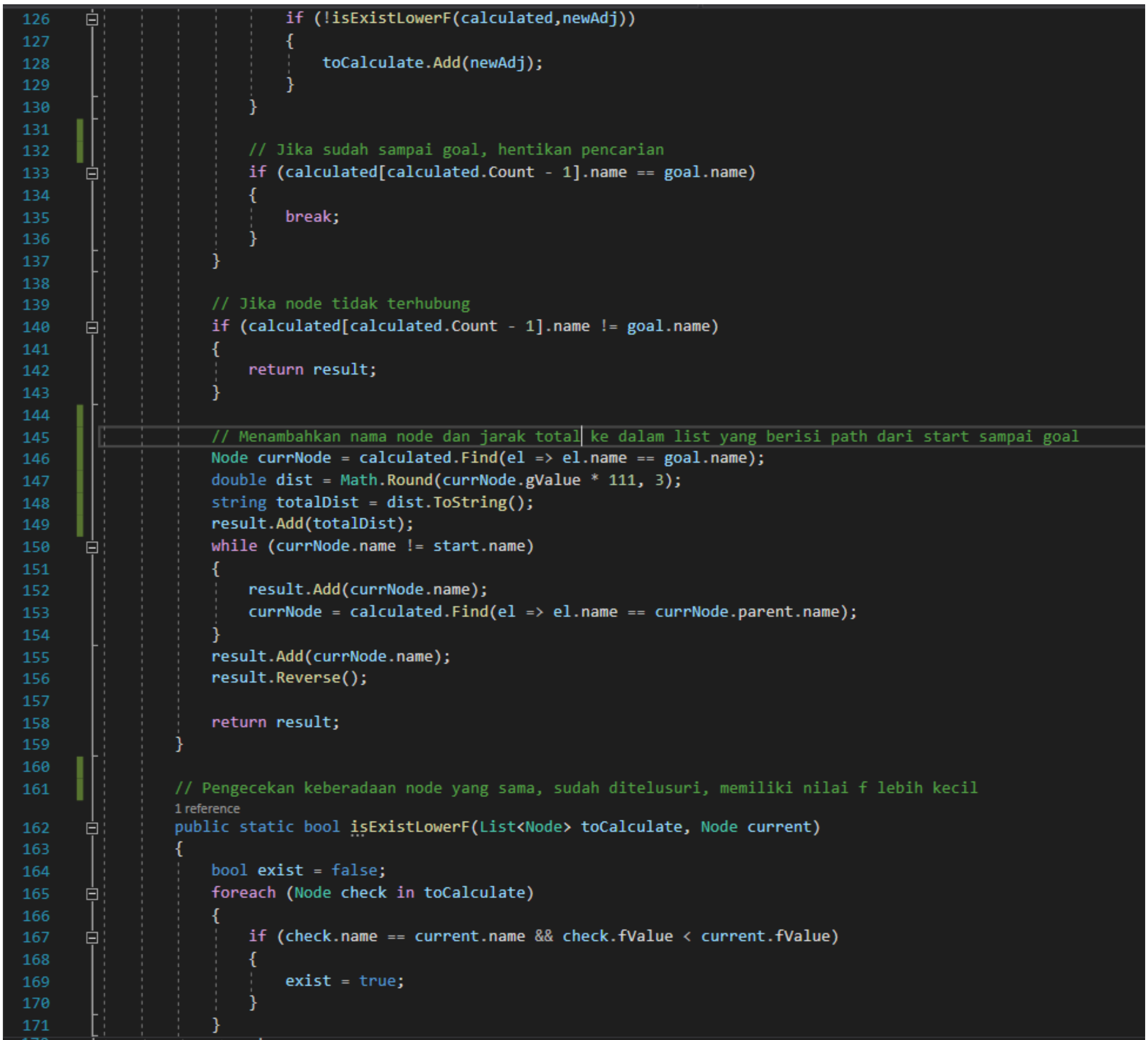
```
using System;
using System.Collections.Generic;

namespace Tucil3Code
{
    42 references
    class Node
    {
        // Nama, koordinat, serta fgh simpul
        public string name;
        public double x;
        public double y;
        public List<Node> adjacent;
        public double gValue;
        public double hValue;
        public double fValue;
        public Node parent;

        // Constructor
        1 reference
        public Node(string name, double x, double y)
        {
            this.name = name;
            this.x = x;
            this.y = y;
            this.adjacent = new List<Node>();
            this.gValue = 0;
            this.hValue = 0;
            this.fValue = 0;
            this.parent = null;
        }

        // Copy Constructor
        2 references
        public Node(Node other)
        {
            this.name = other.name;
            this.x = other.x;
            this.y = other.y;
            this.adjacent = new List<Node>();
            foreach (Node el in other.adjacent)
            {
                this.adjacent.Add(el);
            }
            this.gValue = other.gValue;
            this.hValue = other.hValue;
            this.fValue = other.fValue;
            this.parent = other.parent;
```



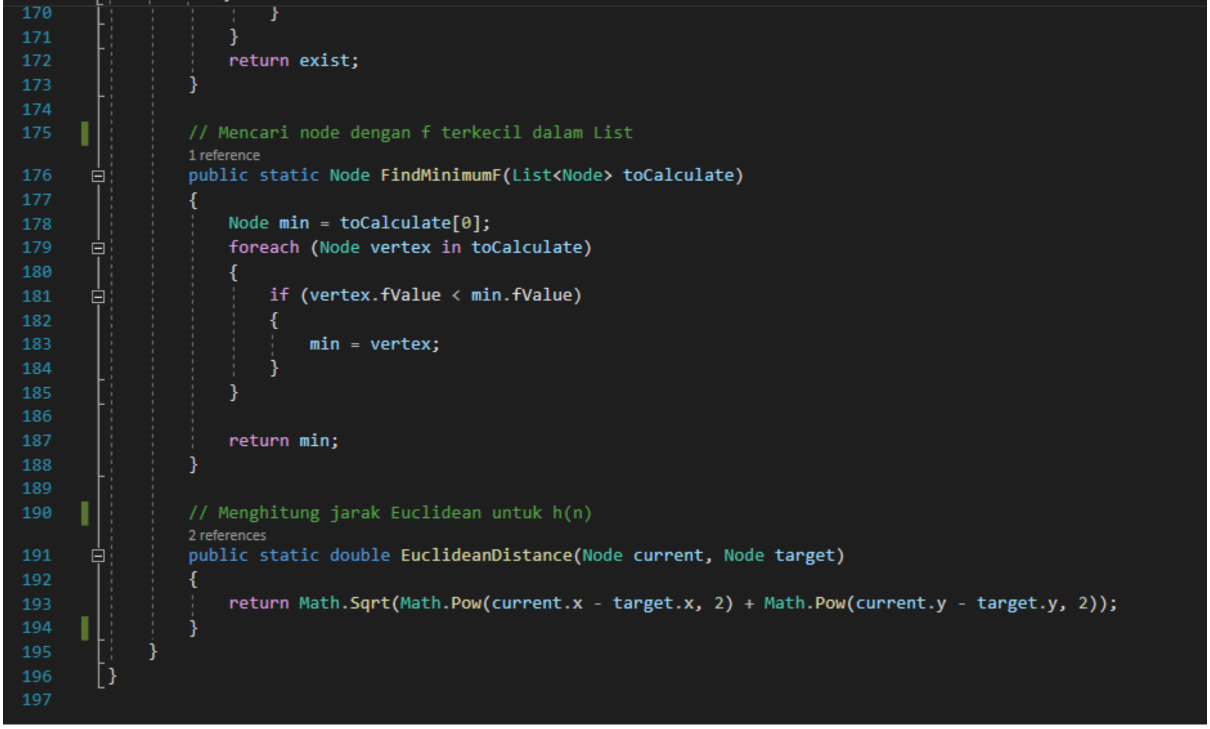
```
            }
            this.gValue = other.gValue;
            this.hValue = other.hValue;
            this.fValue = other.fValue;
            this.parent = other.parent;
        }

        // Setter
        1 reference
        public void setParent(Node parent)
        {
            this.parent = parent;
        }

        2 references
        public void setgValue(double gValue)
        {
            this.gValue = gValue;
        }

        2 references
        public void sethValue(Node goal)
        {
            this.hValue = AStar.EuclideanDistance(this, goal);
        }

        2 references
        public void setfValue()
        {
            this.fValue = this.gValue + this.hValue;
        }

        1 reference
        public void setAll(double gValue, Node parent, Node goal)
        {
            this.setParent(parent);
            this.setgValue(gValue);
            this.sethValue(goal);
            this.setfValue();
        }

        1 reference
        public void setStart(Node goal)
        {
            this.setgValue(0);
            this.sethValue(goal);
            this.setfValue();
        }
```



```
        2 references
        public void addAdjacent(Node adj)
        {
            this.adjacent.Add(adj);
        }
    }

    2 references
    class AStar
    {
        // Mencari rute dengan algoritma A*
        1 reference
        public static List<string> AStarAlgorithm(Node start, Node goal)
        {
            List<Node> toCalculate = new List<Node>();
            List<Node> calculated = new List<Node>();
            List<string> result = new List<string>();

            // Tambahkan node start
            Node first = new Node(start);
            first.setStart(goal);
            toCalculate.Add(first);

            // Algoritma A*
            Node min;
            while (toCalculate.Count != 0)
            {
                // Mencari simpul dengan F minimum
                min = FindMinimumF(toCalculate);
                toCalculate.Remove(min);
                calculated.Add(min);

                // Menelusuri simpul yang bertetangga
                foreach (Node adj in min.adjacent)
                {
                    double distance = EuclideanDistance(min, adj) + min.gValue;
                    Node newAdj = new Node(adj);
                    newAdj.setAll(distance, min, goal);

                    if (newAdj.name == goal.name)
                    {
                        calculated.Add(newAdj);
                        break;
                    }

                    if (!isExistLowerF(calculated,newAdj))
                    {
                        toCalculate.Add(newAdj);
```

