IMPLEMENTASI ALGORITMA A*
UNTUK MENENTUKAN LINTASAN TERPENDEK

LAPORAN TUGAS KECIL 3

Diajukan sebagai salah satu Tugas Kecil 3

IF2211 Strategi Algoritma Semester II tahun 2020/2021

Oleh:

M. Hilal Alhamdy (13519024)

Mgs. Tabrani (13519122)

PROGRAM STUDI TEKNIK INFORMATIKA

SEKOLAH TEKNIK ELEKTRO DAN INFORMATIKA

INSTITUT TEKNOLOGI BANDUNG

BANDUNG

2021

BAGIAN I

ALGORITMA A*

A. Algoritma A*

Algoritma A* (Astar) merupakan salah satu algoritma yang termasuk dalam kategori metode pencarian yang memiliki informasi (informed search method). Algoritma ini sangat baik sebagai solusi proses path finding (pencari jalan). Algoritma ini mencari jarak rute terpendek yang akan ditempuh suatu point awal (starting point) sampai ke objek tujuan. Teknik pencarian yang digunakan dalam simulasi ini adalah menggunakan Algoritma A* dengan fungsi heuristic. Tujuan utama penelitian ini mempelajari cara kerja algoritma A* dalam mencari jarak tercepat, yang disimulasikan seperti kondisi ketika seorang mencari rute dalam keadaan jalanan macet. Simulasi ini memberikan gambaran yang lebih realistis terhadap perilaku algoritma A* dalam pencarian jarak rute terpendek.

B. Implementasi Algoritma A* di Program

Mula-mula input dari file yang berupa nama lokasi, koordinat, beserta lokasi-lokasi yang saling terhubung diubah menjadi graf. Kemudian dipilih lokasi (simpul) awal dan akhir. Simpul awal akan melihat simpul-simpul tetangganya dan menghitung jarak euclidean antara simpul awal dan simpul tetangga ditambah jarak euclidean dari simpul tetangga ke simpul akhir. Simpul tetangga yang memiliki nilai terkecil akan dikunjungi terlebih dahulu. Begitu seterusnya sampai menemukan simpul akhir dengan nilai terkecil.

BAGIAN II

SOURCE CODE PROGRAM

A. File astar.py

```
from graf import Graf

#Algoritma A*
def astar(graf, nodeFrom, nodeTo):
   nodeFrom = nodeFrom
   result = [[nodeFrom]]
   currentState = [nodeFrom]
   cost = [0]
   currentCost = 0
   while(nodeFrom != nodeTo):
       nodeAdjacent = graf.getNodeAdjacent(nodeFrom)
       for i in range(len(result)):
           if(result[i] == currentState):
               result.pop(i)
               cost.pop(i)
               break

       for i in range(len(nodeAdjacent)):
           temp = [[] for i in range(len(currentState))]
           for j in range(len(temp)):
               temp[j] = currentState[j]
           temp.append(nodeAdjacent[i])
           result.append(temp)
                       cost.append(currentCost  +  graf.getDistant(nodeFrom,nodeAdjacent[i])  +
graf.getDistant(nodeAdjacent[i], nodeTo))

       for i in range(len(result)):
           if(cost[i] == min(cost)):
               currentState = result[i]
               currentCost = cost[i] - graf.getDistant(result[i][-1], nodeTo)
               nodeFrom = result[i][len(result[i])-1]
               break

   fixResult = []
   fixCost = 0
   for i in range(len(result)):
       if(cost[i] == min(cost) and result[i][-1] == nodeTo):
           fixResult = result[i]
           fixCost = cost[i] - graf.getDistant(result[i][-1], nodeTo)
           break

   return (fixResult,fixCost)

#Pencarian node menggunakan algoritma DFS
def dfsExploreNode(graf, nodeFrom, nodeTo, dikunjungi, hasil):
   if(dikunjungi[graf.getIdxNode(nodeFrom)] == 0):
       dikunjungi[graf.getIdxNode(nodeFrom)] = 1
       hasil.append(nodeFrom)
   if(nodeFrom == nodeTo):
       return True
   else:
```

```
        if(len(graf.getNodeAdjacent(nodeFrom)) == 0):
            return False
        else:
            isExist = False
            connectNodes = []
            adjacent = graf.getNodeAdjacent(nodeFrom)
            for i in range(len(graf.getNodeAdjacent(nodeFrom))):
                connectNodes.append(adjacent[i])
            connectNodes.sort()
            for i in range(len(graf.getNodeAdjacent(nodeFrom))):
                if(dikunjungi[graf.getIdxNode(connectNodes[i])] == 0):
                    isExist = True
                    break
            if(not isExist):
                hasil.pop(len(hasil) - 1)
                if(len(hasil) == 0):
                    return False
                else:
                        return dfsExploreNode(graf, hasil[len(hasil) - 1], nodeTo, dikunjungi,
hasil)
            else:
                return dfsExploreNode(graf, connectNodes[i], nodeTo, dikunjungi, hasil)
```

B. *File* graf.py

```
class Graf:
    def __init__(self, nodes, adjacentNode, nodeCoordinate):
        self.nodes = nodes
        self.adjacentNode = adjacentNode
        self.nodeCoordinate = nodeCoordinate

    def getNumOfNode(self):
        return len(self.nodes)

    def getIdxNode(self, node):
        for i in range(len(self.nodes)):
            if(node == self.nodes[i]):
                return i

    def getNodeCoordinate(self, node):
        return self.nodeCoordinate[self.getIdxNode(node)]

    def getNodeAdjacent(self, node):
        return self.adjacentNode[self.getIdxNode(node)]

    def getAbsis(self, node):
        coordinate = self.getNodeCoordinate(node)
        return coordinate[0]

    def getOrdinat(self, node):
        coordinate = self.getNodeCoordinate(node)
        return coordinate[1]

    def getDistant(self, nodeFrom, nodeTo):
        return ((self.getAbsis(nodeFrom) - self.getAbsis(nodeTo))**2 +
(self.getOrdinat(nodeFrom) - self.getOrdinat(nodeTo))**2)**(1/2)
```

