Tugas Kecil 3 IF2211 Strategi Algoritma

Semester II tahun 2020/2021

Nizamixavier Rafif Lutvie (13519085), Pratama Andiko (13519112)

A. Source Code

Berikut adalah source code algoritma A* pada program. (Bahasa: Javascript)

```
      // menghitung jarak antar dua titik menggunakan haversine formula
      function getDistanceFromLatLng(lat1, lng1, lat2, lng2, miles) { // miles opsional
        if (typeof miles === "undefined"){miles=false;}
        function deg2rad(deg){return deg * (Math.PI/180);}
        function square(x){return Math.pow(x, 2);}
        var r=6371; // radius bumi dengan satuan km
        lat1=deg2rad(lat1);
        lat2=deg2rad(lat2);
        var lat_dif=lat2-lat1;
        var lng_dif=deg2rad(lng2-lng1);
        var a=square(Math.sin(lat_dif/2))+Math.cos(lat1)*Math.cos(lat2)*square(Math.sin(lng_dif/2));
        var d=2*r*Math.asin(Math.sqrt(a));
        if (miles){return d * 0.621371;} //menghasilkan miles
        else{return d;} //menghasilkan km
      }


      //penambahan dan pengurutan prioritas array dari f(n) terkecil
```

```
function add(arr,x) {
  arr.push(x);
  var n = arr.length;
  for (var i = 0; i < n-1; i++) {
    for (var j = 0; j < n-i-1; j++) {
      if (arr[j][1] < arr[j+1][1]) {
        var temp = arr[j];
        arr[j] = arr[j+1];
        arr[j+1] = temp;
      }
    }
  }
}


// menghitung total jalur yang sudah dilalui, ekuivalen dengan g(n)
function countPrev(prev,distance,x) {
  var total = 0;
  while(prev[x] != -1) {
    total = total + distance[x][prev[x]];
    x = prev[x];
  }
  return Number(total.toFixed(2));
}
```

```
//pemrosesan input file
const fileList = this.files;
var reader = new FileReader;
reader.readAsText(fileList[0]);
reader.onload = function() {
  var rawLog = reader.result;
  var text1 = rawLog.split("\n");
  var newtext = text1;
  for(var i = 0;i<text1.length;i++) {
    newtext[i] = text1[i].split(" ");
  }

  var n = Number(text1[0]);
  var name = [];
  var matrix = [];
  var coordinate = [];
  var dikunjungi = [];
  var distance = [];
  for (var i = 1; i<=n; i++) {
      name.push(newtext[i][0]);
      coordinate.push([newtext[i][1],newtext[i][2]]);
  }
  for (var i = 0; i<coordinate.length; i++) {
    for (var j = 0; j<coordinate[i].length; j++) {
      coordinate[i][j] = Number(coordinate[i][j]);
    }
```

```
      }
      for (var i = n+1; i <= n*2; i++) {
          matrix.push(newtext[i]);
      }
      for (var i = 0; i<matrix.length; i++) {
        for (var j = 0; j<matrix[i].length; j++) {
          matrix[i][j] = Number(matrix[i][j]);
        }
      }
      for(var i = 0;i<=n;i++) {
        dikunjungi.push(0);
      }
      for(var i = 0;i<n;i++) {
        distance.push([]);
        for(var j = 0;j<n;j++) {
          distance[i].push(0);
        }
      }
      for (var i = 0; i<n; i++) {
        for (var j = 0; j<n; j++) {
          distance[i][j] = Number(getDistanceFromLatLng(coordinate[i][0],
coordinate[i][1], coordinate[j][0], coordinate[j][1]));
        }
      }

      //algoritma A*