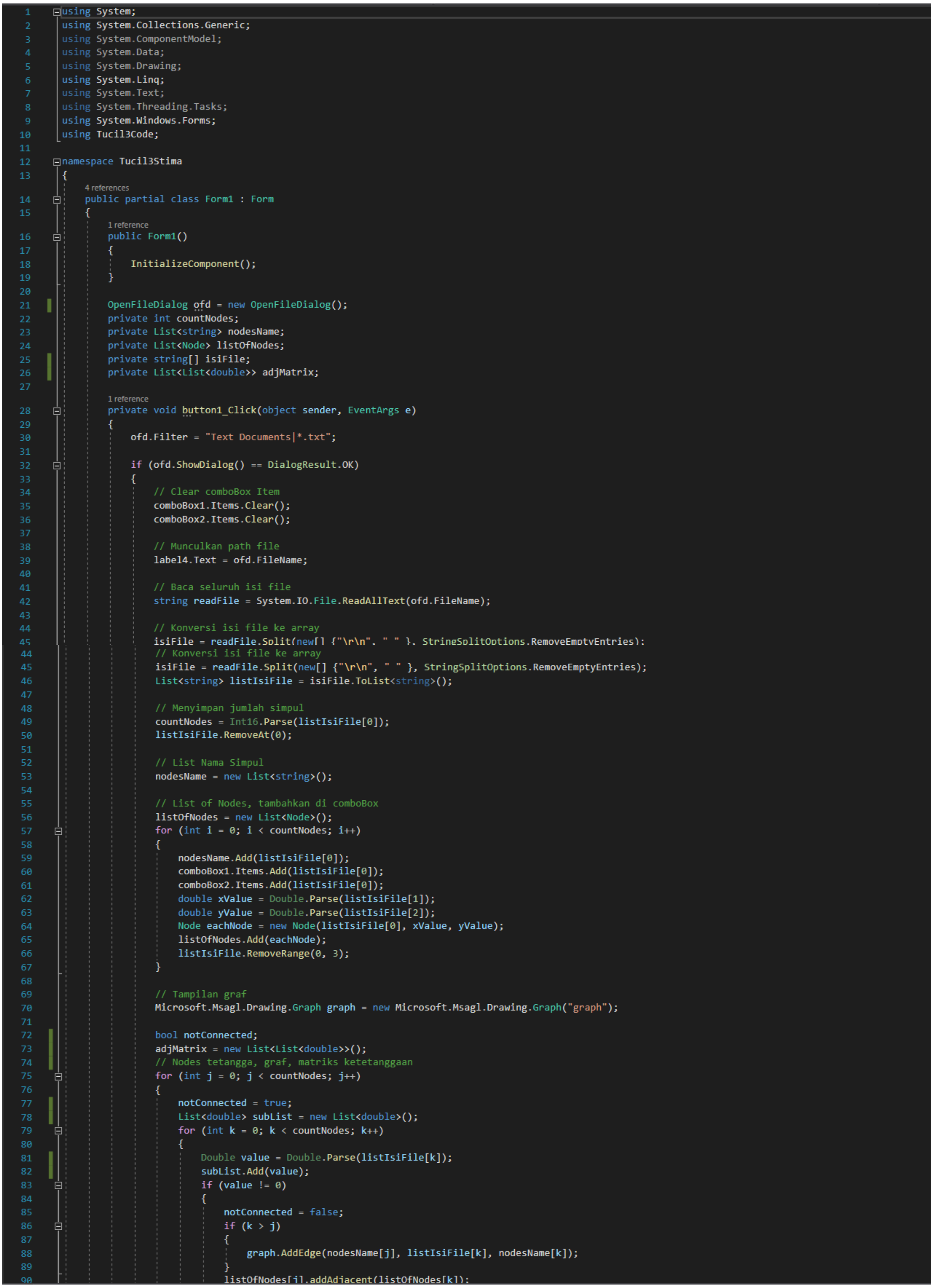


```
                    if (!isExistLowerF(calculated,newAdj))
                    {
                        toCalculate.Add(newAdj);
                    }
                }

                // Jika sudah sampai goal, hentikan pencarian
                if (calculated[calculated.Count - 1].name == goal.name)
                {
                    break;
                }
            }

            // Jika node tidak terhubung
            if (calculated[calculated.Count - 1].name != goal.name)
            {
                return result;
            }

            // Menambahkan nama node dan jarak total ke dalam list yang berisi path dari start sampai goal
            Node currNode = calculated.Find(el => el.name == goal.name);
            double dist = Math.Round(currNode.gValue * 111, 3);
            string totalDist = dist.ToString();
            result.Add(totalDist);
            while (currNode.name != start.name)
            {
                result.Add(currNode.name);
                currNode = calculated.Find(el => el.name == currNode.parent.name);
            }
            result.Add(currNode.name);
            result.Reverse();

            return result;
        }

        // Pengecekan keberadaan node yang sama, sudah ditelusuri, memiliki nilai f lebih kecil
        1 reference
        public static bool isExistLowerF(List<Node> toCalculate, Node current)
        {
            bool exist = false;
            foreach (Node check in toCalculate)
            {
                if (check.name == current.name && check.fValue < current.fValue)
                {
                    exist = true;
                }
            }
```

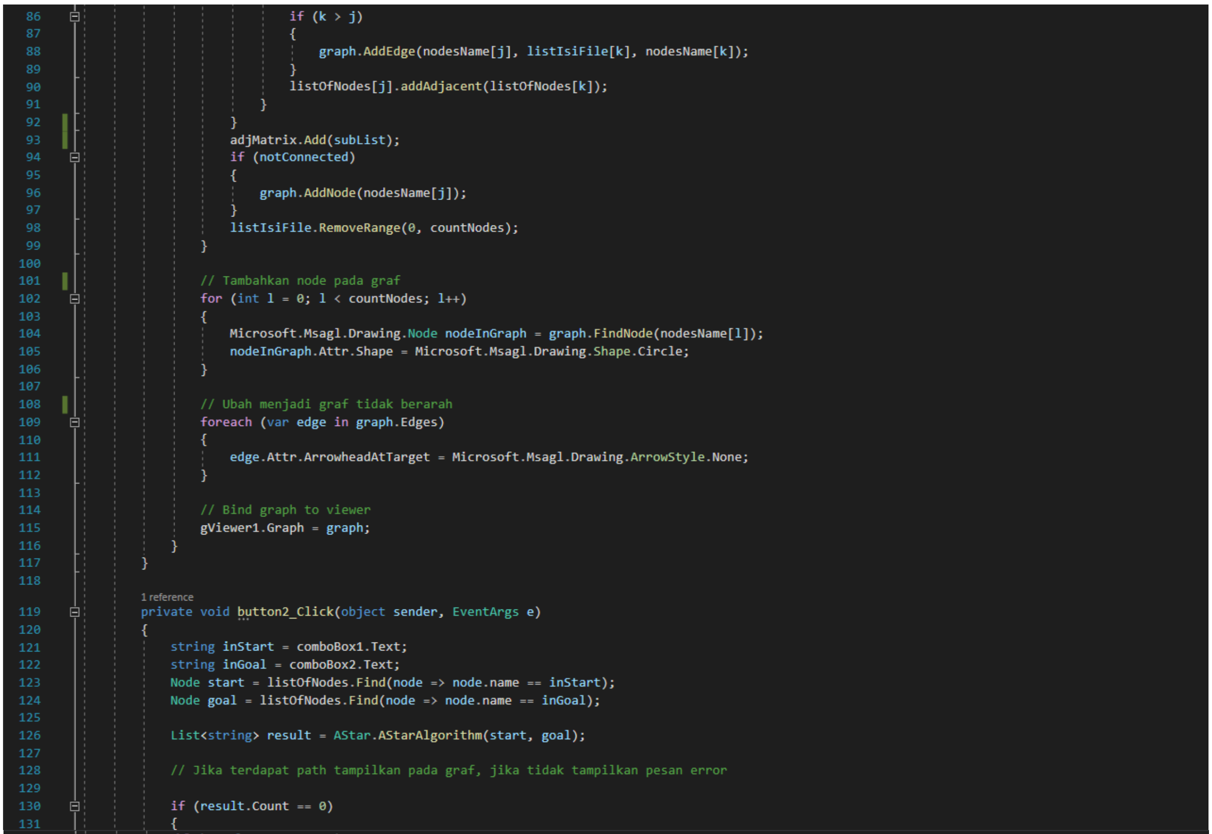


```
                }
            }
            return exist;
        }

        // Mencari node dengan f terkecil dalam List
        1 reference
        public static Node FindMinimumF(List<Node> toCalculate)
        {
            Node min = toCalculate[0];
            foreach (Node vertex in toCalculate)
            {
                if (vertex.fValue < min.fValue)
                {
                    min = vertex;
                }
            }

            return min;
        }

        // Menghitung jarak Euclidean untuk h(n)
        2 references
        public static double EuclideanDistance(Node current, Node target)
        {
            return Math.Sqrt(Math.Pow(current.x - target.x, 2) + Math.Pow(current.y - target.y, 2));
        }
    }
}
```

