Gebze Technical University
Computer Engineering

CSE 222 - 2018 Spring

HOMEWORK 7 REPORT

ZEYNEB AKDOĞAN
181041038

Course Assistant: Fatma Nur Esirci

1 Q1

1.1 Problem Solution Approach

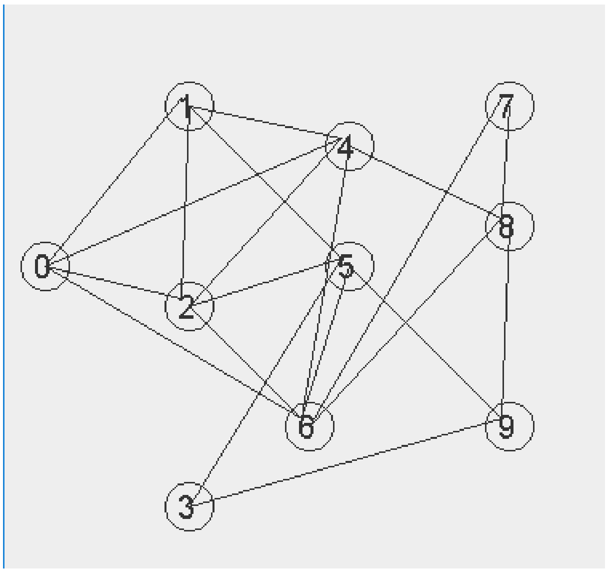
Listgraph, Vertexlerden oluşan ArrayList ve edges lerden oluşan List tanımlandı.

1.2 Test Cases

```
List of Edges
from Source 0
0 - 1 w: 74.0
0 - 2 w: 45.0
0 - 4 w: 45.0
0 - 6 w: 84.0

from Source 1
1 - 2 w: 65.0
1 - 4 w: 86.0
1 - 5 w: 2.0

from Source 2
2 - 4 w: 65.0
2 - 6 w: 90.0
2 - 5 w: 48.0

from Source 3
3 - 5 w: 1.0
3 - 9 w: 68.0

from Source 4
4 - 6 w: 100.0
4 - 8 w: 38.0

from Source 5
5 - 6 w: 83.0
5 - 9 w: 87.0

from Source 6
6 - 7 w: 31.0
6 - 8 w: 96.0

from Source 7
7 - 8 w: 26.0

from Source 8
8 - 9 w: 28.0

from Source 9
```

Show that this func results ->

- plot_graph:



- is_undirected: Graph is directed (okları yapamadım)
- is_acyclic_graph: Graph is acyclic
- shortest_path (use least 3 different label pair):
  Shortest PAth from 0 to 9
- (0 , 4)   w: 45.0
- (4 , 8)   w: 38.0
- (8 , 9)   w: 28.0
- Shortest PAth from 3 to 9
- (3 , 9)   w: 68.0
- Shortest PAth from 3 to 8
- (3 , 5)   w: 1.0
- (5 , 6)   w: 83.0
- (6 , 7)   w: 31.0
- (7 , 8)   w: 26.0