C. *File* grafVisualization.py

```
import networkx as nx
import matplotlib.pyplot as plt

def grafVisualization(g, nodes, edges, found, answer, choosenEdges, nodeFrom, nodeTo):
   #Set posisi node
   pos = {}
   for i in range(g.getNumOfNode()):
       pos[nodes[i]] = g.getNodeCoordinate(nodes[i])

   #Visulisasi Graph
   G = nx.Graph()
   G.add_nodes_from(nodes)
   G.add_edges_from(edges)
   nx.draw_networkx(G, pos, node_color="blue", font_size=8, edge_color="blue")

   #Jika jalur ketemu
   if(found):
       nx.draw_networkx(G, pos, nodelist=answer[0], node_color="red", font_size=8,
edgelist=choosenEdges, edge_color="red")
       nx.draw_networkx(G, pos, nodelist=[answer[0][-1]], node_color="green", font_size=8,
edgelist=choosenEdges, edge_color="red")
       plt.xlabel("Jarak terdekat dari " + nodeFrom + " ke " + nodeTo + " adalah " +
str(answer[1]*100) + " km")

   #Jika tidak punya jalur
   else:
       nx.draw_networkx(G, pos, nodelist=[nodeFrom], node_color="red", font_size=8,
edge_color="blue")
       nx.draw_networkx(G, pos, nodelist=[nodeTo], node_color="green", font_size=8,
edge_color="blue")
       plt.xlabel(nodeFrom + " ke " + nodeTo + " tidak memiliki jalur")

   #Menampilkan graf
   plt.show()
```

D. *File* mapVisualization.py

```
import folium

def mapVisualization(g,nodeFrom,nodeTo,nodes,answer,found, choosenEdges):
   m = folium.Map(location=[g.getOrdinat(nodeTo), g.getAbsis(nodeTo)], zoom_start=20)

   #Mewarnai seluruh node dan edge
   for i in range(g.getNumOfNode()):
       folium.Marker([g.getOrdinat(nodes[i]), g.getAbsis(nodes[i])],popup=nodes[i],
icon=folium.Icon(color='blue', icon='leaf')).add_to(m)
       for j in range(g.getNumOfNode()):
           adjacentNode = g.getNodeAdjacent(nodes[j])
           for k in range(len(g.getNodeAdjacent(nodes[j]))):

folium.PolyLine([[g.getOrdinat(nodes[j]),g.getAbsis(nodes[j])],[g.getOrdinat(adjacentNode[k]),g
.getAbsis(adjacentNode[k])]], color="blue").add_to(m)
```

```
    folium.Marker([g.getOrdinat(nodeFrom), g.getAbsis(nodeFrom)],popup=nodeFrom,
icon=folium.Icon(color='red', icon='leaf')).add_to(m)
    folium.Marker([g.getOrdinat(nodeTo), g.getAbsis(nodeTo)],popup=nodeTo,
icon=folium.Icon(color='green', icon='leaf')).add_to(m)

    #Mewarnai rute

    #Jika jalur ketemu
    if(found):
        for i in range(len(answer[0])):
            folium.Marker([g.getOrdinat(answer[0][i]),
g.getAbsis(answer[0][i])],popup=answer[0][i], icon=folium.Icon(color='red',
icon='leaf')).add_to(m)
        folium.Marker([g.getOrdinat(answer[0][-1]),
g.getAbsis(answer[0][i])],popup=answer[0][-1], icon=folium.Icon(color='green',
icon='leaf')).add_to(m)
        for i in range(len(choosenEdges)):
            text = "Jarak terdekat dari " + nodeFrom + " ke " + nodeTo + " adalah " +
str(answer[1]*100) + " km"

folium.PolyLine([[g.getOrdinat(choosenEdges[i][0]),g.getAbsis(choosenEdges[i][0])],[g.getOrdina
t(choosenEdges[i][1]),g.getAbsis(choosenEdges[i][1])]], color="red", tooltip=text).add_to(m)

    #Jika jalur tidak ada
    else:
        text = nodeFrom + " ke " + nodeTo + " tidak memiliki jalur"
        for j in range(g.getNumOfNode()):
            adjacentNode = g.getNodeAdjacent(nodes[j])
            for k in range(len(g.getNodeAdjacent(nodes[j]))):

folium.PolyLine([[g.getOrdinat(nodes[j]),g.getAbsis(nodes[j])],[g.getOrdinat(adjacentNode[k]),g
.getAbsis(adjacentNode[k])]], color="blue", tooltip=text).add_to(m)

    #Menyimpan map
    m.save('map.html')
```

E. *File* input.py

```
#Input file
print(50*"=")
fileName = input("Masukkan file: ")
fileName = open(fileName, 'r')
numOfNode = int(fileName.readline())

#Graf Variables
nodes = []
adjacentNode = []
nodeCoordinate = []
edges = []

#Mendapatkan koordinat dan node
for i in range(numOfNode):

    #Mendapatkan koordinat
    content = fileName.readline()
    j = 1
```

```
    x = ''
    y = ''
    node = ''
    while(content[j] != ','):
        x += content[j]
        j += 1
    j += 1
    while(content[j] != ')'):
        y += content[j]
        j += 1
    nodeCoordinate.append((float(x),float(y)))
    j += 1

    #Mendapatkan node
    while(content[j] != '\n'):
        node += content[j]
        j += 1
    nodes.append(node)

#Mendapatkan adjacent node
for i in range(numOfNode):
    #Inisialisasi adjacent
    adjacentNode.append([])

    #Menentukan adjacent node
    adjacent = fileName.readline()
    adjacent = list(map(str, adjacent.split(' ')))
    if(adjacent[len(adjacent)-1][-1] == '\n'):
        adjacent[len(adjacent)-1] = adjacent[len(adjacent)-1][:-1]

    #Memasukkan adjacent node
    for j in range(numOfNode):
        if(adjacent[j] == '1'):
            adjacentNode[i].append(nodes[j])
            edges.append([nodes[i],nodes[j]])
```