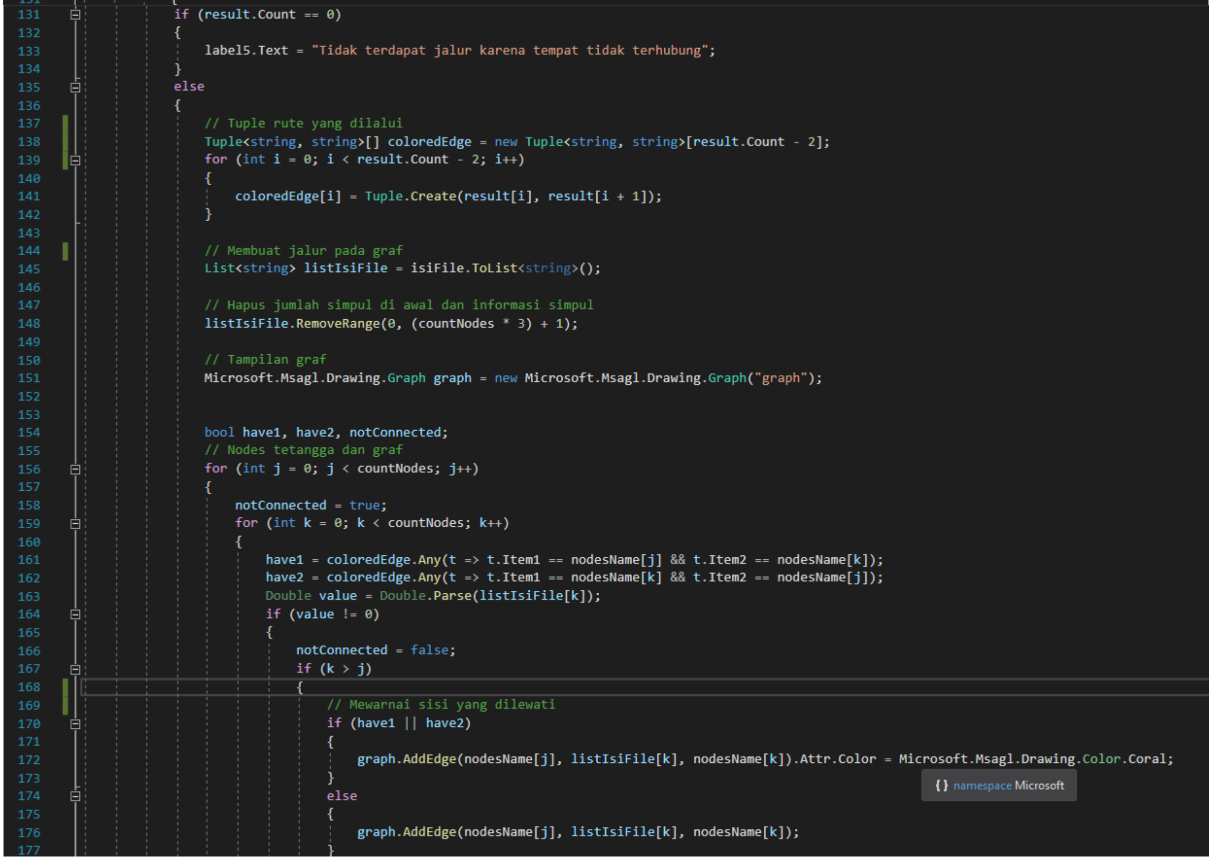
Visualisasi



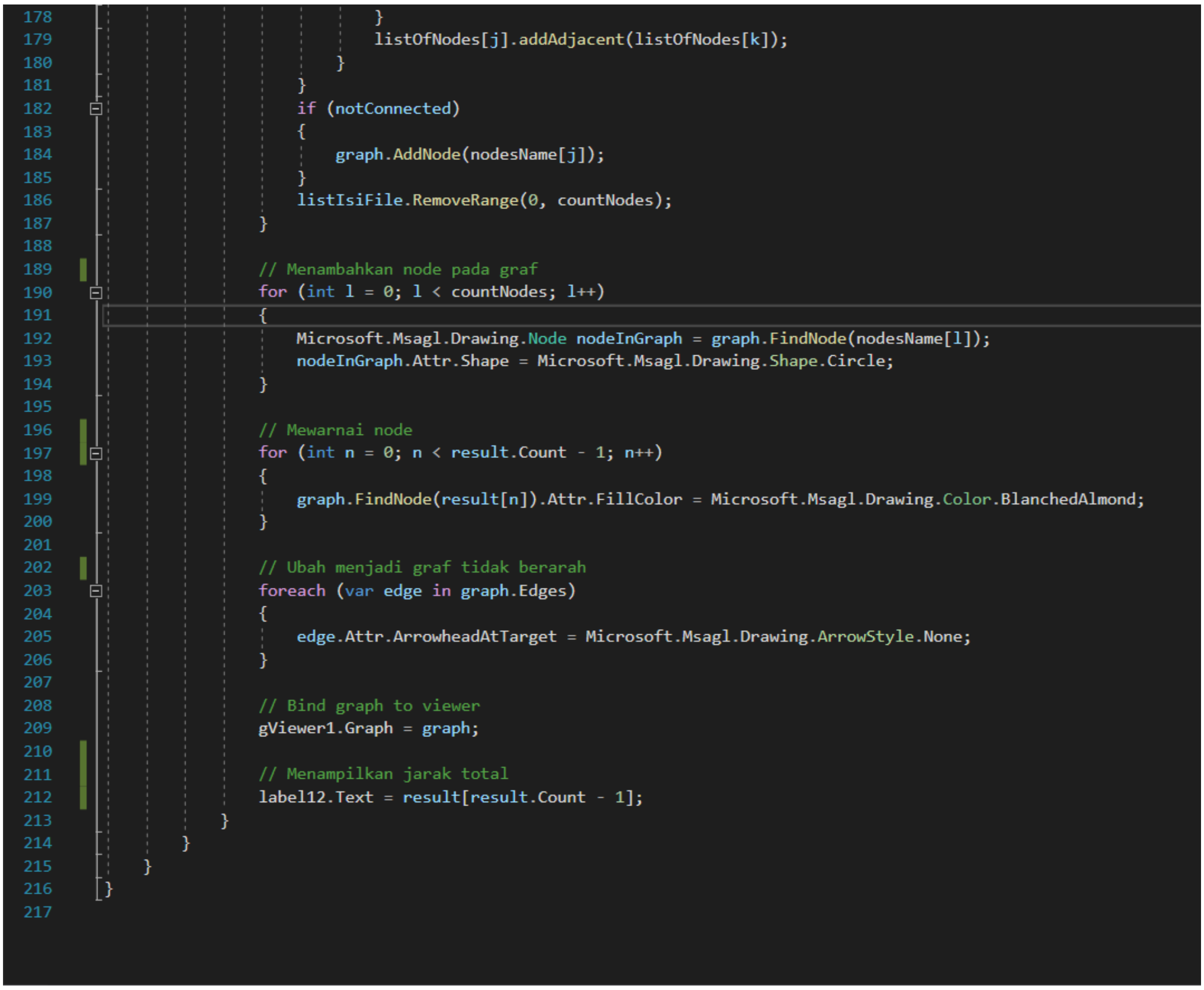
```
using System;
using System.Collections.Generic;
using System.ComponentModel;
using System.Data;
using System.Drawing;
using System.Linq;
using System.Text;
using System.Threading.Tasks;
using System.Windows.Forms;
using Tucil3Code;

namespace Tucil3Stima
{
    4 references
    public partial class Form1 : Form
    {
        1 reference
        public Form1()
        {
            InitializeComponent();
        }

        OpenFileDialog ofd = new OpenFileDialog();
        private int countNodes;
        private List<string> nodesName;
        private List<Node> listOfNodes;
        private string[] isiFile;
        private List<List<double>> adjMatrix;

        1 reference
        private void button1_Click(object sender, EventArgs e)
        {
            ofd.Filter = "Text Documents|*.txt";

            if (ofd.ShowDialog() == DialogResult.OK)
            {
                // Clear comboBox Item
                comboBox1.Items.Clear();
                comboBox2.Items.Clear();

                // Munculkan path file
                label4.Text = ofd.FileName;

                // Baca seluruh isi file
                string readFile = System.IO.File.ReadAllText(ofd.FileName);

                // Konversi isi file ke array
                isiFile = readFile.Split(new[] {"\r\n", " " }, StringSplitOptions.RemoveEmptyEntries);
                // Konversi isi file ke array
                isiFile = readFile.Split(new[] {"\r\n", " " }, StringSplitOptions.RemoveEmptyEntries);
                List<string> listIsiFile = isiFile.ToList<string>();

                // Menyimpan jumlah simpul
                countNodes = Int16.Parse(listIsiFile[0]);
                listIsiFile.RemoveAt(0);

                // List Nama Simpul
                nodesName = new List<string>();

                // List of Nodes, tambahkan di comboBox
                listOfNodes = new List<Node>();
                for (int i = 0; i < countNodes; i++)
                {
                    nodesName.Add(listIsiFile[0]);
                    comboBox1.Items.Add(listIsiFile[0]);
                    comboBox2.Items.Add(listIsiFile[0]);
                    double xValue = Double.Parse(listIsiFile[1]);
                    double yValue = Double.Parse(listIsiFile[2]);
                    Node eachNode = new Node(listIsiFile[0], xValue, yValue);
                    listOfNodes.Add(eachNode);
                    listIsiFile.RemoveRange(0, 3);
                }

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

                bool notConnected;
                adjMatrix = new List<List<double>>();
                // Nodes tetangga, graf, matriks ketetanggaan
                for (int j = 0; j < countNodes; j++)
                {
                    notConnected = true;
                    List<double> subList = new List<double>();
                    for (int k = 0; k < countNodes; k++)
                    {
                        Double value = Double.Parse(listIsiFile[k]);
                        subList.Add(value);
                        if (value != 0)
                        {
                            notConnected = false;
                            if (k > j)
                            {
                                graph.AddEdge(nodesName[j], listIsiFile[k], nodesName[k]);
                            }
                            listOfNodes[j].addAdjacent(listOfNodes[k]);
```



```
                            if (k > j)
                            {
                                graph.AddEdge(nodesName[j], listIsiFile[k], nodesName[k]);
                            }
                            listOfNodes[j].addAdjacent(listOfNodes[k]);
                        }
                    }
                    adjMatrix.Add(subList);
                    if (notConnected)
                    {
                        graph.AddNode(nodesName[j]);
                    }
                    listIsiFile.RemoveRange(0, countNodes);
                }

                // Tambahkan node pada graf
                for (int l = 0; l < countNodes; l++)
                {
                    Microsoft.Msagl.Drawing.Node nodeInGraph = graph.FindNode(nodesName[l]);
                    nodeInGraph.Attr.Shape = Microsoft.Msagl.Drawing.Shape.Circle;
                }

                // Ubah menjadi graf tidak berarah
                foreach (var edge in graph.Edges)
                {
                    edge.Attr.ArrowheadAtTarget = Microsoft.Msagl.Drawing.ArrowStyle.None;
                }

                // Bind graph to viewer
                gViewer1.Graph = graph;
            }
        }

        1 reference
        private void button2_Click(object sender, EventArgs e)
        {
            string inStart = comboBox1.Text;
            string inGoal = comboBox2.Text;
            Node start = listOfNodes.Find(node => node.name == inStart);
            Node goal = listOfNodes.Find(node => node.name == inGoal);

            List<string> result = AStar.AStarAlgorithm(start, goal);

            // Jika terdapat path tampilkan pada graf, jika tidak tampilkan pesan error

            if (result.Count == 0)
            {
```



