LAPORAN TUGAS KECIL 3

IF2211 – STRATEGI ALGORITMA

Implementasi Algoritma A* untuk Menentukan Lintasan Terpendek

Oleh :

Rayhan Alghifari Fauzta **13519039**

Raihan Astrada Fathurrahman **13519113**

SEKOLAH TEKNIK ELEKTRO DAN INFORMATIKA

INSTITUT TEKNOLOGI BANDUNG

2021

1. Source Code Program

```
# Modul lain yang digunakan
import os
from math import *
import networkx as nx
import matplotlib.pyplot as plt

class Graph:
  def __init__(self):
    self.components = []

  def loadFile(self, filename):
    cur_path = os.path.dirname(__file__)
    fpath = os.path.join(cur_path, '../test/'+filename)
    try:
      f = open(fpath, 'r')
      lines = f.readlines()
      for i in range(len(lines)):
        lines[i] = lines[i].replace("\n", "")
      f.close()

      # Ambil jumlah simpul
      count_nodes = int(lines[0])
      # Ambil nama simpul
      nodes = []
      for i in range(1, count_nodes + 1):
        dot = lines[i].split(" ")
        nodes.append(dot[2])
      # Ambil koordinat
      coordinates = []
      for i in range(1, count_nodes + 1):
        dot = lines[i].split(" ")
        coordinates.append((float(dot[0]), float(dot[1])))
      # Ambil adjacent list
      adj = []
      for i in range(count_nodes + 1, len(lines)):
        dot = lines[i].split(" ")
```

```
        tmp = []
        for val in dot:
          tmp.append(float(val))
        adj.append(tmp)

      for i in range(count_nodes):
        if (nodes[i] not in self.components):
          weight = []
          for j in range(count_nodes):
            if (adj[i][j] != 0):
              weight.append(self.haversine(coordinates[i],coordinates[j]))
            else:
              weight.append(0)
          value = [nodes[i],coordinates[i],adj[i],weight]
          self.components.append(value)

    except:
      raise

  def getCoordinate(self, key):
    for value in self.components:
      if (value[0] == key):
        return value[1]

  def getComponent(self, key):
    for value in self.components:
      if (value[0] == key):
        return value

  # distance in a spherical object, such as Earth, uses haversine formula
  # https://en.wikipedia.org/wiki/Haversine_formula
  def haversine(self, startPosition, targetPosition):
    r = 6378  # earth radius in equator (kilometer)
    p1 = pi/180 * (targetPosition[0] - startPosition[0])
    p2 = pi/180 * (targetPosition[1] - startPosition[1])
      d = 2 * r * asin(sqrt(sin(p1/2)**2 + cos((pi/180)*targetPosition[0]) *
cos((pi/180)*startPosition[0]) * sin(p2/2)**2))
```

```
    return d

  def sphericalDistance(self, root):
    hn = {}
    for i in range(len(self.components)):
      target = self.components[i][0]
                                                                hn[target]          =
self.haversine(self.getCoordinate(root),self.getCoordinate(target))
    return hn

  def astar(self,root,target):
    queue = []
    visited = set()
    hn = self.sphericalDistance(root)

    # Element Queue = [Kota, Distance, Path]
    queue.append([root,0,[root]])
    visited.add(root)
    while (len(queue) != 0):
      fn = []
      if(queue[0][0] == target):
        # Target Found
        break
      else:
        # f(n) = g(n)+h(n)
        current = self.getComponent(queue[0][0])
        for i in range(len(current[3])):
          path = []
          for node in queue[0][2]:
            path.append(node)
          if (current[3][i] != 0 and self.components[i][0] not in visited):
            nodeName = self.components[i][0]
            fn = queue[0][1]+current[3][i]+hn.get(nodeName)

            path.append(nodeName)
            queue.append([nodeName,fn,path])
```

```
        queue.pop(0)
        if (len(queue) != 0):
          # Sort & Choose Lowest f(n)
          sorted(queue, key = lambda x: x[1])
          visited.add(queue[0][0])

    # {EOP : Head queue simpul target atau panjang queue 0}
    if (len(queue) != 0):
      print("Jarak terdekat dari "+root+" ke "+target+" adalah ", end = "")
      print('%.2f'%queue[0][1], end = "")
      print(" km dengan rute lintasan ", end="")
      print(queue[0][2])
      return queue

  def drawGraph(self, result):
    G = nx.Graph()
    nodePosition = {}

    path = []
    for i in range(len(result[0][2]) - 1):
      path.append((result[0][2][i], result[0][2][i+1]))
      path.append((result[0][2][i+1], result[0][2][i]))

    for i in range(len(self.components)):
      # Masukkin posisi simpul
      nodePosition[self.components[i][0]] = self.components[i][1]
      for j in range(len(self.components[i][2])):
        # Masukkin sisi & color sisi
        if (self.components[i][2][j]):
          if(self.components[i][0], self.components[j][0]) in path:
            if (self.components[j][0] == result[0][2][-1]):
              # Nampilin jarak
                labels = {(self.components[i][0], self.components[j][0]): "Total
jarak: "'%.2f'%result[0][1]+" km"}
            color = "red"
          else:
            color = "black"
```

```
                        G.add_edge(self.components[i][0],  self.components[j][0],
weight=self.components[i][3][j], color=color)

    # Color node
    colorNode = []
    for node in G:
      if node in result[0][2]:
        colorNode.append("blue")
      else:
        colorNode.append("white")

    options = {
        "with_labels": True,
        "node_color": colorNode,
        "edge_color": [G[i][j]["color"] for i,j in G.edges()],
        "edgecolors": "black"
    }

    # draw
    nx.draw_networkx(G, nodePosition, **options)

    # labels node & edge target
                    nx.draw_networkx_labels(G,     nodePosition,     font_size=10,
font_family="sans-serif")
    nx.draw_networkx_edge_labels(G,nodePosition,edge_labels=labels,font_size = 8)

    # show graph
    ax = plt.gca()
    ax.margins(0.08)
    plt.axis("off")
    plt.tight_layout()
    plt.show()
```