F. *File* main.py

```
from astar import *
from input import *
from mapVisualization import *
from grafVisualization import *

#Membuat graf dari file eksternal
g = Graf(nodes, adjacentNode, nodeCoordinate)

while(True):
    #Input node awal dan node akhir
    print(50*"=")
    for i in range(g.getNumOfNode()):
        print(str(i+1) + ". " + nodes[i])
    nodeFromInt = int(input("Masukkan lokasi awal (angka sesuai yang di atas): "))
    nodeToInt = int(input("Masukkan lokasi tujuan (angka yang sesuai di atas): "))
    nodeFrom = nodes[nodeFromInt-1]
    nodeTo = nodes[nodeToInt-1]
    #Mencari apakah nodeFrom dan nodeTo memiliki jalur
```

```
hasil = []
dikunjungi = []
for i in range(g.getNumOfNode()):
    dikunjungi.append(0)
found = dfsExploreNode(g, nodeFrom, nodeTo, dikunjungi, hasil)

answer = []
choosenEdges = []

print(50*"=")
#Jika ada jalur
if(found):
    #Mencari jarak dan rute terpendek
    answer = astar(g,nodeFrom, nodeTo)
    print("Jarak terdekat: " + str(answer[1]*100) + " km")
    print("Rute yang ditempuh: ", end=" ")
    for i in range(len(answer[0])-1):
        print(answer[0][i], end=" -> ")
    print(answer[0][len(answer[0])-1])

    #Set edge yang diwarnai berbeda
    for i in range(len(answer[0]) - 1):
        choosenEdges.append([answer[0][i],answer[0][i+1]])

#Jika tidak ada jalur
else:
    print(nodeFrom, "ke", nodeTo, "tidak memiliki jalur.")

#Menu pilih visualisasi
print(50*"=")
print("Visualisasi:")
print("1. Graf")
print("2. Map")
visualize = int(input("Masukkan pilihan: "))

if(visualize == 1):
    grafVisualization(g, nodes, edges, found, answer, choosenEdges, nodeFrom, nodeTo)
elif(visualize == 2):
    mapVisualization(g,nodeFrom,nodeTo,nodes,answer, found, choosenEdges)

#Menu lanjut program
print(50*"=")
print("Lanjut?")
print("1. Ya")
print("2. Tidak")
pilihan = int(input("Masukkan pilihan: "))

if(pilihan == 1):
    continue
elif(pilihan == 2):
    exit()
```

BAGIAN III

TANGKAPAN LAYER

A. Case 1 : Peta jalan sekitar kampus ITB/Dago

1. *Input :*

```
15
(107.610430,-6.893164)Pintu Depan ITB
(107.611293,-6.893300)Masjid Salman ITB
(107.611949,-6.893594)Simpang3 Masjid Salman ITB
(107.611733,-6.894748)Jl Ciungwanara
(107.610154,-6.894768)Jl Gelap Nyawang
(107.609868,-6.893293)Jl Skanda
(107.608830,-6.894876)Jl Tamansari
(107.608441,-6.893875)Jl Tamansari 82
(107.608060,-6.891888)Kebun Binatang
(107.608095,-6.888586)Batan
(107.608332,-6.887800)Tikungan
(107.609947,-6.887704)Sabuga
(107.611456,-6.887379)Pintu Belakang ITB
(107.613555,-6.887409)Jl Dago
(107.612971,-6.893763)Jl Djuanda
0 1 0 0 0 1 0 0 0 0 0 0 0 0 0
1 0 1 0 0 0 0 0 0 0 0 0 0 0 0
0 1 0 1 0 0 0 0 0 0 0 0 0 0 1
0 0 1 0 1 0 0 0 0 0 0 0 0 0 0
0 0 0 1 0 1 1 0 0 0 0 0 0 0 0
1 0 0 0 1 0 0 1 0 0 0 0 0 0 0
0 0 0 0 1 0 0 1 0 0 0 0 0 0 0
0 0 0 0 0 1 1 0 1 0 0 0 0 0 0
0 0 0 0 0 0 0 1 0 1 0 0 0 0 0
0 0 0 0 0 0 0 0 1 0 1 0 0 0 0
0 0 0 0 0 0 0 0 0 1 0 1 0 0 0
0 0 0 0 0 0 0 0 0 0 1 0 1 0 0
0 0 0 0 0 0 0 0 0 0 0 1 0 1 0
0 0 0 0 0 0 0 0 0 0 0 0 1 0 1
0 0 1 0 0 0 0 0 0 0 0 0 0 1 0
```