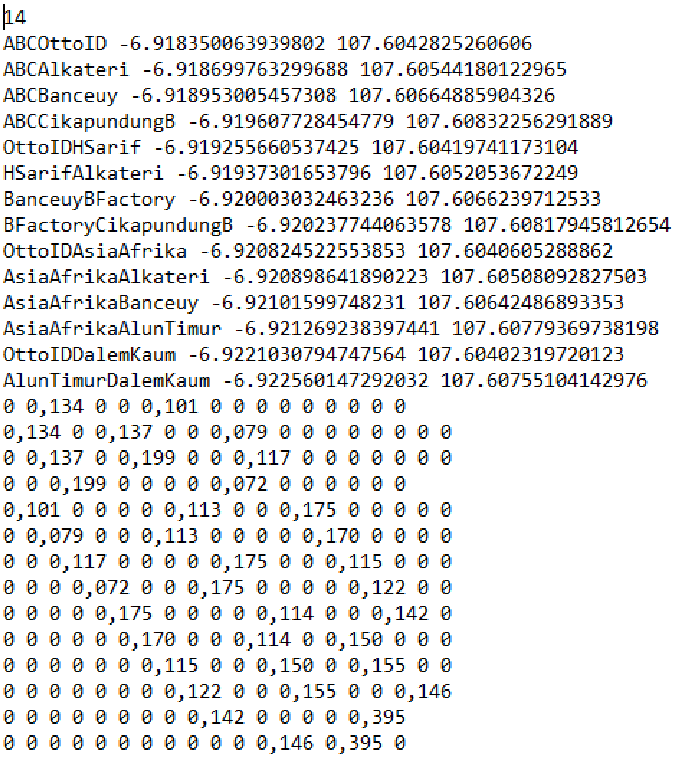
```
            if (result.Count == 0)
            {
                label5.Text = "Tidak terdapat jalur karena tempat tidak terhubung";
            }
            else
            {
                // Tuple rute yang dilalui
                Tuple<string, string>[] coloredEdge = new Tuple<string, string>[result.Count - 2];
                for (int i = 0; i < result.Count - 2; i++)
                {
                    coloredEdge[i] = Tuple.Create(result[i], result[i + 1]);
                }

                // Membuat jalur pada graf
                List<string> listIsiFile = isiFile.ToList<string>();

                // Hapus jumlah simpul di awal dan informasi simpul
                listIsiFile.RemoveRange(0, (countNodes * 3) + 1);

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


                bool have1, have2, notConnected;
                // Nodes tetangga dan graf
                for (int j = 0; j < countNodes; j++)
                {
                    notConnected = true;
                    for (int k = 0; k < countNodes; k++)
                    {
                        have1 = coloredEdge.Any(t => t.Item1 == nodesName[j] && t.Item2 == nodesName[k]);
                        have2 = coloredEdge.Any(t => t.Item1 == nodesName[k] && t.Item2 == nodesName[j]);
                        Double value = Double.Parse(listIsiFile[k]);
                        if (value != 0)
                        {
                            notConnected = false;
                            if (k > j)
                            {
                                // Mewarnai sisi yang dilewati
                                if (have1 || have2)
                                {
                                    graph.AddEdge(nodesName[j], listIsiFile[k], nodesName[k]).Attr.Color = Microsoft.Msagl.Drawing.Color.Coral;
                                }
                                else
                                {
                                    graph.AddEdge(nodesName[j], listIsiFile[k], nodesName[k]);
                                }
```



```
                            }
                            listOfNodes[j].addAdjacent(listOfNodes[k]);
                        }
                    }
                    if (notConnected)
                    {
                        graph.AddNode(nodesName[j]);
                    }
                    listIsiFile.RemoveRange(0, countNodes);
                }

                // Menambahkan node pada graf
                for (int l = 0; l < countNodes; l++)
                {
                    Microsoft.Msagl.Drawing.Node nodeInGraph = graph.FindNode(nodesName[l]);
                    nodeInGraph.Attr.Shape = Microsoft.Msagl.Drawing.Shape.Circle;
                }

                // Mewarnai node
                for (int n = 0; n < result.Count - 1; n++)
                {
                    graph.FindNode(result[n]).Attr.FillColor = Microsoft.Msagl.Drawing.Color.BlanchedAlmond;
                }

                // Ubah menjadi graf tidak berarah
                foreach (var edge in graph.Edges)
                {
                    edge.Attr.ArrowheadAtTarget = Microsoft.Msagl.Drawing.ArrowStyle.None;
                }

                // Bind graph to viewer
                gViewer1.Graph = graph;

                // Menampilkan jarak total
                label12.Text = result[result.Count - 1];
            }
        }
    }
}
```

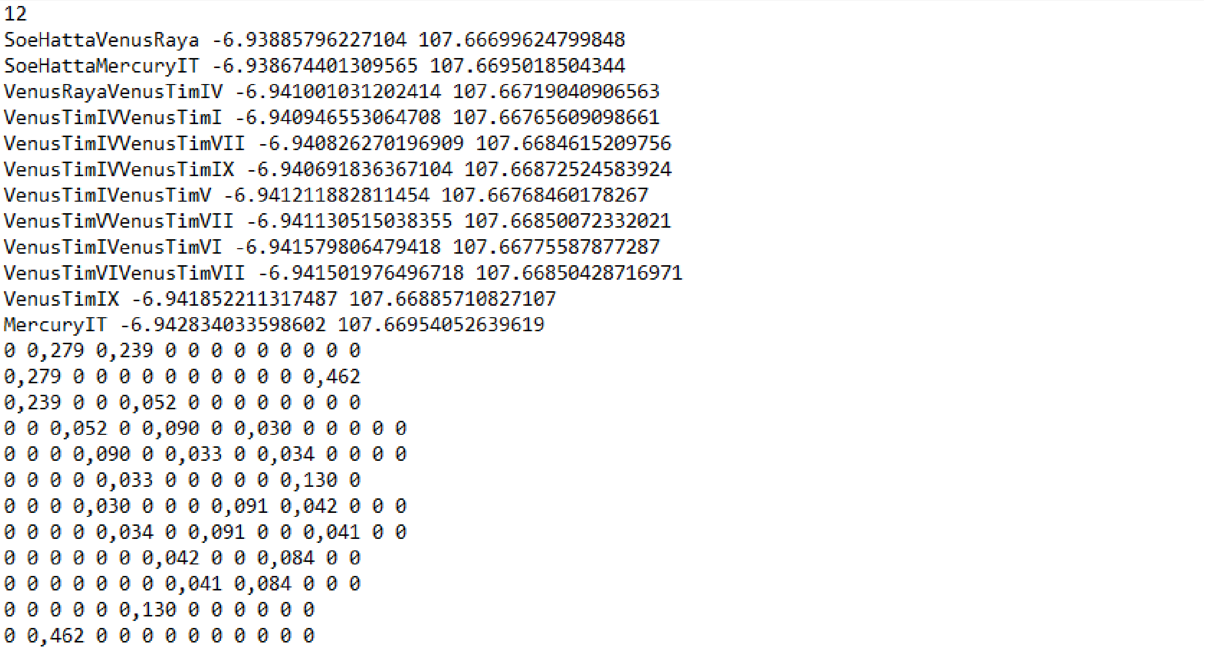
Graf Input

Peta Alun-Alun Bandung



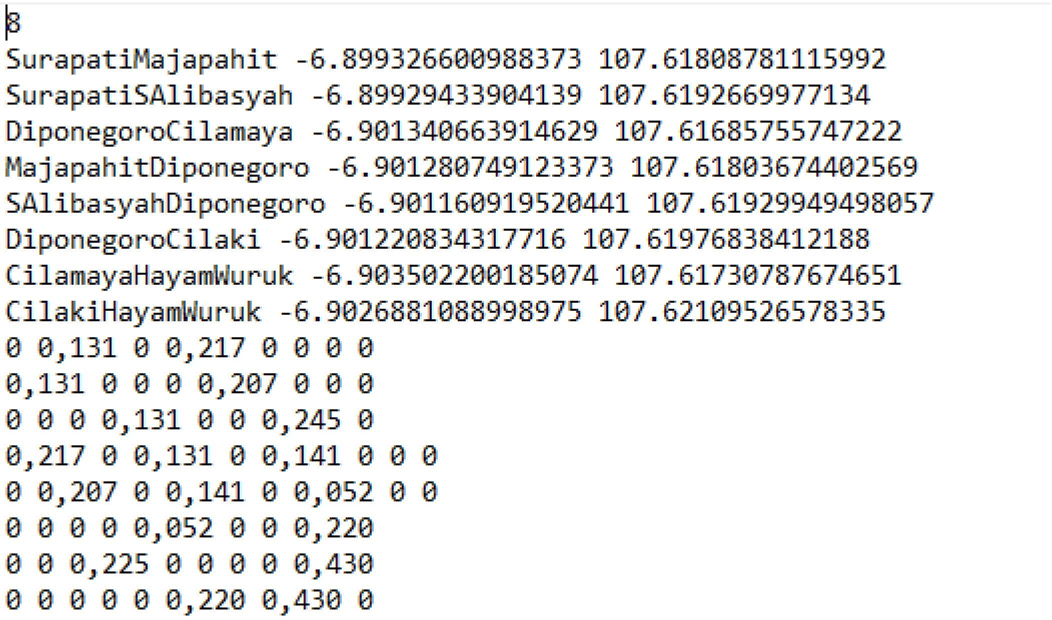
```
14
ABCOttoID -6.918350063939802 107.6042825260606
ABCAlkateri -6.918699763299688 107.60544180122965
ABCBanceuy -6.918953005457308 107.60664885904326
ABCCikapundungB -6.919607728454779 107.60832256291889
OttoIDHSarif -6.919255660537425 107.60419741173104
HSarifAlkateri -6.91937301653796 107.6052053672249
BanceuyBFactory -6.920003032463236 107.6066239712533
BFactoryCikapundungB -6.920237744063578 107.60817945812654
OttoIDAsiaAfrika -6.920824522553853 107.6040605288862
AsiaAfrikaAlkateri -6.920898641890223 107.60508092827503
AsiaAfrikaBanceuy -6.92101599748231 107.60642486893353
AsiaAfrikaAlunTimur -6.921269238397441 107.60779369738198
OttoIDDalemKaum -6.9221030794747564 107.60402319720123
AlunTimurDalemKaum -6.922560147292032 107.60755104142976
0 0,134 0 0 0,101 0 0 0 0 0 0 0 0 0
0,134 0 0,137 0 0 0,079 0 0 0 0 0 0 0 0
0 0,137 0 0,199 0 0 0,117 0 0 0 0 0 0 0
0 0 0,199 0 0 0 0 0,072 0 0 0 0 0 0
0,101 0 0 0 0 0,113 0 0 0,175 0 0 0 0 0
0 0,079 0 0 0,113 0 0 0 0 0,170 0 0 0 0
0 0 0,117 0 0 0 0 0,175 0 0 0,115 0 0 0
0 0 0 0,072 0 0 0,175 0 0 0 0 0,122 0 0
0 0 0 0 0,175 0 0 0 0 0,114 0 0 0,142 0
0 0 0 0 0 0,170 0 0 0,114 0 0,150 0 0 0
0 0 0 0 0 0 0,115 0 0 0,150 0 0,155 0 0
0 0 0 0 0 0 0 0,122 0 0 0,155 0 0 0,146
0 0 0 0 0 0 0 0 0,142 0 0 0 0 0,395
0 0 0 0 0 0 0 0 0 0 0 0,146 0,395 0
```

