

**Gebze Technical University
Computer Engineering**

CSE 222 - 2018 Spring

HOMEWORK 7 REPORT

**Burak Özdemir
141044027**

Course Assistant: Fatma Nur Esirci

1 Q1

This part about Question1 in HW7

1.1 Problem Solution Approach

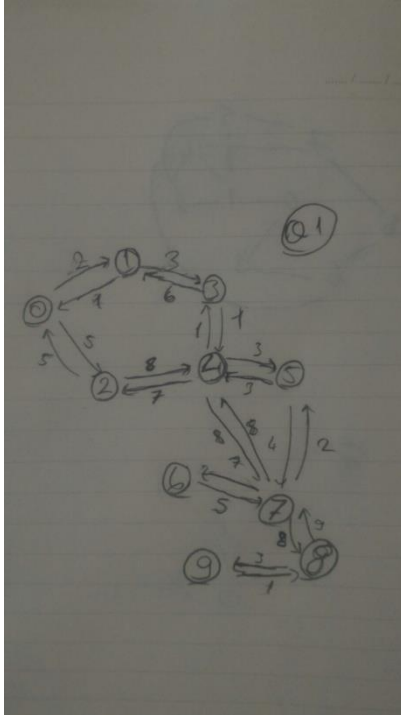
Graf vertex sayısını dosyanın ilk elemanı olarak alınır . Graph objesi oluşturulur ve dosyadan teker teker edge bilgileri alınıp graph objesine insert edilir . Shortest_path metodunda dijkstra algoritması kullanıldı . Specific vertexlerin sınır kabul edildiği pathı 'Result' sınıfının içine vector olarak koyuyor ve distance degerinide 'Result' sınıfının içine koyuyor . Daha sonra bu sınıfı return ediyor .

NOT==> "acycle" ifadesini döngüsel olmayan olarak anladım ve soruyu ona göre yaptım.

1.2 Test Cases

Show that this func results ->

- plot_graph
Test edildi(Basarılı)
- is_undirected
Test edildi(Basarılı)
- is_acyclic_graph
Test edildi(Basarılı)
- shortest_path (use least 3 different label pair)
shortest_path(0,2):Basarılı , shortest_path(0,4):Basarılı , shortest_path(0,7):Basarılı



GRAF ÇİZİMİ(Q1)

```

public class Q1Test {
    public static void main(String []args) throws IOException {
        BufferedReader bf = new BufferedReader(new FileReader("src/q1.txt"));
        Scanner scann = new Scanner(bf);
        MatrixGraph objM = (MatrixGraph) AbstractGraphExtended.createGraph(scann);
        System.out.println("plot_graph(objM)");
        myTesterClass.plot_graph(objM);
        System.out.println("is_undirected(objM):" + myTesterClass.is_undirected(objM));
        System.out.println("is_acyclic_graph(objM):" + myTesterClass.is_acyclic(objM));
        System.out.println("-----");

        myTesterClass.Result res= myTesterClass.shortest_path(objM, v0: 0 , v2: 4);
        System.out.println("shortest_path(0,2):");
        for (int i = 0; i < res.getPath().size(); i++) {
            System.out.print(res.getPath().elementAt(i)+" ");
        }
        System.out.println();
        System.out.println("dist:"+res.getDist());
        System.out.println("-----");

        res= myTesterClass.shortest_path(objM, v0: 0 , v2: 7);
        System.out.println("shortest_path(0,7):");
        for (int i = 0; i < res.getPath().size(); i++) {
            System.out.print(res.getPath().elementAt(i)+" ");
        }
        System.out.println();
        System.out.println("dist:"+res.getDist());
        System.out.println("-----");

        res= myTesterClass.shortest_path(objM, v0: 0 , v2: 11);
        System.out.println("shortest_path(0,11):");
        for (int i = 0; i < res.getPath().size(); i++) {
            System.out.print(res.getPath().elementAt(i)+" ");
        }
        System.out.println();
        System.out.println("dist:"+res.getDist());
    }
}

```

```

[0, 1]: 2.0
[0, 2]: 5.0
-----
[1, 0]: 1.0
[1, 3]: 3.0
-----
[2, 0]: 5.0
[2, 4]: 8.0
-----
[3, 1]: 6.0
[3, 4]: 1.0
-----
[4, 2]: 7.0
[4, 3]: 1.0
[4, 5]: 3.0
[4, 7]: 8.0
-----
[5, 4]: 3.0
[5, 7]: 2.0
-----
[6, 7]: 5.0
-----
[7, 4]: 8.0
[7, 5]: 4.0
[7, 6]: 7.0
[7, 8]: 8.0
-----
[8, 7]: 9.0
[8, 9]: 3.0
-----
[9, 8]: 1.0
-----
is_undirected(objM):false
is_acyclic_graph(objM):false
shortest_path(0,2):0 2
dist:5
-----
shortest_path(0,4):0 1 3 4
dist:6
-----
shortest_path(0,7):0 1 3 4 5 7
dist:11
-----
Process finished with exit code 0