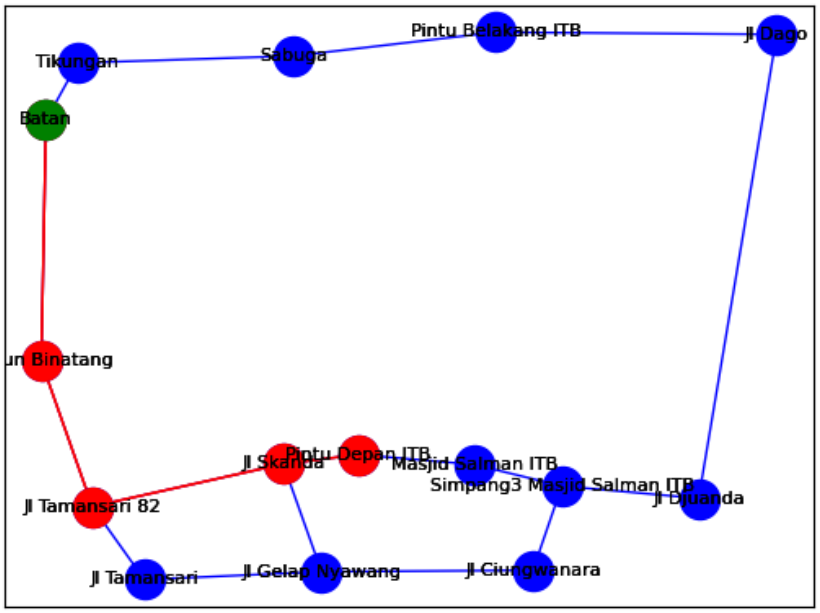
Lokasi awal : Pintu Depan ITB
Lokasi tujuan : Batan

2. *Output :*

Jarak terdekat: 0.7443119271101578 km
Rute yang ditempuh: Pintu Depan ITB -> Jl Skanda -> Jl Tamansari 82 -> Kebun Binatang -> Batan

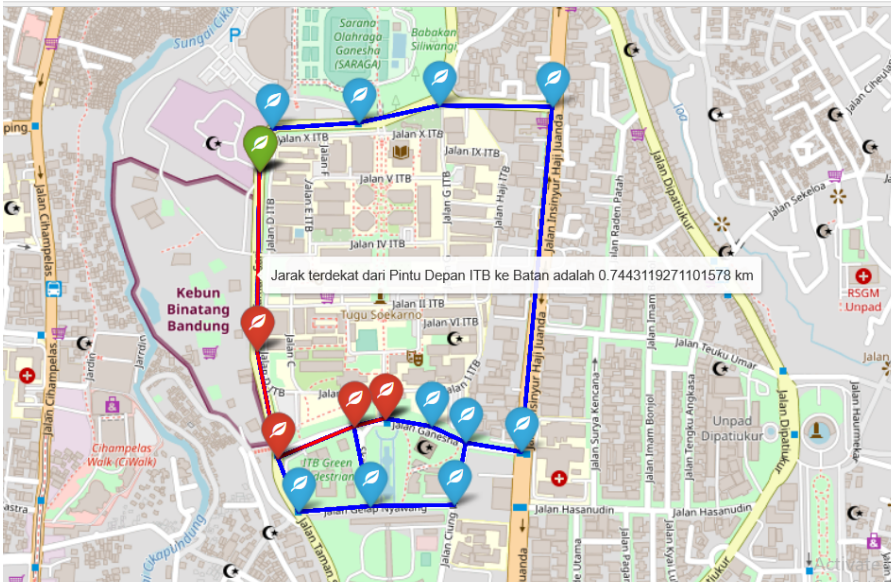
Hasil output visualisasi :

1. Graf



Jarak terdekat dari Pintu Depan ITB ke Batan adalah 0.7443119271101578 km

2. Map/Peta



B. Case 2 : Peta jalan sekitar Alun-alun Bandung

1. *Input :*

10
(107.607625,-6.921283)Tugu Asia Afrika
(107.607915,-6.921278)Terowongan Jalan Asia-Afrika
(107.608301,-6.921289)PT. PLN (Persero) Unit Induk Distribusi Jawa Barat
(107.607191,-6.921119)ATM BANK BRI
(107.607669,-6.921497)Masjid Agung Bandung
(107.607616,-6.921790)Aneka Rasa Toko
(107.607624,-6.922560)Ambu Kitchen
(107.608712,-6.921329)Kebab Bosman Cikawao
(107.608805,-6.921023)Asia Afrika
(107.609384,-6.921413)Wisma Bumiputera
0 1 0 1 1 0 0 0 0 0
1 0 1 0 0 0 0 0 0 0
0 1 0 0 0 0 0 1 0 0
1 0 0 0 0 0 0 0 0 0
1 0 0 0 0 1 0 0 0 0
0 0 0 0 1 0 1 0 0 0
0 0 0 0 0 1 0 0 0 0
0 0 1 0 0 0 0 0 1 1
0 0 0 0 0 0 0 1 0 0
0 0 0 0 0 0 0 1 0 0

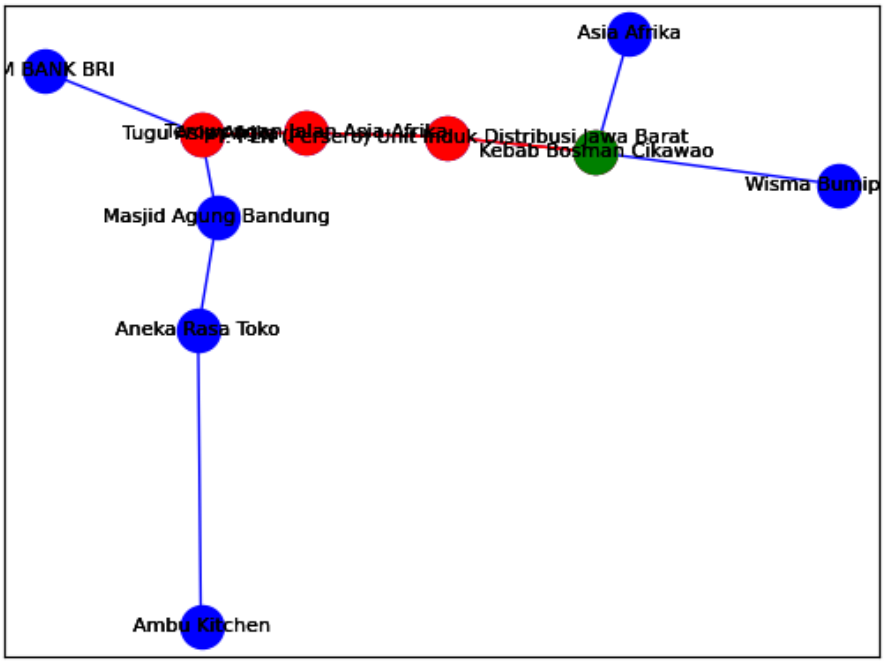
Lokasi awal : Tugu Asia Afrika
Lokasi tujuan : Kebab Bosman Cikawao