2. File Input

Tabel 1. File Input Program

No	Daerah	Input
1	ITB/Dago (File ‘itb.txt’)	12 -6.884893 107.611445 A -6.885191 107.613017 B -6.885257 107.613733 C -6.887256 107.611540 D -6.887386 107.613611 E -6.887910 107.608289 F -6.893882 107.608450 G -6.893230 107.610447 H -6.893605 107.611944 I -6.893780 107.613036 J -6.894759 107.611723 K -6.894883 107.608839 L 0 0.17686726502403394 0 0.2632517439789043 0 0 0 0 0 0 0 0 0.17686726502403394 0 0.07946859144593632 0.28192895069042706 0 0 0 0 0 0 0 0 0 0.07946859144593632 0 0 0.23737731468051107 0 0 0 0 0 0 0 0.2632517439789043 0.28192895069042706 0 0 0.22933116685049545 0.3665819638890329 0 0 0 0 0 0 0 0 0.23737731468051107 0.22933116685049545 0 0 0 0 0 0.7145925075720253 0 0 0 0 0 0.3665819638890329 0 0 0.6650237807660428 0 0 0 0 0 0 0 0 0 0 0.6650237807660428 0 0.23232125564821227 0 0 0 0.11943352380600843 0 0 0 0 0 0 0.23232125564821227 0 0.17062232500909322 0 0 0 0 0 0 0 0 0 0 0.17062232500909322 0 0.12224166728949747 0.13076103083345564 0 0 0 0 0 0.7145925075720253 0 0 0 0.12224166728949747 0 0 0 0 0 0 0 0 0 0 0 0.13076103083345564 0 0 0.3190155925057135 0 0 0 0 0 0 0.11943352380600843 0 0 0 0.3190155925057135 0
2	Alun-alun Bandung (file ‘alunbandung.txt’)	17 -6.918268 107.604297 A -6.918658 107.605451 B

```
-6.918932 107.606668 C
-6.920138 107.612217 D
-6.921770 107.612008 E
-6.921218 107.607724 F
-6.921055 107.606473 G
-6.920945 107.605065 H
-6.920830 107.604110 I
-6.922065 107.604009 J
-6.922410 107.606361 K
-6.922554 107.607605 L
-6.922839 107.609812 M
-6.923403 107.606276 N
-6.923096 107.603966 O
-6.919565 107.608346 P
-6.921266 107.608050 Q
0 0.13471176844466234 0 0 0 0 0 0 0.28594209216231176 0 0 0 0 0 0 0 0
0.13471176844466234 0 0.13790174212632667 0 0 0 0 0.2581309434978446 0
0 0 0 0 0 0 0 0
0  0.13790174212632667  0  0  0  0  0.23730659677255975  0  0  0  0  0  0  0  0
0.19836659323542816 0
0 0 0 0 0.1831317100473715 0 0 0 0 0 0 0 0 0 0.4324989310078779 0
0  0  0  0.1831317100473715  0  0  0  0  0  0  0  0  0.27027661679532633  0  0  0
0.44096569579186906
0  0  0  0  0  0  0.13942859518010012  0  0  0  0  0.14929990025384676  0  0  0  0
0.03641902208625105
0 0 0.23730659677255975 0 0 0.13942859518010012 0 0.15607347316754297 0
0 0 0 0 0 0 0 0
0 0.2581309434978446 0 0 0 0 0 0.0 0.10630680704952646 0 0 0 0 0 0 0 0
0.28594209216231176   0   0   0   0   0   0   0.10630680704952646   0
0.1379289327451398 0 0 0 0 0 0 0
0  0  0  0  0  0  0  0  0.1379289327451398  0  0.26273134540559595  0  0  0
0.11486625589233625 0 0
0  0  0  0  0  0  0  0  0  0.26273134540559595  0  0.13840040068272444  0
0.11093624643133183 0 0 0
0  0  0  0  0  0.14929990025384676  0  0  0  0  0.13840040068272444  0
0.2459405832760499 0 0 0 0
0 0 0 0 0.27027661679532633 0 0 0 0 0 0 0.2459405832760499 0 0 0 0 0
```