Peta Buah Batu



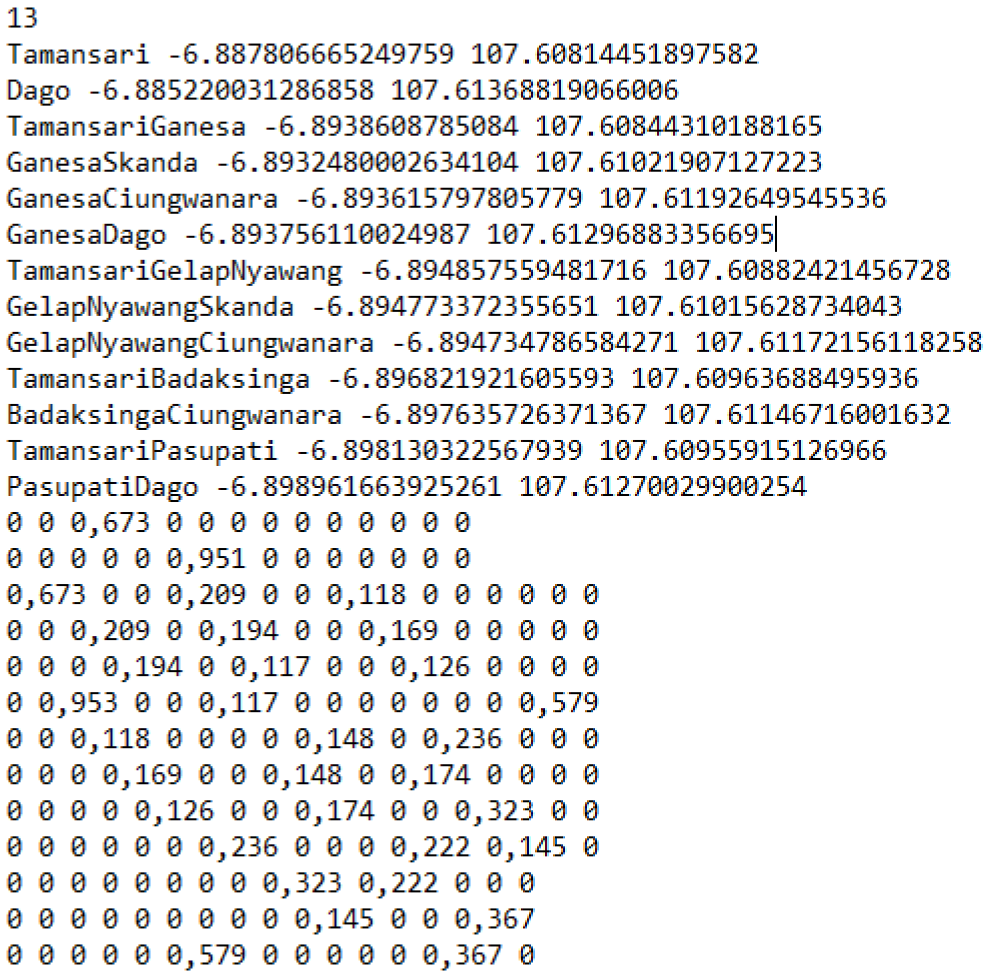
```
12
SoeHattaVenusRaya -6.93885796227104 107.66699624799848
SoeHattaMercuryIT -6.938674401309565 107.6695018504344
VenusRayaVenusTimIV -6.941001031202414 107.66719040906563
VenusTimIVVenusTimI -6.940946553064708 107.66765609098661
VenusTimIVVenusTimVII -6.94082627019690‌9 107.6684615209756
VenusTimIVVenusTimIX -6.940691836367104 107.66872524583924
VenusTimIVenusTimV -6.94121188281145‌4 107.66768460178267
VenusTimVVenusTimVII -6.941130515038355 107.66850072332021
VenusTimIVenusTimVI -6.941579806479418 107.66775587877287
VenusTimVIVenusTimVII -6.941501976496718 107.66850428716971
VenusTimIX -6.941852211317487 107.66885710827107
MercuryIT -6.942834033598602 107.66954052639619
0 0,279 0,239 0 0 0 0 0 0 0 0 0
0,279 0 0 0 0 0 0 0 0 0 0 0,462
0,239 0 0 0,052 0 0 0 0 0 0 0 0
0 0 0,052 0 0,090 0 0,030 0 0 0 0 0
0 0 0 0,090 0 0,033 0 0,034 0 0 0 0
0 0 0 0 0,033 0 0 0 0 0 0,130 0
0 0 0 0,030 0 0 0 0,091 0,042 0 0 0
0 0 0 0 0,034 0 0,091 0 0 0,041 0 0
0 0 0 0 0 0 0,042 0 0 0,084 0 0
0 0 0 0 0 0 0 0,041 0,084 0 0 0
0 0 0 0 0 0,130 0 0 0 0 0 0
0 0,462 0 0 0 0 0 0 0 0 0 0
```

Peta Gasibu



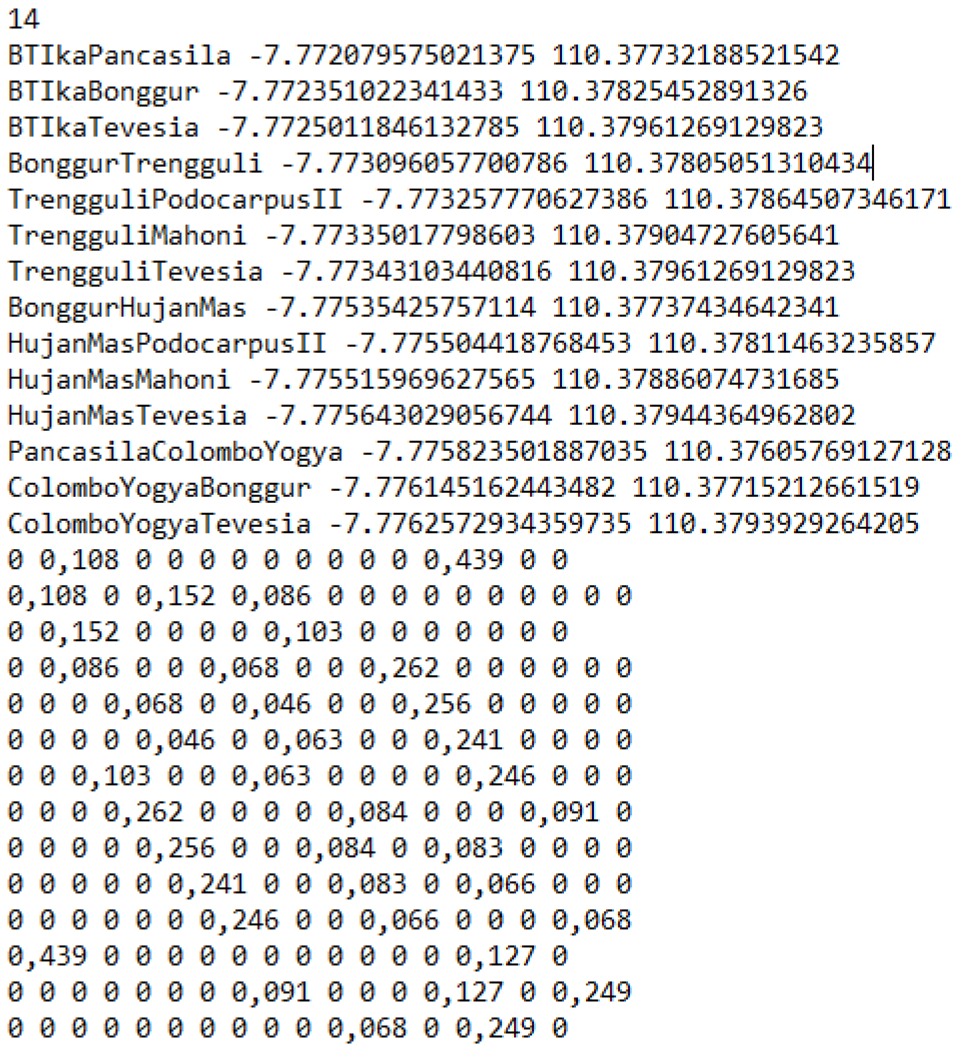
```
8
SurapatiMajapahit -6.899326600988373 107.61808781115992
SurapatiSAlibasyah -6.89929433904139 107.6192669977134
DiponegoroCilamaya -6.901340663914629 107.61685755747222
MajapahitDiponegoro -6.901280749123373 107.61803674402569
SAlibasyahDiponegoro -6.901160919520441 107.61929949498057
DiponegoroCilaki -6.901220834317716 107.61976838412188
CilamayaHayamWuruk -6.903502200185074 107.61730787674651
CilakiHayamWuruk -6.9026881088998975 107.62109526578335
0 0,131 0 0,217 0 0 0 0
0,131 0 0 0 0,207 0 0 0
0 0 0 0,131 0 0 0,245 0
0,217 0 0,131 0 0,141 0 0 0
0 0,207 0 0,141 0 0,052 0 0
0 0 0 0 0,052 0 0 0,220
0 0 0,225 0 0 0 0 0,430
0 0 0 0 0 0,220 0,430 0
```

Peta ITB



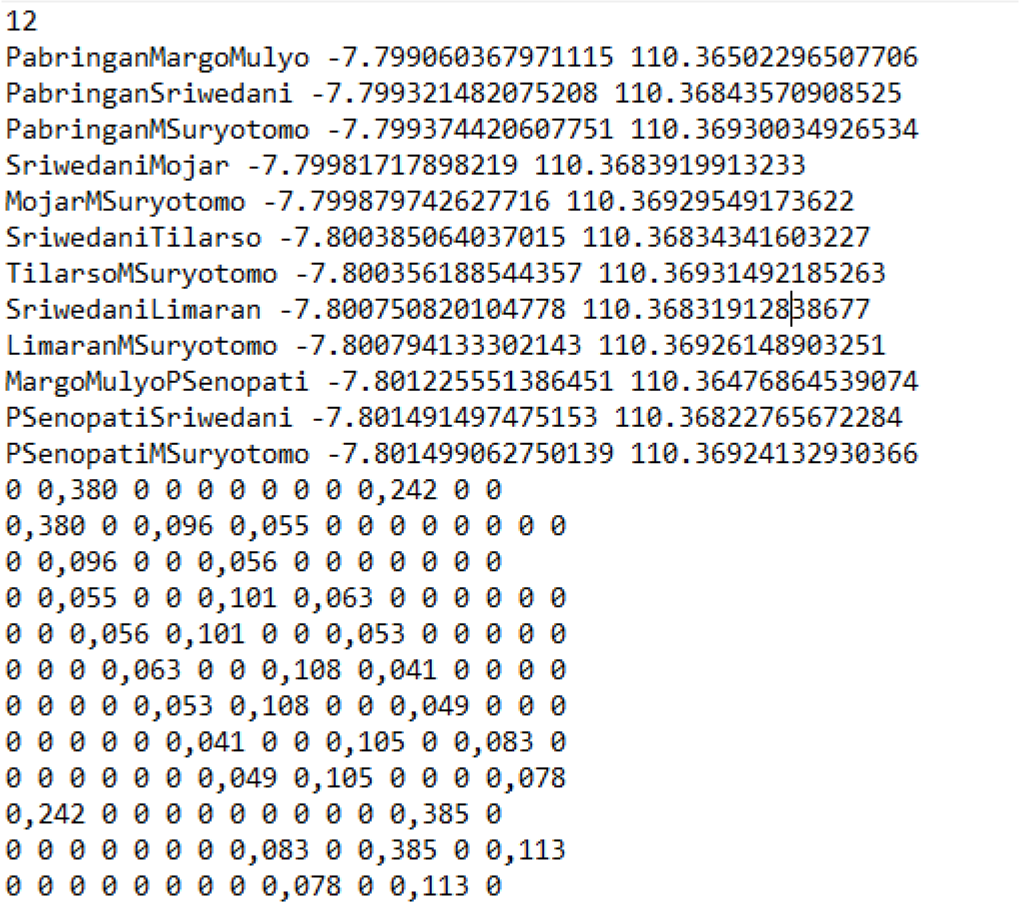
```
13
Tamansari -6.887806665249759 107.60814451897582
Dago -6.885220031286858 107.61368819066006
TamansariGanesa -6.8938608785084 107.60844310188165
GanesaSkanda -6.8932480002634104 107.61021907127223
GanesaCiungwanara -6.893615797805779 107.61192649545536
GanesaDago -6.893756110024987 107.61296883356695
TamansariGelapNyawang -6.894857559481716 107.60882421456728
GelapNyawangSkanda -6.894773372355651 107.61015628734043
GelapNyawangCiungwanara -6.894734786584271 107.61172156118258
TamansariBadaksinga -6.896821921605593 107.60963688495936
BadaksingaCiungwanara -6.897635726371367 107.61146716001632
TamansariPasupati -6.898130322567939 107.60955915126966
PasupatiDago -6.898961663925261 107.61270029900254
0 0 0,673 0 0 0 0 0 0 0 0 0 0
0 0 0 0 0 0,951 0 0 0 0 0 0 0
0,673 0 0 0,209 0 0 0,118 0 0 0 0 0 0
0 0 0,209 0 0,194 0 0 0,169 0 0 0 0 0
0 0 0 0,194 0 0,117 0 0 0,126 0 0 0 0
0 0,953 0 0 0,117 0 0 0 0 0 0 0 0,579
0 0 0,118 0 0 0 0 0,148 0 0,236 0 0 0
0 0 0 0,169 0 0 0,148 0 0,174 0 0 0 0
0 0 0 0 0,126 0 0 0,174 0 0 0,323 0 0
0 0 0 0 0 0 0,236 0 0 0 0,222 0,145 0
0 0 0 0 0 0 0 0 0,323 0,222 0 0 0
0 0 0 0 0 0 0 0 0 0,145 0 0 0,367
0 0 0 0 0 0,579 0 0 0 0 0 0,367 0
```