2. *Output :*

Jarak terdekat: 0.10891416887175921 km
Rute yang ditempuh: Tugu Asia Afrika -> Terowongan Jalan Asia-Afrika -> PT. PLN (Persero) Unit Induk Distribusi Jawa Barat -> Kebab Bosman Cikawao

Hasil output visualisasi :

1. Graf



rak terdekat dari Tugu Asia Afrika ke Kebab Bosman Cikawao adalah 0.1089141688717

2. Map/Peta

C. Case 3 : Peta jalan sekitar Buahbatu

1. *Input :*

10
(107.641804,-6.946293)SAMSAT
(107.640250,-6.954075)Jl Margacinta
(107.641692,-6.954372)BRI
(107.647529,-6.954982)Jl Cipalago
(107.651788,-6.955439)BORMA
(107.655117,-6.955761)Samudra Bangunan
(107.643774,-6.944861)Universitas Islam Nusantara
(107.646891,-6.943967)Bengkel Motor
(107.648994,-6.944346)Cafe Resto
(107.648282,-6.950633)Jl Cijawura
0 1 0 0 0 0 1 0 0 0
1 0 1 0 0 0 0 0 0 0
0 1 0 1 0 0 0 0 0 0
0 0 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 0 0 0
1 0 0 0 0 0 0 1 0 0
0 0 0 0 0 0 1 0 1 0
0 0 0 0 0 0 0 1 0 1
0 0 0 1 0 0 0 0 1 0

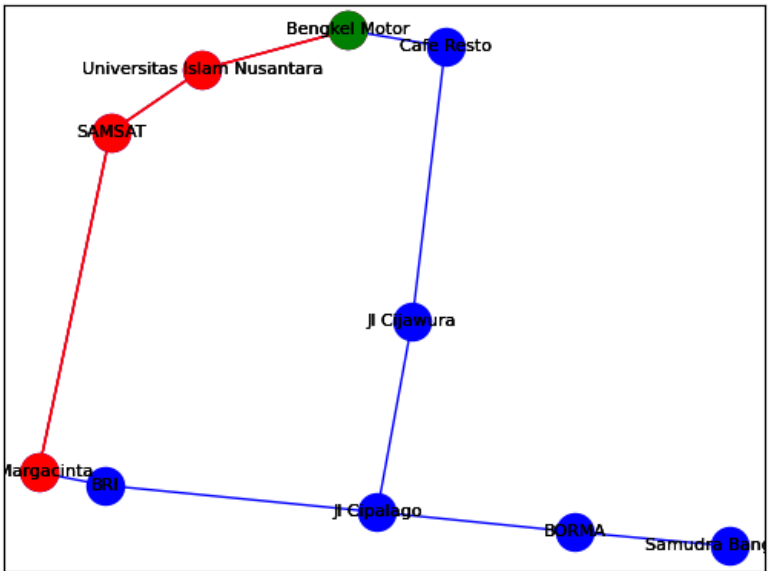
Lokasi awal : Jl Margacinta
Lokasi tujuan : Bengkel Motor

2. *Output :*

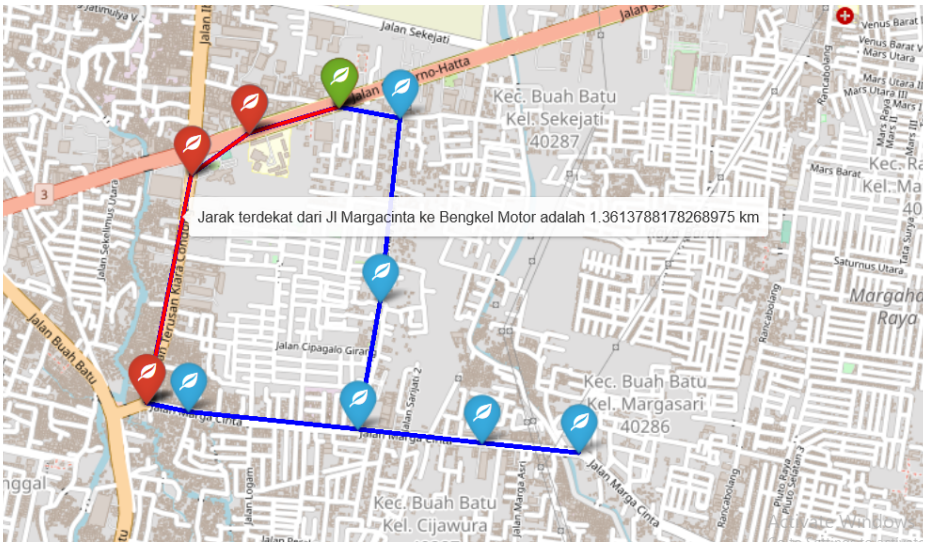
Jarak terdekat: 1.3613788178268975 km
Rute yang ditempuh: Jl Margacinta -> SAMSAT -> Universitas Islam Nusantara -> Bengkel Motor

Hasil output visualisasi :