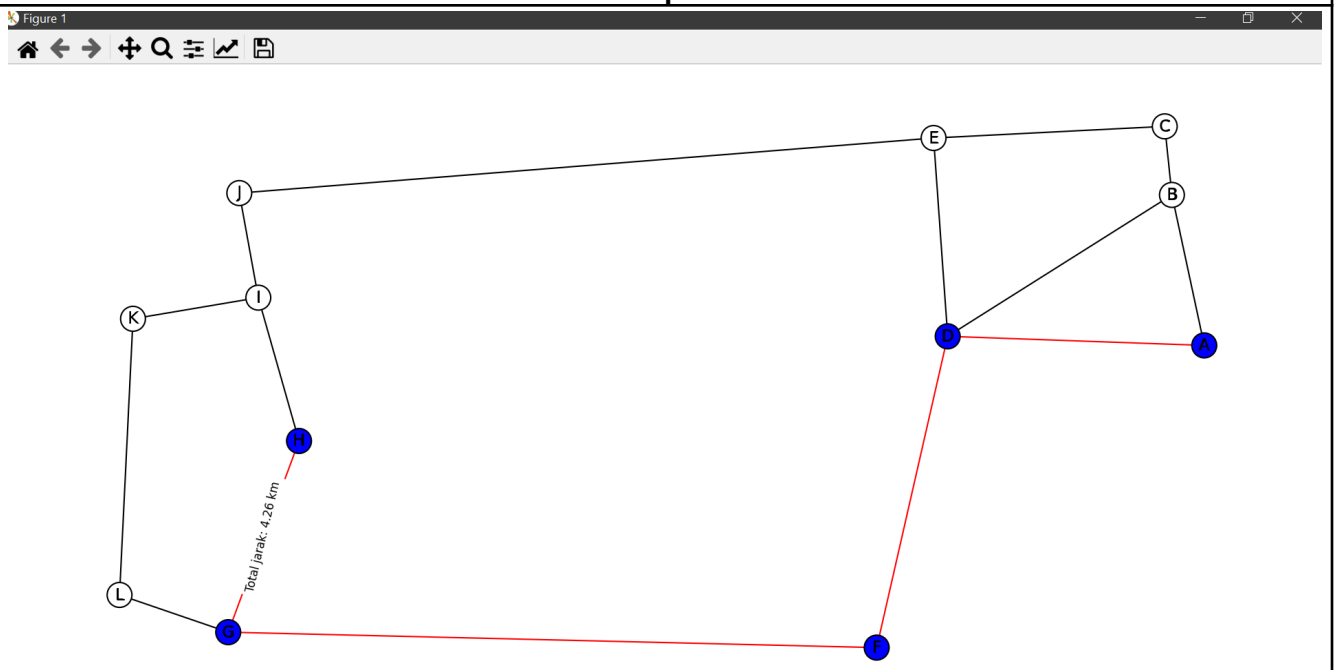
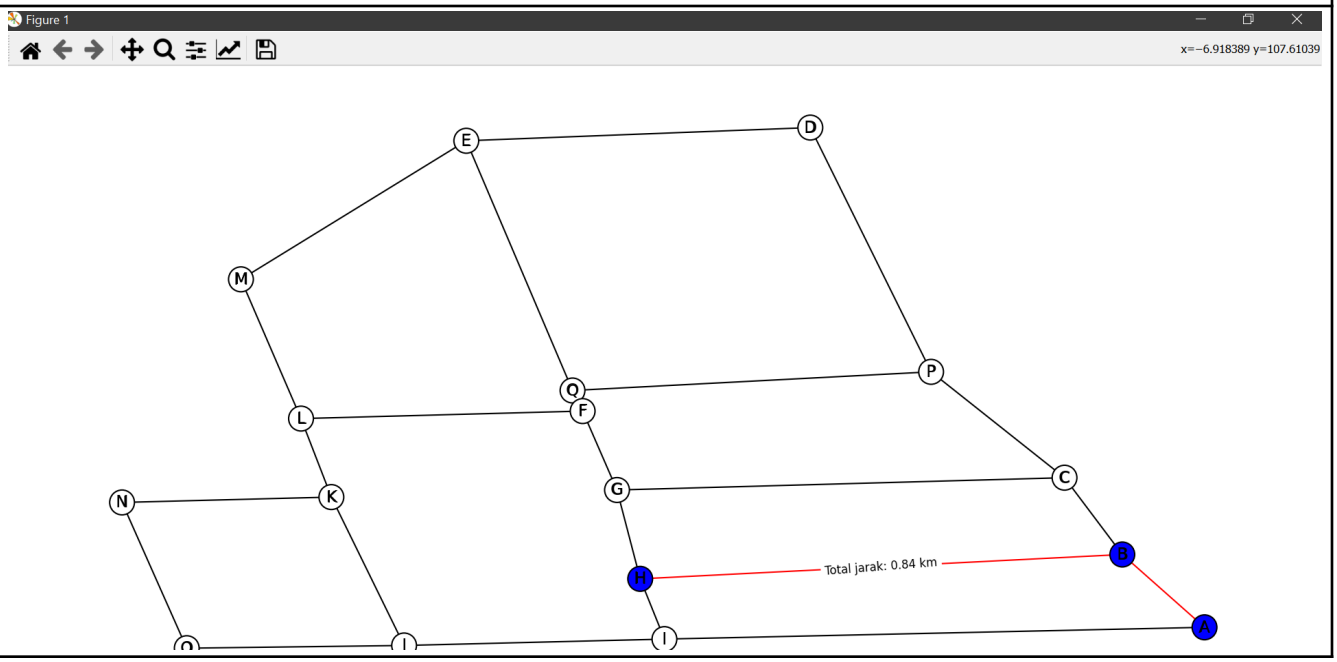
		0 0 0 0 0 0 0 0 0 0 0.11093624643133183 0 0 0 0.2575449621541899 0 0 0 0 0 0 0 0 0 0 0 0.11486625589233625 0 0 0 0.2575449621541899 0 0 0 0 0 0.19836659323542816 0.4324989310078779 0 0 0 0 0 0 0 0 0 0 0 0 0.19215488596049382 0 0 0 0 0.44096569579186906 0.03641902208625105 0 0 0 0 0 0 0 0 0 0.19215488596049382 0
3	Buahbatu (file ‘buahbatu.txt’)	8 -6.940351 107.658245 A -6.939252 107.663915 B -6.943234 107.663564 C -6.942138 107.652719 D -6.955690 107.654484 E -6.956029 107.662112 F -6.954222 107.639885 G -6.946367 107.641756 H 0 0.6383757440660554 0 0.6422143355099146 0 0 0 0 0.6383757440660554 0 0.4449583511965164 0 0 0 0 0 0 0.4449583511965164 0 0 0 1.4333107062323849 0 0 0.6422143355099146 0 0 0 1.521124159571173 0 0 1.2996715959379843 0 0 0 1.521124159571173 0 0.8437213748336192 1.621415943452029 0 0 0 1.4333107062323849 0 0.8437213748336192 0 0 0 0 0 0 0 1.621415943452029 0 0 0.8985048923087647 0 0 0 1.2996715959379843 0 0 0.8985048923087647 0
4	Jakarta (file ‘jkt.txt’)	15 -6.245131 106.788753 A -6.246568 106.791361 B -6.246584 106.791921 C -6.243914 106.791703 D -6.244357 106.790184 E -6.244625 106.789645 F -6.244878 106.789209 G -6.246105 106.791314 H -6.245636 106.791288 I -6.245589 106.791838 J -6.245394 106.790106 K