```

Main Test(Q1)

2 Q2

This part about Question2 in HW7

2.1 Problem Solution Approach

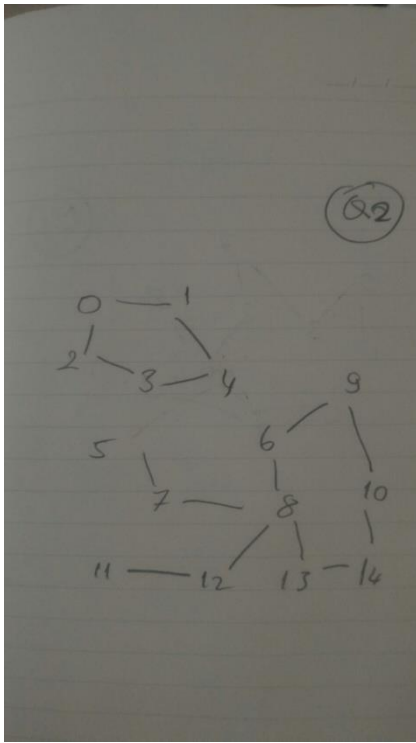
Graf vertex sayısını dosyanın ilk elemanı olarak alınır . Graph objesi oluşturulur ve dosyadan teker teker edge bilgileri alınıp graph objesine insert edilir . Shortest_path metodunda dijkstra algoritması kullanıldı . Specific vertexlerin sınır kabul edildiği pathı ‘Result’ sınıfının içine vector olarak koyuyor ve distance degerinide ‘Result’ sınıfının içine koyuyor . Daha sonra bu sınıfı return ediyor .

NOT==> “acycle” ifadesini döngüsel olmayan olarak anladım ve soruyu ona göre yaptım.

2.2 Test Cases

Show that this func results ->

- plot_graph
Test edildi(Basarılı)
- is_undirected
Test edildi(Basarılı)
- is_acyclic_graph
Test edildi(Basarılı)
- is_connected function (use least 3 different label pair)
is_connected(0,4):Basarılı ,is_connected(4,12):Basarılı , is_connected(9,11):Basarılı



GRAF ÇİZİMİ (Q2)

```

import java.io.BufferedReader;
import java.io.FileReader;
import java.io.IOException;
import java.util.Scanner;

public class Q2Test {
    public static void main(String[] args) throws Exception {
        BufferedReader bf = new BufferedReader(new FileReader("src/q2"));
        Scanner scann = new Scanner(bf);
        MatrixGraph objM = (MatrixGraph) AbstractGraphExtended.createGraph(scann);
        System.out.println("plot_graph(objM)");
        myTesterClass.plot_graph(objM);
        System.out.println("is_undirected(objM):"+myTesterClass.is_undirected(objM));
        System.out.println("is_acyclic_graph(objM):"+myTesterClass.is_acyclic_graph(objM));
        System.out.println("-----");

        System.out.println("is_connected(objM,0,4):"+myTesterClass.is_connected(objM,0,4));
        System.out.println("-----");

        System.out.println("is_connected(objM,4,12):"+myTesterClass.is_connected(objM,4,12));
        System.out.println("-----");

        System.out.println("is_connected(objM,9,11):"+myTesterClass.is_connected(objM,9,11));
        System.out.println("-----");
    }
}

```

```

[ (4, 1): 1.0]
[ (4, 3): 1.0]
[ (5, 7): 1.0]
[ (6, 8): 1.0]
[ (6, 9): 1.0]
[ (7, 5): 1.0]
[ (7, 8): 1.0]
[ (8, 6): 1.0]
[ (8, 7): 1.0]
[ (8, 12): 1.0]
[ (8, 13): 1.0]
[ (9, 6): 1.0]
[ (9, 10): 1.0]
[ (10, 9): 1.0]
[ (10, 14): 1.0]
[ (11, 12): 1.0]
[ (12, 8): 1.0]
[ (12, 11): 1.0]
[ (13, 8): 1.0]
[ (13, 14): 1.0]
[ (14, 10): 1.0]
[ (14, 13): 1.0]
-----
is_undirected(objM):true
is_acyclic_graph(objM):false
-----
is_connected(objM,0,4):true
-----
is_connected(objM,4,12):false
-----
is_connected(objM,9,11):true
-----
Process finished with exit code 0

```

Main Test(Q2)