Peta UGM



```
14
BTIkaPancasila -7.772079575021375 110.37732188521542
BTIkaBonggur -7.772351022341433 110.37825452891326
BTIkaTevesia -7.7725011846132785 110.37961269129823
BonggurTrengguli -7.773096057700786 110.37805051310434
TrengguliPodocarpusII -7.773257770627386 110.37864507346171
TrengguliMahoni -7.77335017798603 110.37904727605641
TrengguliTevesia -7.77343103440816 110.37961269129823
BonggurHujanMas -7.77535425757114 110.37737434642341
HujanMasPodocarpusII -7.775504418768453 110.37811463235857
HujanMasMahoni -7.775515969627565 110.37886074731685
HujanMasTevesia -7.775643029056744 110.37944364962802
PancasilaColomboYogya -7.775823501887035 110.37605769127128
ColomboYogyaBonggur -7.776145162443482 110.37715212661519
ColomboYogyaTevesia -7.7762572934359735 110.3793929264205
0 0,108 0 0 0 0 0 0 0 0 0 0,439 0 0
0,108 0 0,152 0,086 0 0 0 0 0 0 0 0 0 0
0 0,152 0 0 0 0 0,103 0 0 0 0 0 0 0
0 0,086 0 0 0,068 0 0 0,262 0 0 0 0 0 0
0 0 0 0,068 0 0,046 0 0 0,256 0 0 0 0 0
0 0 0 0 0,046 0 0,063 0 0 0,241 0 0 0 0
0 0 0,103 0 0 0,063 0 0 0 0 0,246 0 0 0
0 0 0 0,262 0 0 0 0 0,084 0 0 0 0,091 0
0 0 0 0 0,256 0 0 0,084 0 0,083 0 0 0 0
0 0 0 0 0 0,241 0 0 0,083 0 0,066 0 0 0
0 0 0 0 0 0 0,246 0 0 0,066 0 0 0 0,068
0,439 0 0 0 0 0 0 0 0 0 0 0 0,127 0
0 0 0 0 0 0 0 0,091 0 0 0 0,127 0 0,249
0 0 0 0 0 0 0 0 0 0 0,068 0 0,249 0
```

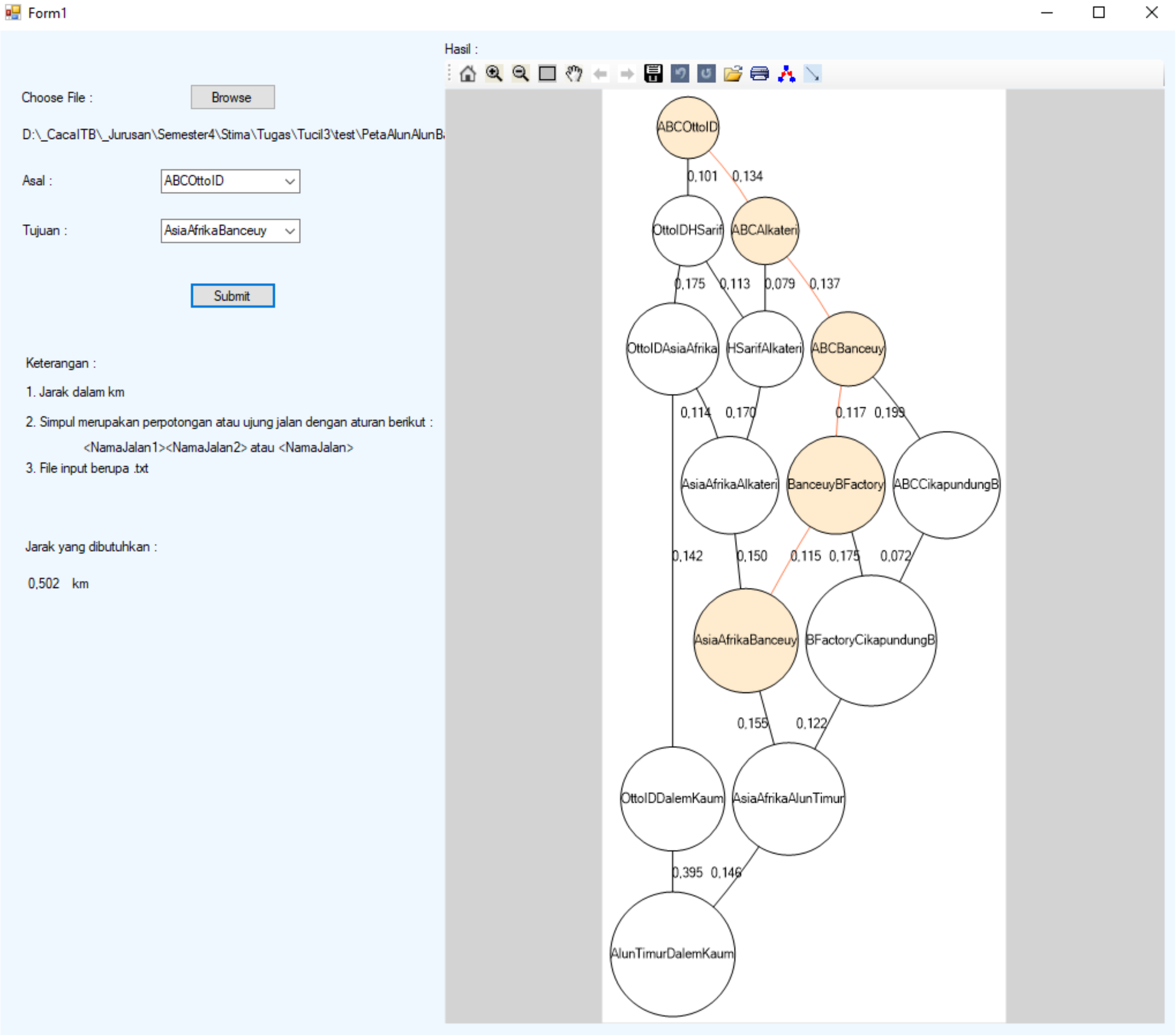
Peta Vredeburg



```
12
PabringanMargoMulyo -7.799060367971115 110.36502296507706
PabringanSriwedani -7.799321482075208 110.36843570908525
PabringanMSuryotomo -7.799374420607751 110.36930034926534
SriwedaniMojar -7.79981717898219 110.3683919913233
MojarMSuryotomo -7.799879742627716 110.36929549173622
SriwedaniTilarso -7.800385064037015 110.36834341603227
TilarsoMSuryotomo -7.800356188544357 110.36931492185263
SriwedaniLimaran -7.800750820104778 110.36831912838677
LimaranMSuryotomo -7.800794133302143 110.36926148903251
MargoMulyoPSenopati -7.801225551386451 110.36476864539074
PSenopatiSriwedani -7.80149149747515З 110.36822765672284
PSenopatiMSuryotomo -7.801499062750139 110.36924132930366
0 0,380 0 0 0 0 0 0 0 0,242 0 0
0,380 0 0,096 0,055 0 0 0 0 0 0 0 0
0 0,096 0 0 0,056 0 0 0 0 0 0 0
0 0,055 0 0 0,101 0,063 0 0 0 0 0 0
0 0 0,056 0,101 0 0 0,053 0 0 0 0 0
0 0 0 0,063 0 0 0,108 0,041 0 0 0 0
0 0 0 0 0,053 0,108 0 0 0,049 0 0 0
0 0 0 0 0 0,041 0 0 0,105 0 0,083 0
0 0 0 0 0 0 0,049 0,105 0 0 0 0,078
0,242 0 0 0 0 0 0 0 0 0 0,385 0
0 0 0 0 0 0 0 0,083 0 0,385 0 0,113
0 0 0 0 0 0 0 0 0,078 0 0,113 0
```

Beberapa Lintasan Terpendek

Peta Alun-Alun Bandung


Form1

Choose File : Browse

D:_CacaITB_Jurusan\Semester4\Stima\Tugas\Tucil3\test\PetaAlunAlunB

Asal : ABCOttoID

Tujuan : AsiaAfrikaBanceuy

Submit

Keterangan :