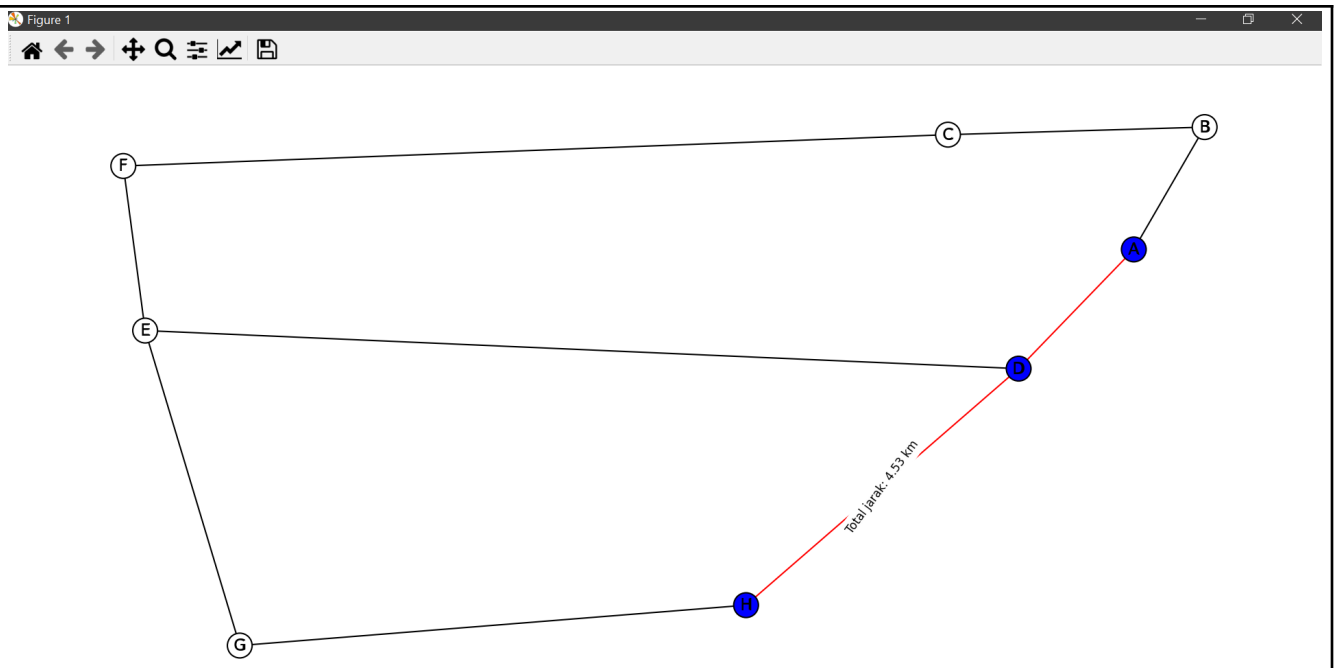
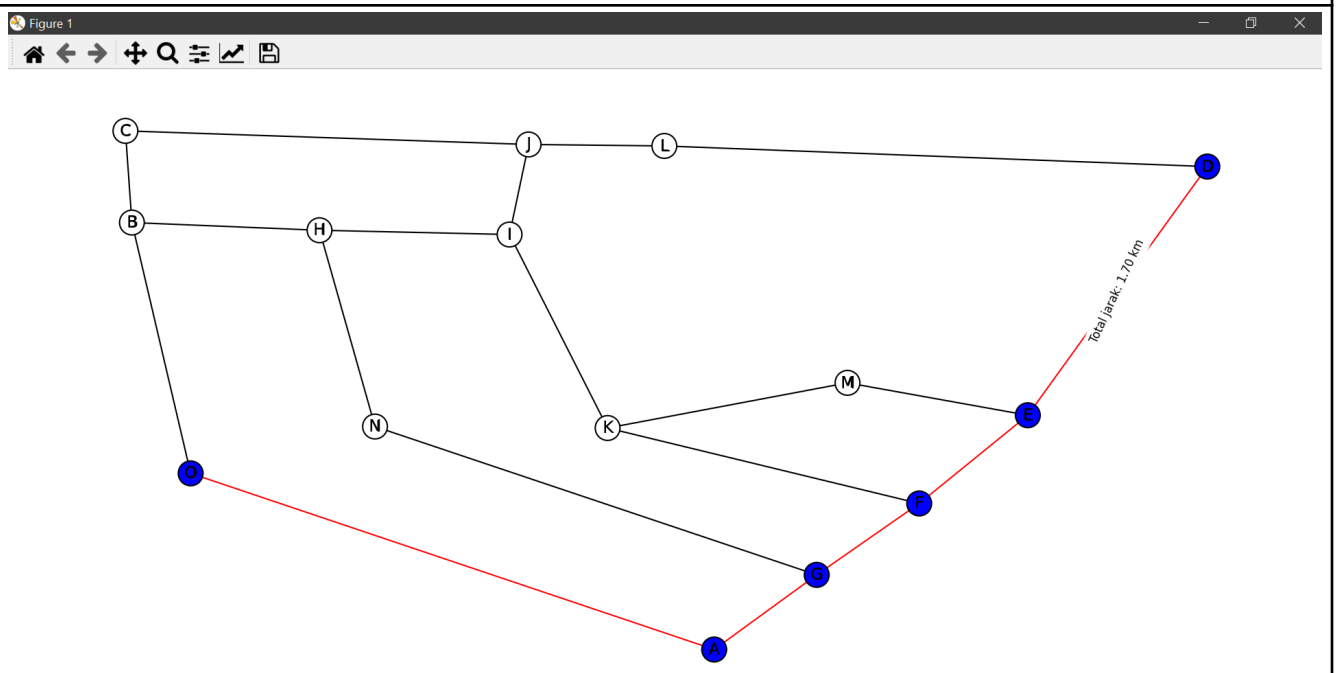
		-6.245254 106.791829 L -6.244802 106.790382 M -6.245968 106.790115 N -6.246423 106.789829 O 0 0 0 0 0 0 0.05778681391705183 0 0 0 0 0 0 0 0.18671221168035393 0 0 0.061993058403779826 0 0 0 0 0.05180155909799334 0 0 0 0 0 0 0.17029199326369335 0 0.061993058403779826 0 0 0 0 0 0 0 0.1111406588104678 0 0 0 0 0 0 0 0 0 0.17517202950873728 0 0 0 0 0 0 0.1498151235946974 0 0 0 0 0 0 0.17517202950873728 0 0.06668886476057448 0 0 0 0 0 0 0.05416525320403055 0 0 0 0 0 0 0.06668886476057448 0 0.055864744647323765 0 0 0 0.09965008321575428 0 0 0 0 0.05778681391705183 0 0 0 0 0.055864744647323765 0 0 0 0 0 0 0 0.15739551564233994 0 0 0.05180155909799334 0 0 0 0 0 0 0.05228693435673806 0 0 0 0 0.1335505105390448 0 0 0 0 0 0 0 0 0.05228693435673806 0 0.06108548367929538 0.1335412260435828 0 0 0 0 0 0 0.1111406588104678 0 0 0 0 0 0.06108548367929538 0 0 0.03730452449700794 0 0 0 0 0 0 0 0 0.09965008321575428 0 0 0.1335412260435828 0 0 0 0.07263289962435965 0 0 0 0 0 0.1498151235946974 0 0 0 0 0 0.03730452449700794 0 0 0 0 0 0 0 0 0 0.05416525320403055 0 0 0 0 0 0.07263289962435965 0 0 0 0 0 0 0 0 0 0 0.15739551564233994 0.1335505105390448 0 0 0 0 0 0 0 0.18671221168035393 0.17029199326369335 0 0 0 0 0 0 0 0 0 0 0 0 0
5	Tebet (file ‘tebetz.txt’)	8 -6.235536679520217 106.85527842250073 Smpn73 -6.233743889747193 106.85063836232926 NokiEsports -6.230013827050964 106.85246231829736 WartegWarmo -6.226889873483907 106.85824284564163 StasiunTebet -6.242227419818099 106.85416610947357 SignaturePark -6.242765361706683 106.8584218079315 StasiunCawang -6.224294522153229 106.85139329929041 CervinoVillage -6.23167178187699 106.84564704393786 McDonalds