1. Graf



2. Map/Peta



D. Case 4 : Peta jalan sebuah kawasan di kotamu

1. *Input :*

10
(104.747688,-2.954355)Jupiter Futsal
(104.748348,-2.955044)Palcomtech
(104.748240,-2.955862)Rumah Sakit Hermina Palembang
(104.746038,-2.955786)favehotel Palembang
(104.746828,-2.952563)Masjid Jami' Al Burhan
(104.746985,-2.952955)STIQ Al-Lathifiyyah Palembang
(104.746680,-2.952319)Rumah Saya
(104.746032,-2.951536)Waduk FIF Group Palembang
(104.748520,-2.952824)Pondok Pesantren Tahfizhul Qur'an Putri
(104.750578,-2.954225)Kawasaki Basuki Rahmat
0 1 0 0 0 1 0 0 0 0
1 0 1 1 0 0 0 0 0 1
0 1 0 0 0 0 0 0 0 0
0 1 0 0 0 0 0 0 0 0
0 0 0 0 0 1 1 0 0 0
1 0 0 0 1 0 0 0 0 0
0 0 0 0 1 0 0 1 0 0
0 0 0 0 0 0 1 0 0 0
0 0 0 0 0 0 0 0 0 0
0 1 0 0 0 0 0 0 0 0

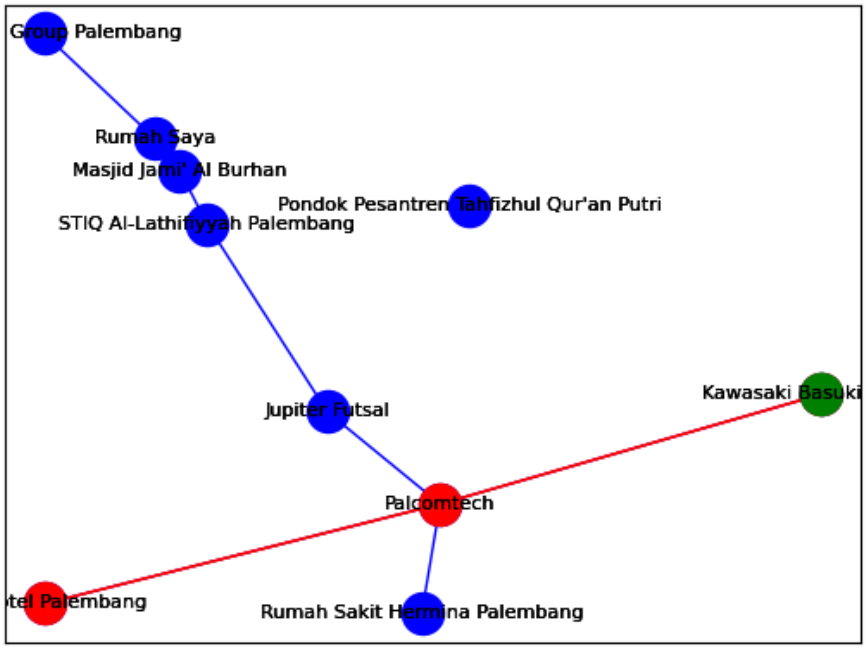
Lokasi awal : favehotel Palembang
Lokasi tujuan : Kawasaki Basuki Rahmat

2. *Output :*

Jarak terdekat: 0.4801883907049338 km
Rute yang ditempuh: favehotel Palembang -> Palcomtech -> Kawasaki Basuki Rahmat

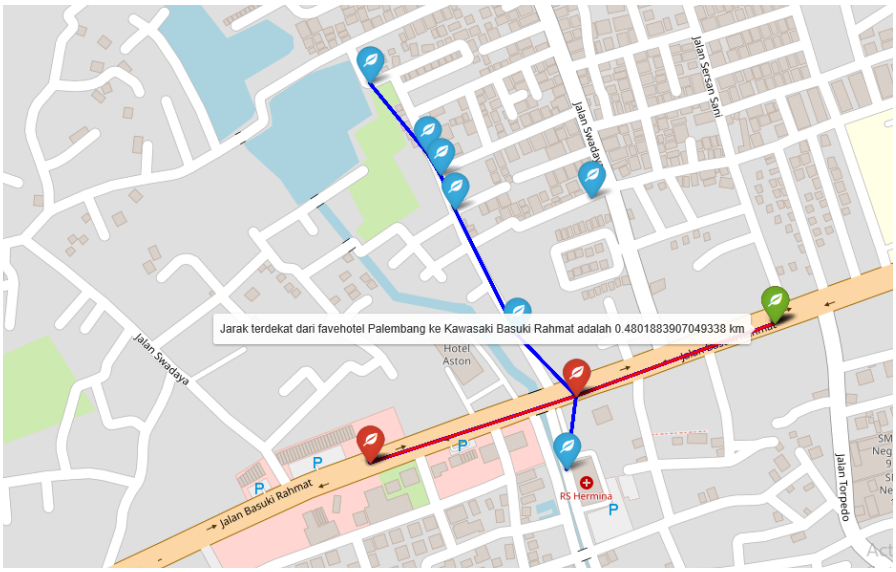
Hasil output visualisasi :

1. Graf



terdekat dari favehotel Palembang ke Kawasaki Basuki Rahmat adalah 0.4801883907

2. Map/Peta



E. Case 5 : Peta Provinsi

1. *Input :*

```
12
(95.359804,5.450791)Bandah Aceh
(98.747829,3.558711)Medan
(101.458298,0.490884)Pekanbaru
(100.377730,-0.307052)Bukit Tinggi
(103.613302,-1.617299)Jambi
(104.768454,-2.996110)Palembang
(105.271765,-5.417047)Bandar Lampung
(106.839804,-6.227680)Jakarta
(107.603329,-6.918100)Bandung
(108.548542,-6.746949)Cirebon
(110.439034,-6.997464)Semarang
(110.818296,-7.572481)Surakarta
0 1 0 0 0 0 0 0 0 0 0 0
1 0 1 0 0 0 0 0 0 0 0 0
0 1 0 1 1 0 0 0 0 0 0 0
0 0 1 0 0 0 0 0 0 0 0 0
0 0 1 0 0 1 0 0 0 0 0 0
0 0 0 0 1 0 1 0 0 0 0 0
0 0 0 0 0 1 0 0 0 0 0 0
0 0 0 0 0 0 0 0 1 1 0 0
0 0 0 0 0 0 0 1 0 1 0 0
0 0 0 0 0 0 0 1 1 0 1 0
0 0 0 0 0 0 0 0 0 1 0 1
0 0 0 0 0 0 0 0 0 0 1 0
```