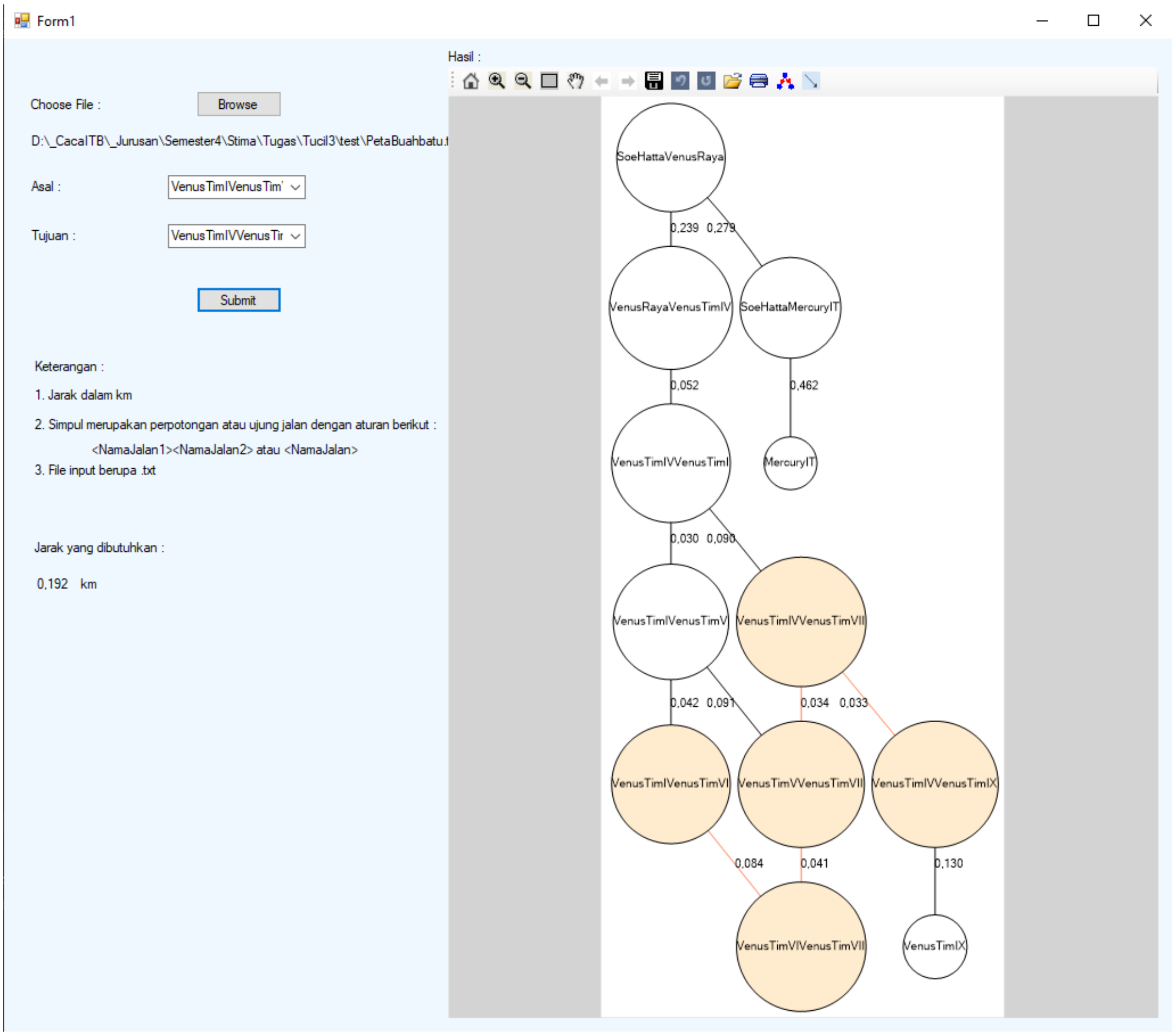
1. Jarak dalam km
2. Simpul merupakan perpotongan atau ujung jalan dengan aturan berikut :
   <NamaJalan1><NamaJalan2> atau <NamaJalan>
3. File input berupa .txt

Jarak yang dibutuhkan :

0,502 km

Hasil :

Peta Buah Batu



Form1

Choose File : Browse

D:_CacaITB_Jurusan\Semester4\Stima\Tugas\Tucil3\test\PetaBuahbatu.t

Asal : VenusTimIVenusTim'

Tujuan : VenusTimIVVenusTi

Submit

Keterangan :

1. Jarak dalam km

2. Simpul merupakan perpotongan atau ujung jalan dengan aturan berikut :
   <NamaJalan1><NamaJalan2> atau <NamaJalan>

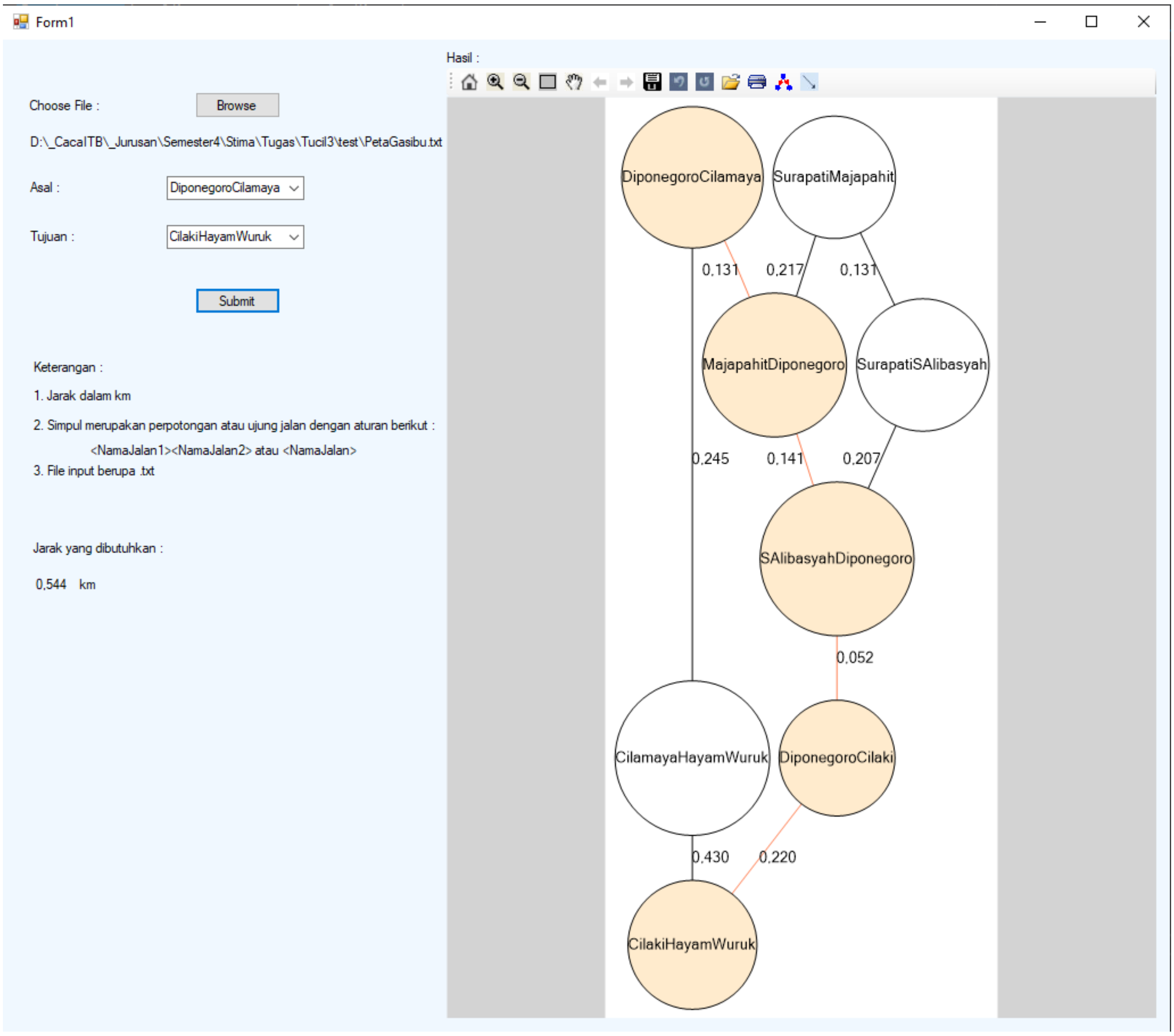
3. File input berupa .txt

Jarak yang dibutuhkan :

0,192 km

Hasil :

Peta Gasibu



Form1

Choose File :

D:_CacaITB_Jurusan\Semester4\Stima\Tugas\Tucil3\test\PetaGasibu.txt

Asal : DiponegoroCilamaya

Tujuan : CilakiHayamWuruk

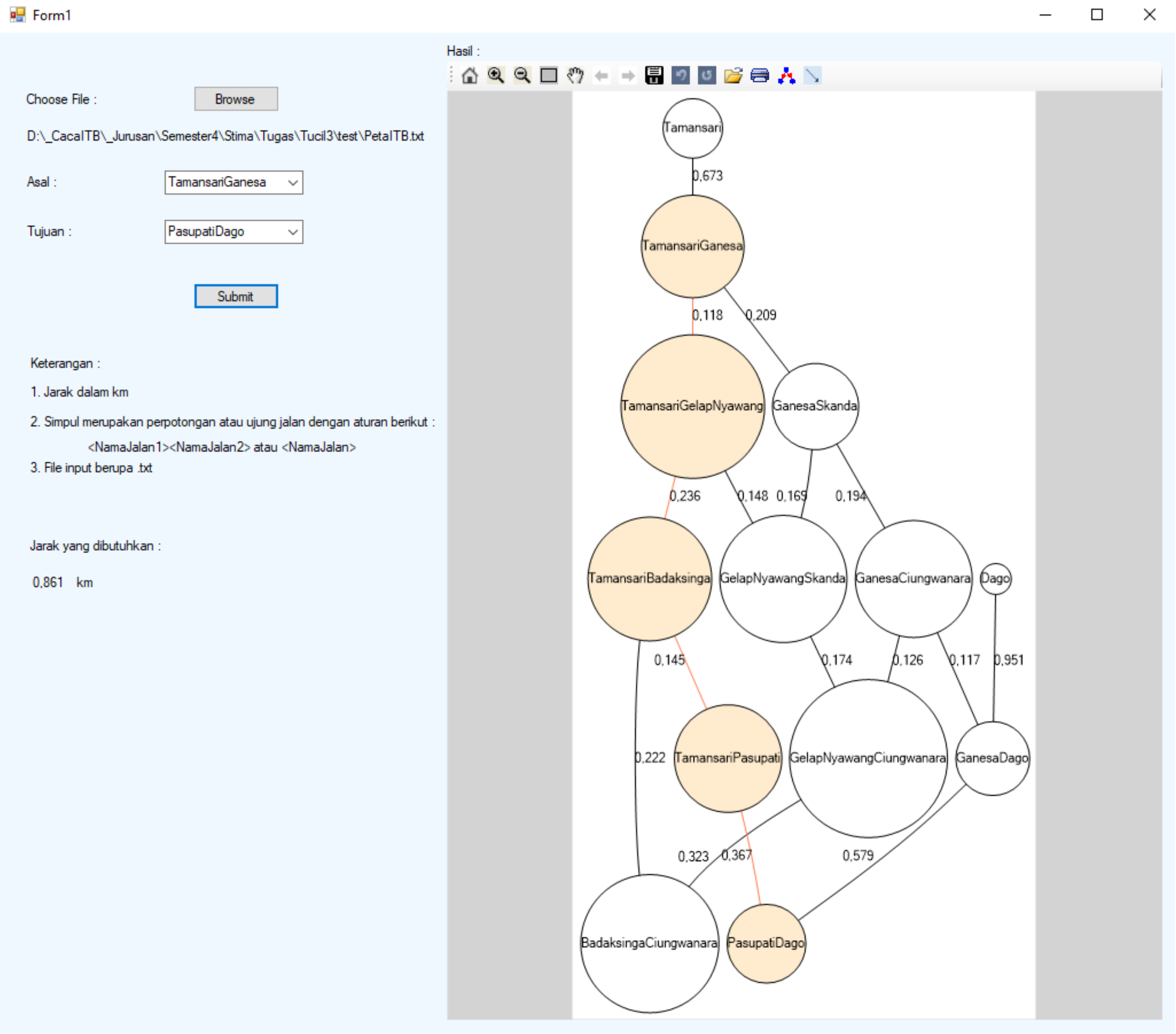
Keterangan :