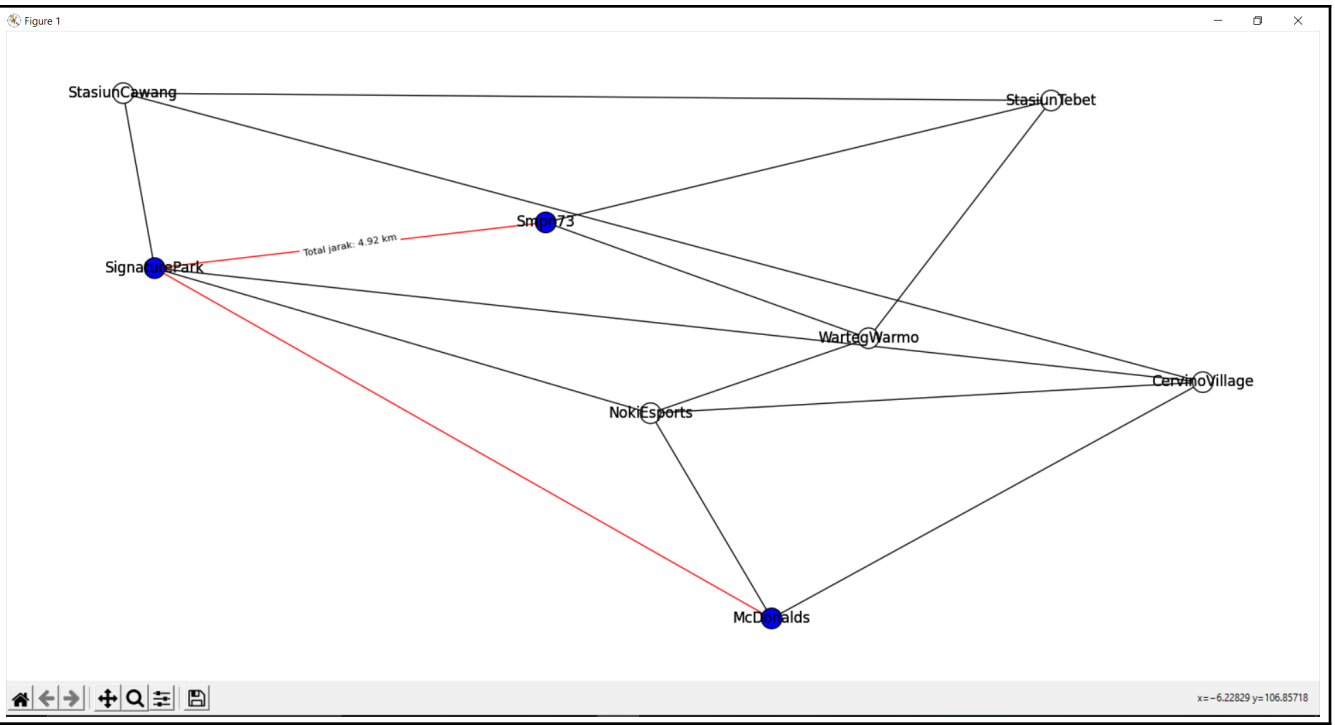
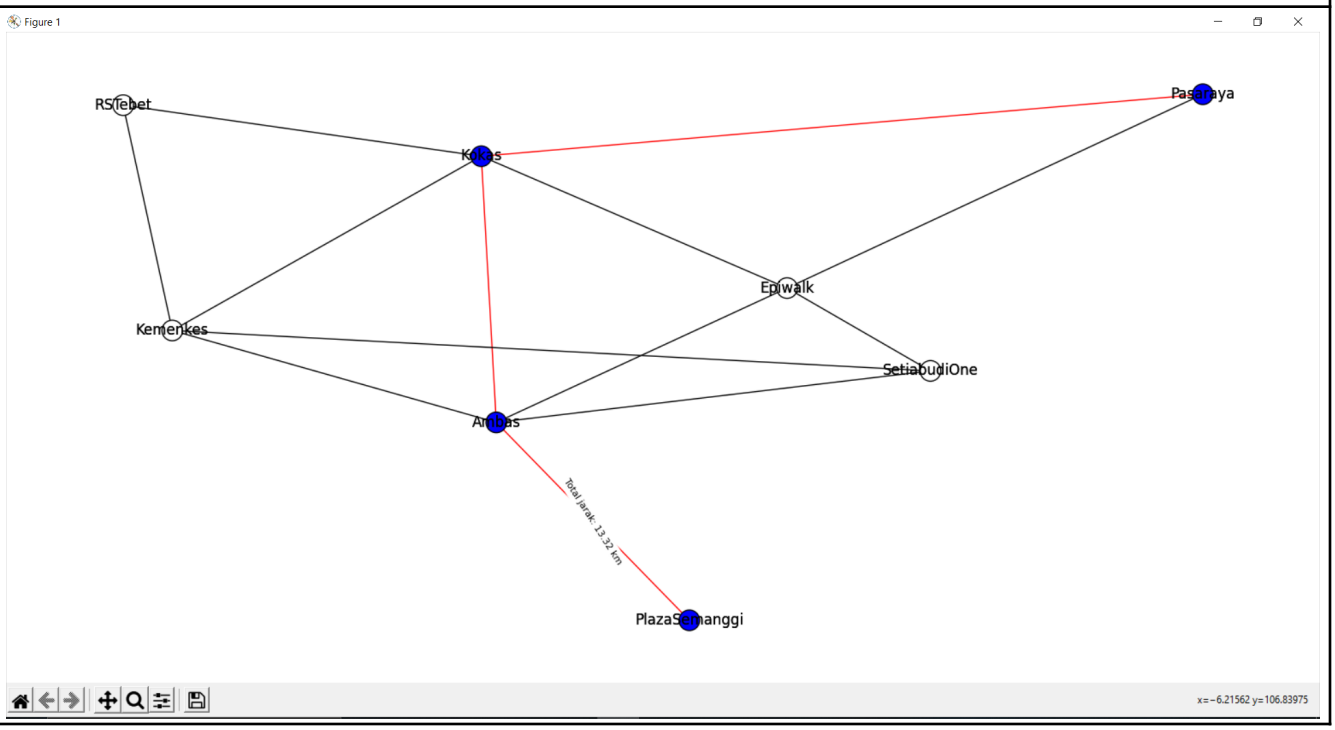
		0 0 0.6892575215773871 1.0169018356455397 0.7548960243636511 0 0 0 0 0 0.4616772684513392 0 1.0218664108126787 0 1.0551884516882268 0.5935827238558 0.6892575215773871 0.4616772684513392 0 0.7280876063783246 0 0 0 0 1.0169018356455397 0 0.7280876063783246 0 0 1.7673242638603361 0 0 0.7548960243636511 1.0218664108126787 0 0 0 0.47471506724879536 2.019695786902 1.5064432156255818 0 0 0 1.7673242638603361 0.47471506724879536 0 2.1983067832991834 0 0 1.0551884516882268 0 0 2.019681995786902 2.1983067832991834 0 1.0387758329848 0 0.5985635827238558 0 0 1.5064432156255818 0 1.0386237758329848 0
6	Jakarta Selatan (file ‘jaksel.txt’)	8 -6.223541633664353 106.8432401838233 Kokas -6.223229131621265 106.82669727332109 Ambas -6.219205651311271 106.81439814983958 PlazaSemanggi -6.208502262545617 106.84709102760661 Pasaraya -6.217168332633138 106.83504634912669 