3 Q3

This part about Question3 in HW7

3.1 Problem Solution Approach

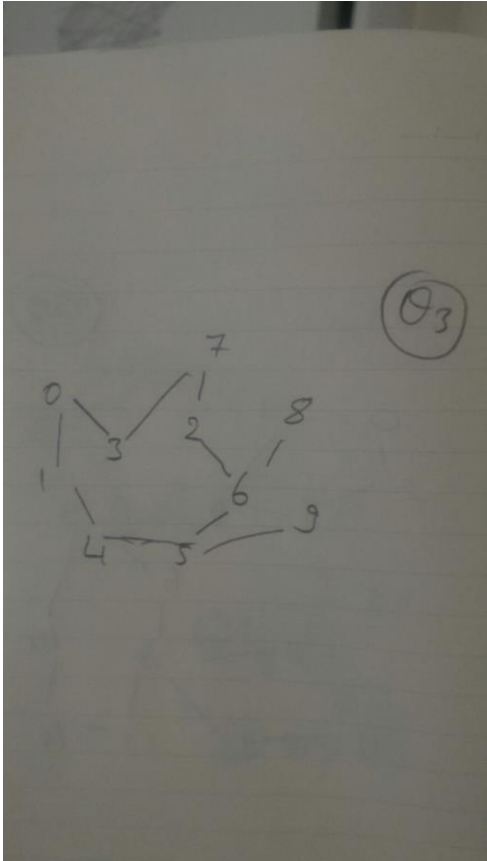
Graf vertex sayısını dosyanın ilk elemanı olarak alınır . Graph objesi oluşturulur ve dosyadan teker teker edge bilgileri alınıp graph objesine insert edilir . Shortest_path metodunda dijkstra algoritması kullanıldı . Specific vertexlerin sınır kabul edildiği pathı 'Result' sınıfının içine vector olarak koyuyor ve distance degerinide 'Result' sınıfının içine koyuyor . Daha sonra bu sınıfı return ediyor .

NOT==> "acycle" ifadesini döngüsel olmayan olarak anladım ve soruyu ona göre yaptım.

3.2 Test Cases

Show that this func results ->

- plot_graph Test edildi(Basarılı)
- is_undirected Test edildi(Basarılı)
- is_acyclic_graph Test edildi(Basarılı)
- DepthFirstSearch (Show that spanning tree)
- BreathFirstSearch (Show that spanning tree)



GRAF ÇİZİMİ(Q3)

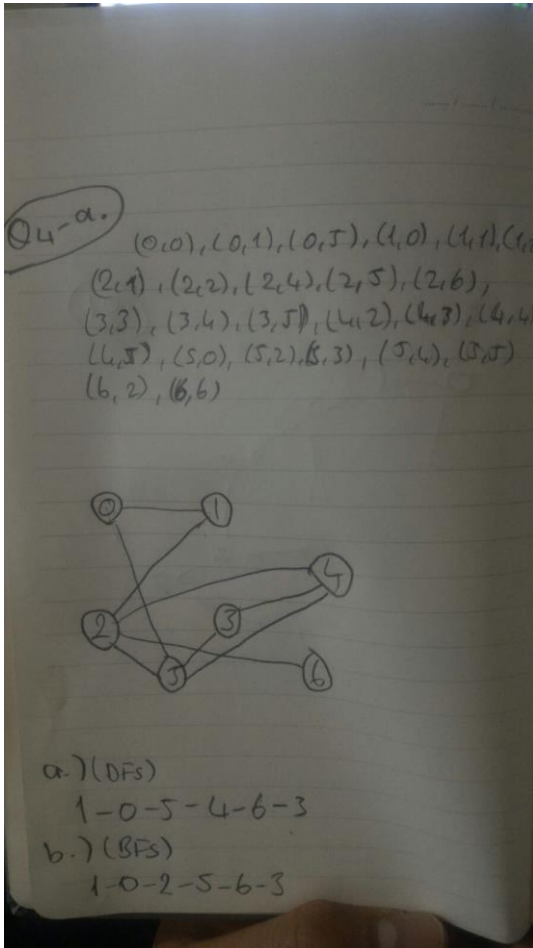
4 Q4

BFS :

- BFS de kök vertexten başlayınca level level diğer vertexlere doğru traverse eder.
- Implement edilirken queue data yapısından yardım alınır .
- Tek aşamada çalışır . Ziyaret edilen koseler queue yapısından çıkartılır .
- DFS ye göre daha yavaştır
- DFS ye göre daha çok memory kullanır.
- Shortest path bulunurken daha faydalıdır .

DFS:

- DFS de vertexlerin sürekli en dip kısmına kadar ilerlenir . Null olunca geri dönülür.
- Implement edilirken stack data yapısından yardım alınır .
- Algoritma 2 aşamada çalışır . İlk aşamada ziyaret edilen koseler stack e konulur . Daha sonra ziyaret edilecek vertex olmadığında teker teker stackten çıkartılarak diğer vertexler işlenir .
- BFS e göre daha hızlıdır .
- BFS e göre daha az memory kullanır .
- Shortest path bulurken faydalı değildir.



QUESTION 4 (a,b)