1. Jarak dalam km
2. Simpul merupakan perpotongan atau ujung jalan dengan aturan berikut :
   <NamaJalan1><NamaJalan2> atau <NamaJalan>
3. File input berupa .txt

Jarak yang dibutuhkan :

0,544 km

Peta ITB



Form1

Choose File : Browse

D:_CacaITB_Jurusan\Semester4\Stima\Tugas\Tucil3\test\PetaITB.txt

Asal : TamansariGanesa

Tujuan : PasupatiDago

Submit

Keterangan :

1. Jarak dalam km

2. Simpul merupakan perpotongan atau ujung jalan dengan aturan berikut :

<NamaJalan1><NamaJalan2> atau <NamaJalan>

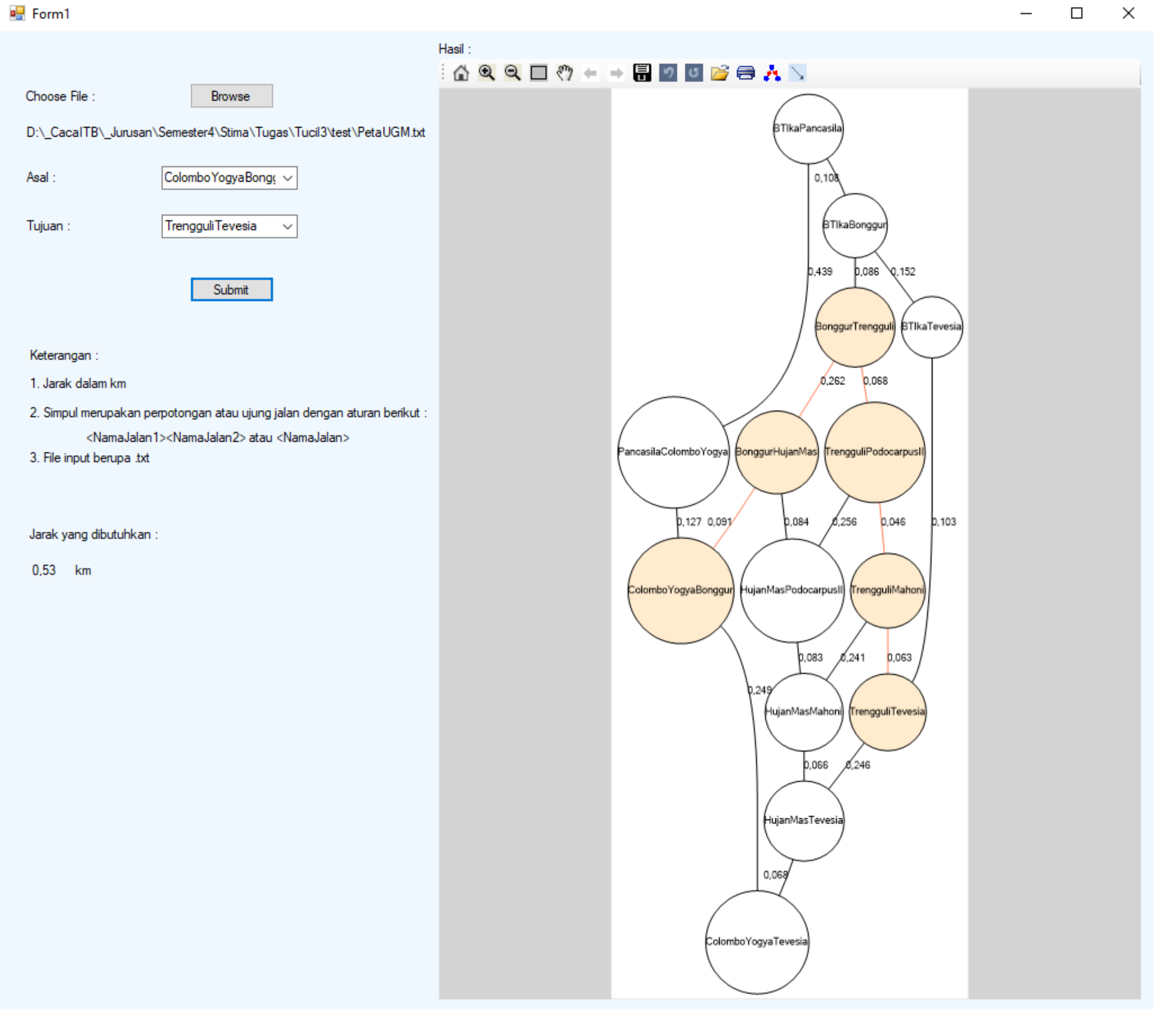
3. File input berupa .txt

Jarak yang dibutuhkan :

0,861 km

Hasil :

Peta UGM



Hasil :

Choose File : Browse

D:_CacaITB_Jurusan\Semester4\Stima\Tugas\Tucil3\test\PetaUGM.txt

Asal : ColomboYogyaBongg

Tujuan : TrengguliTevesia

Submit

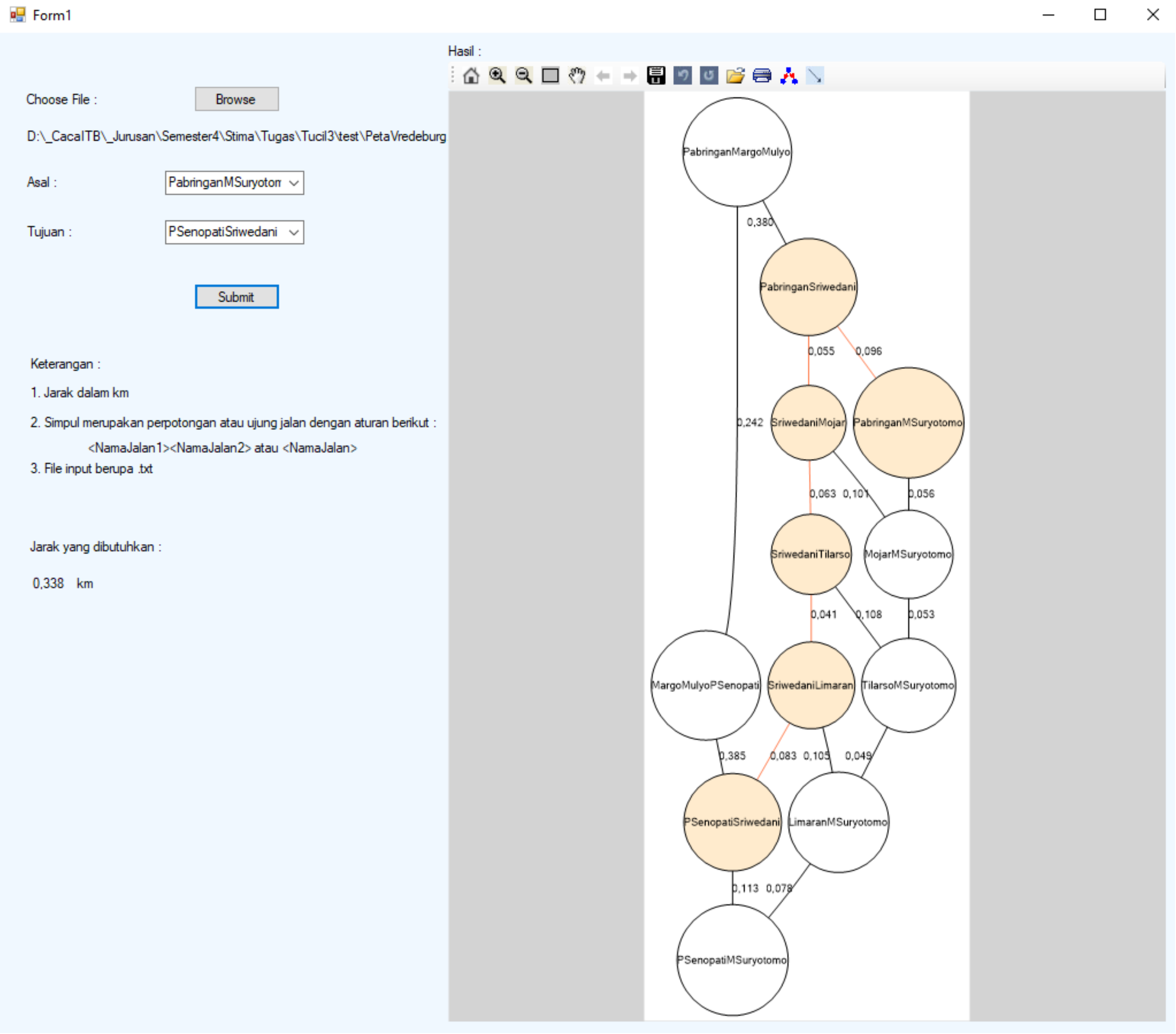
Keterangan :

1. Jarak dalam km
2. Simpul merupakan perpotongan atau ujung jalan dengan aturan berikut :
   <NamaJalan1><NamaJalan2> atau <NamaJalan>
3. File input berupa .txt

Jarak yang dibutuhkan :

0,53 km

Peta Vredeburg



Choose File :

D:_CacaITB_Jurusan\Semester4\Stima\Tugas\Tucil3\test\PetaVredeburg

Asal : PabringanMSuryotom

Tujuan : PSenopatiSriwedani

Keterangan :

1. Jarak dalam km
2. Simpul merupakan perpotongan atau ujung jalan dengan aturan berikut :
   <NamaJalan1><NamaJalan2> atau <NamaJalan>
3. File input berupa .txt

Jarak yang dibutuhkan :

0,338 km

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	

Alamat Kode Program

Link: https://github.com/AlifahRBasyasya/Tucil3-Stima