        function astar(awal,akhir,name,matrix,distance) {
```

```
        var t1,t2;
        var dikunjungi = [];
        var prio = [];
        var prev = [];
        var found = 0;
        for(var i = 0;i<name.length;i++) {
          if(awal == name[i]) {
            t1 = i;
          }
          if(akhir == name[i]) {
            t2 = i;
          }
        }
        for(var i = 0;i<name.length;i++) {
          dikunjungi.push(0);
        }
        for(var i = 0;i<name.length;i++) {
          prev.push(-1);
        }
        add(prio,[t1,distance[t1][t2]]);
        while(found == 0 && prio.length != 0) {
          for(var i = 0;i<prio.length;i++) {
            prio[i] = prio[i];
          }
          var simpul = prio.pop();
          dikunjungi[simpul[0]] = 1;
```

```
            if(simpul[0] == t2) {
              found = 1;


            }
            else {
              for(var i = 0;i<matrix[simpul[0]].length;i++) {
                if(dikunjungi[i] == 0 && matrix[simpul[0]][i] == 1) {
                  prev[i] = simpul[0];
                  var estimate = Number((countPrev(prev,distance,i) +
distance[i][t2]).toFixed(2));
                  add(prio,[i,estimate]);
                }
              }
            }


          }
          if(found == 1) {
            var result = [];
            var carry = t2;
            var total_distance = countPrev(prev,distance,t2);
            while(carry != -1) {
              result.push(carry);
              carry = prev[carry];
            }
            result.reverse();
            if(result.length == 1) {
              //console.log("Lokasi tujuan sama dengan lokasi awal");
```

```
            outputField.append("Lokasi tujuan sama dengan lokasi awal");
            outputField.append(document.createElement("br"));
          }
          else {
            //mewarna jalur graph terdekat
            polylinePoints = [];
            for (let i = 0; i < result.length; i++) {
              polylinePoints.push(coordinate[result[i]]);
            }
            var polyline = new L.polyline(polylinePoints, {
              color: '#46eb34',
              smoothFactor: 1
            });
            map.addLayer(polyline);

            var answer = "Titik yang dilewati : ";
            answer = answer + name[result[0]];
            for(var i = 1;i<result.length;i++) {
              answer = answer + " -> " + name[result[i]] ;
            }
            //console.log("Jarak yang ditempuh : " + total_distance.toFixed(2)
+ " km");
            //console.log(answer);
            outputField.append("Jarak yang ditempuh : " +
total_distance.toFixed(2) + " km");
            outputField.append(document.createElement("br"));
            outputField.append(answer);
```

```
            outputField.append(document.createElement("br"));
          }
        }
        else {
          //console.log("Tujuan tidak bisa dicapai");
          outputField.append("Tujuan tidak bisa dicapai");
          outputField.append(document.createElement("br"));
        }
      }
```