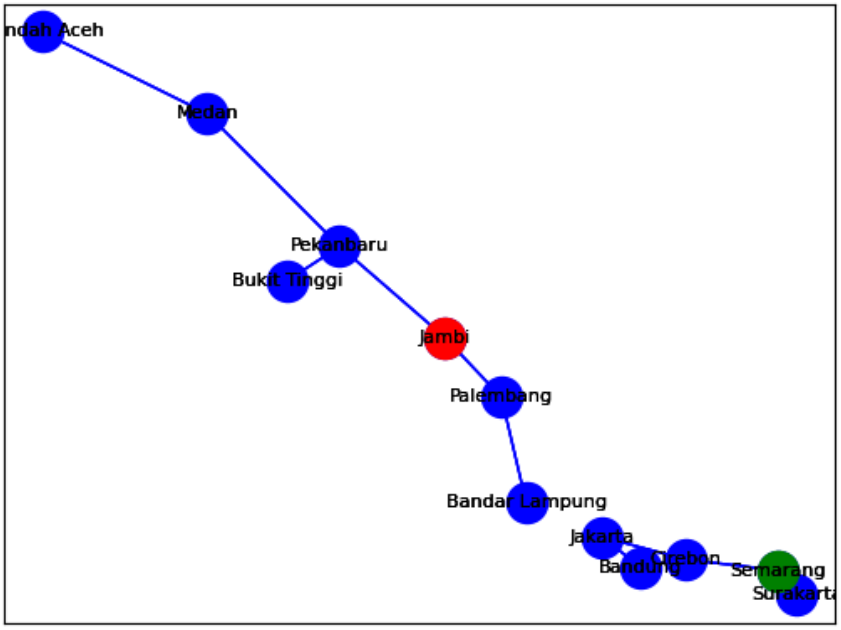
Lokasi awal : Jambi
Lokasi tujuan : Semarang

2. *Output :*

Jambi ke Semarang tidak memiliki jalur.

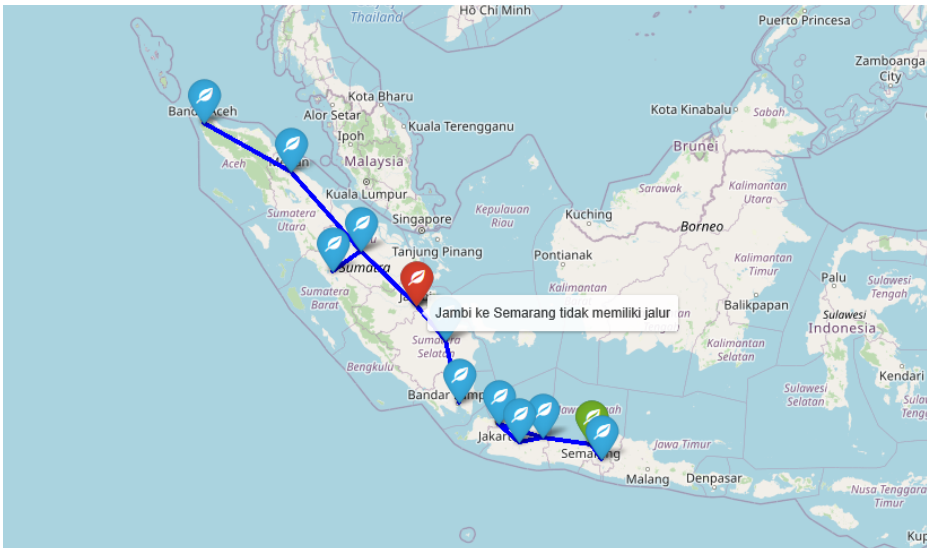
Hasil output visualisasi :

1. Graf



Jambi ke Semarang tidak memiliki jalur

2. Map/Peta



F. Case 6 : Peta OldTrafford

1. *Input :*

11
(-2.275084,53.461008)Masjid E Hidayah
(-2.268837,53.463587)Chorlton Street Post Office
(-2.264685,53.464392)Old Traffod Community Academy
(-2.262467,53.464797)Bright Horizons Trafford Nursery and Preschool
(-2.261317,53.465030)The Rivers Apartments
(-2.263074,53.465104)Jordans Displays
(-2.260117,53.465256)Erskine Street
(-2.261579,53.465871)Glazebrook Apartments
(-2.260193,53.466267)Judah City Of Praise
(-2.260472,53.466757)Christ Family Church
(-2.260579,53.466951)St Lawrence Pharmacy
0 0 0 0 0 0 0 0 0 0 0
0 0 1 0 0 0 0 0 0 0 0
0 1 0 1 0 0 0 0 0 0 0
0 0 1 0 1 0 0 0 0 0 0
0 0 0 1 0 0 1 0 0 0 0
0 0 0 0 0 0 0 0 0 0 0
0 0 0 0 1 0 0 1 0 0 0
0 0 0 0 0 0 1 0 0 0 0
0 0 0 0 0 0 0 0 0 1 0
0 0 0 0 0 0 0 0 1 0 1
0 0 0 0 0 0 0 0 0 1 0