Epiwalk -6.21417953456092 106.82989828073448 SetiabudiOne -6.229983123423717 106.83241053812739 Kemenkes -6.231006655533594 106.8464132842394 RSTebet 0 1.83 0 1.73 1.15 0 1.39 0.9 1.83 0 1.43 0 1.14 1.07 2.34 0 0 1.43 0 0 0 0 0 0 1.73 0 0 0 1.64 0 0 0 1.15 1.14 0 1.64 0 0.66 0 0 0 1.07 0 0 0.66 0 1.78 0 1.39 2.34 0 0 0 1.78 0 1.55 0.9 0 0 0 0 0 1.55 0

3. Screenshot Output

Tabel 2. Screenshot Output Program

No	Output
1	
2	

3	 Figure 1 C B F A E D Total jarak: 4.53 km H G
4	 Figure 1 C J L D B H I total jarak: 1.70 km M E N K O F G A

5	
6	

4. Alamat Kode Program

Link Github : https://github.com/rayfazt/stima-astar

5. Checklist Program

Tabel 3. Checklist Program

Poin	Ya	Tidak
1. Program dapat menerima input graf	V	
2. Program dapat menghitung lintasan terpendek	V	
3. Program dapat menampilkan lintasan terpendek serta jaraknya	V	
4. Bonus : Program dapat menerima input peta dengan Google Map API dan menampilkan peta		V