B. Input dan Screenshot

Berikut adalah graf input dan hasil *screenshot* dari eksekusi program.

1. itb.txt

```
10
A -6.893862 107.608453
B -6.893219 107.610436
C -6.893731 107.612947
D -6.887351 107.613535
E -6.892590 107.610410
F -6.887909915582224 107.60824184460915
G -6.887210742330957 107.6113987513734
H -6.885181843109814 107.61295689769521
I -6.884901243188653 107.61146833695106
J -6.885195015034634 107.61367947079181
0 1 0 0 0 1 0 0 0 0
1 0 1 0 1 0 0 0 0 0
0 1 0 1 0 0 0 0 0 0
0 0 1 0 0 0 1 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 1 0 1 1 0
0 0 0 0 0 0 1 0 1 1
0 0 0 0 0 0 1 1 0 0
0 0 0 1 0 0 0 1 0 0
```

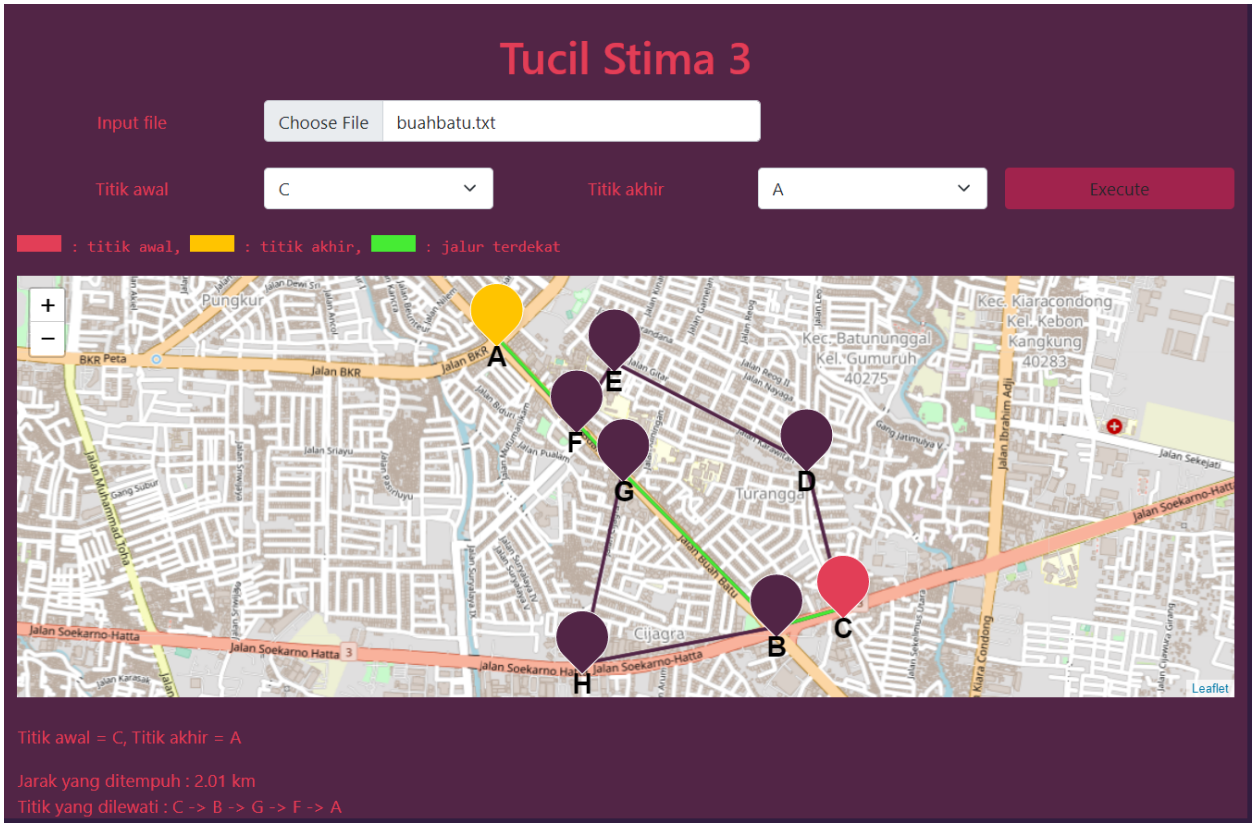
2. alun2.txt

```
16
A -6.92206 107.60399
B -6.92254 107.60762
C -6.92303 107.61187
D -6.92176 107.612
E -6.9201 107.61218
F -6.91971 107.60992
G -6.92145 107.60979
H -6.91956 107.60832
I -6.92123 107.60799
J -6.91896 107.60665
K -6.92023 107.60817
L -6.92001 107.60658
M -6.92104 107.60644
N -6.91834 107.60427
O -6.92081 107.60408
P -6.92276 107.60978
```

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

3. buahbatu.txt

```
8
A -6.936757 107.622691
B -6.947901 107.633486
C -6.947165 107.636042
D -6.941532 107.634648
E -6.937735 107.627215
F -6.940052 107.625765
G -6.941947 107.627582
H -6.949322 107.625973
0 0 0 0 0 1 0 0
0 0 1 0 0 0 1 1
0 1 0 1 0 0 0 0
0 0 1 0 1 0 0 0
0 0 0 1 0 1 0 0
1 0 0 0 1 0 1 0
0 1 0 0 0 1 0 1
0 1 0 0 0 0 1 0
```