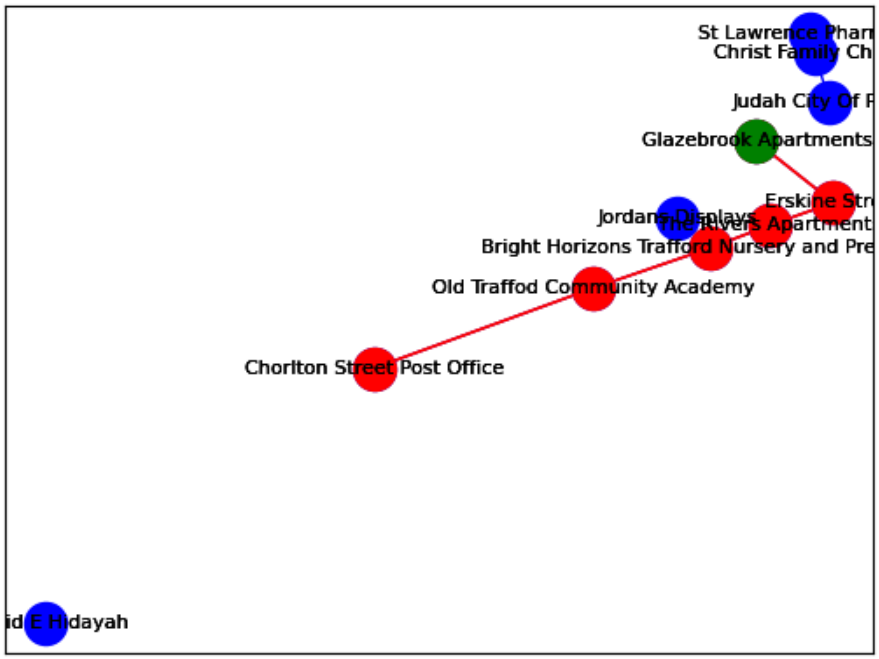
Lokasi awal : Chorlton Street Post Office
Lokasi tujuan : Glazebrook Apartments

2. *Output :*

Jarak terdekat: 1.046453931340319 km
Rute yang ditempuh: Chorlton Street Post Office -> Old Traffod Community Academy -> Bright Horizons Trafford Nursery and Preschool -> The Rivers Apartments -> Erskine Street -> Glazebrook Apartments

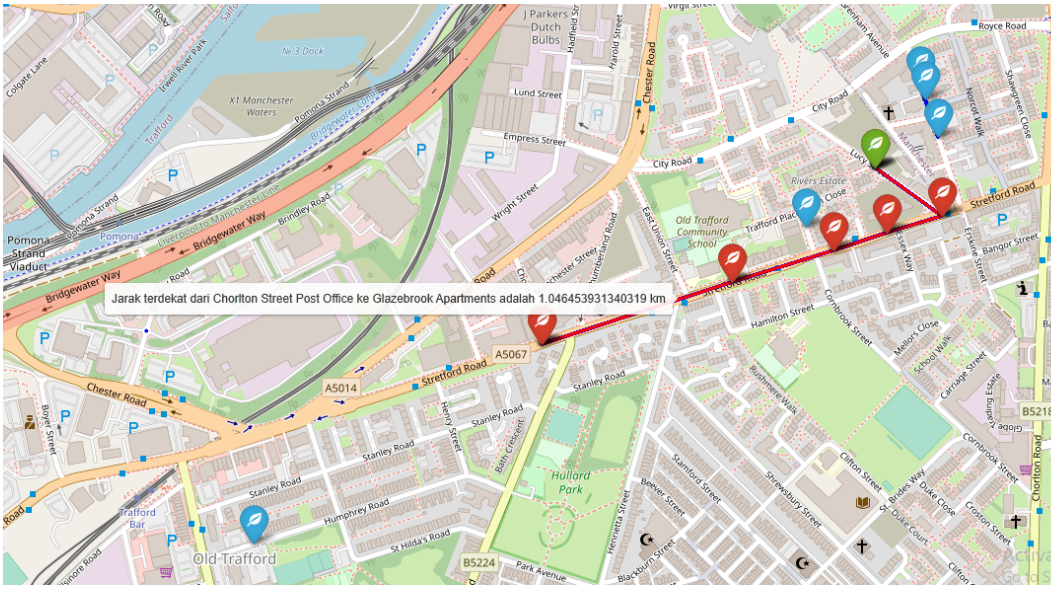
Hasil output visualisasi :

1. Graf



terdekat dari Chorlton Street Post Office ke Glazebrook Apartments adalah 1.046453931340

2. Map/Peta



BAGIAN IV

ALAMAT TEMPAT KODE SUMBER PROGRAM

https://github.com/mgstabrani/shortest-path

REFERENSI

https://python-visualization.github.io/folium/ (diakses pada 5 April 2021).

https://matplotlib.org/ (diakses pada 5 April 2021).

https://www.google.com/maps/ (diakses pada 5 April 2021).

https://www.youtube.com/watch?v=4RnU5qKTfYY&t=692s (diakses pada 5 April 2021).

https://informatika.stei.itb.ac.id/~rinaldi.munir/Stmik/2020-2021/Tugas-Kecil-3-(2021).pdf (diakses pada 31 Maret 2021).

https://informatika.stei.itb.ac.id/~rinaldi.munir/Stmik/2020-2021/Route-Planning-Bagian2-2021.pdf (diakses pada 6 April 2021).

https://networkx.org/documentation/stable/tutorial.html (diakses pada 5 April 2021).

https://media.neliti.com/media/publications/235453-penerapan-algoritma-a-star-menggunakan-g-fa0b1902.pdf (diakses pada 7 April 2021).

LAMPIRAN

Poin	Ya	Tidak
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	✓	