4. monas.txt

21
A -6.18186 106.81367
B -6.18175 106.81664
C -6.18493 106.81487
D -6.18345 106.82294
E -6.1826 106.83329
F -6.17873 106.84188
G -6.16388 106.83833
H -6.15928 106.83724
I -6.16252 106.82877
J -6.16002 106.81899
K -6.16755 106.82075
L -6.16791 106.83105
M -6.17104 106.82189
N -6.1764 106.83108
O -6.17228 106.83408
P -6.17282 106.83008
Q -6.17096 106.8334
R -6.19925 106.85389
S -6.18684 106.83612
T -6.18691 106.82306
U -6.19491 106.82302
0 1 1 0 0 0 0 0 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 0 0 0 0 0
1 1 0 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 1 0 0 0 0 0 0 1 0
0 0 0 1 0 1 0 0 0 0 0 0 1 0 0 0 0 1 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 1 0 1 0 0 0 1 0 0 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 0 0 0 0 0 0 1 0 1 0 1 0 0 0 0 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 0 0 0 0 0 0 1 0 1 1 0 0 0 0 0 0 0 0
0 0 0 0 0 0 1 0 1 0 1 0 0 0 0 1 1 0 0 0 0
0 1 0 1 0 0 0 0 0 0 1 0 0 0 0 1 0 0 0 0 0
0 0 0 0 1 0 0 0 0 0 0 0 0 0 0 1 1 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 0 0 0 0 0 0 0 1 1 1 0 0 1 0 0
0 0 0 0 0 0 0 0 0 0 0 0 1 0 0 1 1 0 0 0 0
0 0 0 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 0 0 0 0 0 0 0 0 0 0 0 0 1 0
0 0 0 1 0 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 0 0 0 0 0 0 1 0

5. blitar.txt

```
11
A -8.099045120315456 112.16493522999168
B -8.100840204992812 112.16458117838853
C -8.100340981063543 112.16244613993318
D -8.098853927392977 112.16268217433529
E -8.099013254834688 112.16429149980415
F -8.099066363967909 112.16586863876363
G -8.099140716742642 112.16737067586789
H -8.097600549319546 112.16749942190542
I -8.097419977580683 112.16605102898342
J -8.097165052635042 112.16450607653333
K -8.096888883761814 112.1628645645551
0 1 0 0 1 1 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 1 0 0 0 0 0 0
1 0 0 1 0 0 0 0 0 1 0
```

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

6. kebumen.txt

```
16
A -7.667774 109.653350
B -7.667766 109.652778
C -7.666429 109.652743
D -7.667776 109.655498
E -7.667763 109.650759
F -7.668724 109.650765
```

G -7.669685 109.650829
H -7.669717 109.652774
I -7.668693 109.652774
J -7.670967 109.660889
K -7.673264 109.662716
L -7.675881 109.664863
M -7.675897 109.665656
N -7.676139 109.670385
O -7.676100 109.671964
P -7.675690 109.671975
0 1 0 1 0 0 0 0 0 0 0 0 0 0 0 0
1 0 1 0 1 0 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 0 0 0 0 0 0 0
0 1 0 0 0 1 0 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 0 1 0 1 0 0 0 0 0 0 0 0
0 1 0 0 0 0 1 0 1 0 0 0 0 0 0 0
0 1 0 0 1 0 0 1 0 0 0 0 0 0 0 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 0 1 0 1 0 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 0 1 0 1 0 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 0 1 0 1
0 0 0 0 0 0 0 0 0 0 0 0 0 0 1 0

C. Alamat Program

Berikut adalah alamat menuju repository Drive yang berisi kode program.

https://github.com/pratamaandiko/TucilStima3

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	✓