2 Q2

2.1 Problem Solution Approach

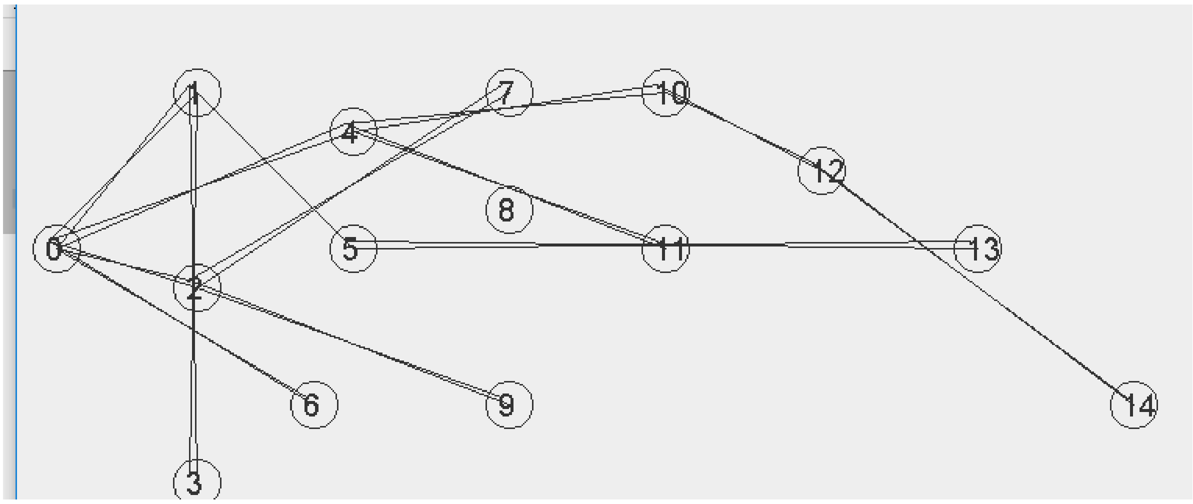
2.2 Test Cases

List of Edges
from Source 0
0 - 1 w: 1.0
0 - 2 w: 1.0
0 - 4 w: 1.0
0 - 6 w: 1.0

```
from Source 1
1 - 0 w: 1.0
1 - 3 w: 1.0
1 - 5 w: 1.0
1 - 5 w: 1.0

from Source 2
2 - 0 w: 1.0
2 - 7 w: 1.0
2 - 9 w: 1.0

from Source 3
3 - 1 w: 1.0

from Source 4
4 - 0 w: 1.0
4 - 10 w: 1.0
4 - 11 w: 1.0

from Source 5
5 - 1 w: 1.0
5 - 1 w: 1.0
5 - 13 w: 1.0

from Source 6
6 - 0 w: 1.0

from Source 7
7 - 2 w: 1.0

from Source 8

from Source 9
9 - 2 w: 1.0

from Source 10
10 - 4 w: 1.0
10 - 12 w: 1.0

from Source 11
11 - 4 w: 1.0

from Source 12
12 - 10 w: 1.0
12 - 14 w: 1.0

from Source 13
13 - 5 w: 1.0

from Source 14
14 - 12 w: 1.0
```

Show that this func results ->

- plot_graph:



- is_undirected: Graph is undirected
- is_acyclic_graph: Graph is acyclic
- is_connected function (use least 3 different label pair)

3 Q3

This part about Question3 in HW7

3.1 Problem Solution Approach

3.2 Test Cases

List of Edges
from Source 0
0 - 1 w: 1.0
0 - 2 w: 1.0
0 - 4 w: 1.0
0 - 6 w: 1.0

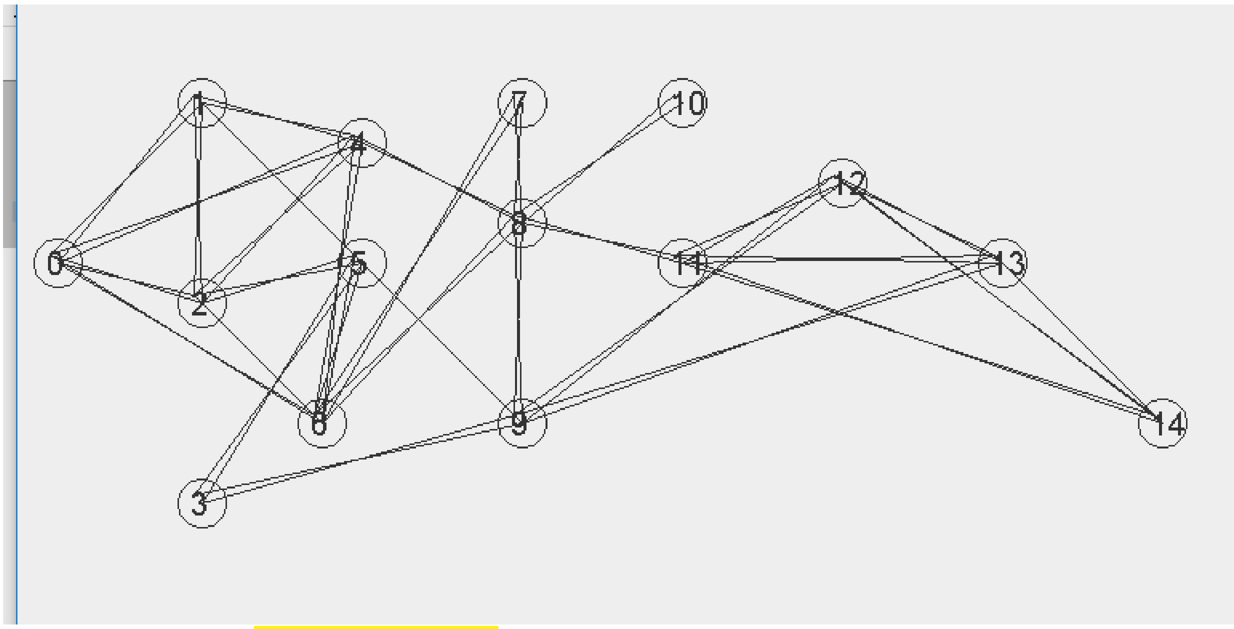
from Source 1
1 - 0 w: 1.0
1 - 2 w: 1.0
1 - 4 w: 1.0
1 - 5 w: 1.0

```
from Source 2
2 - 0 w: 1.0
2 - 1 w: 1.0
2 - 4 w: 1.0
2 - 6 w: 1.0
2 - 5 w: 1.0

from Source 3
3 - 5 w: 1.0
3 - 9 w: 1.0

from Source 4
4 - 0 w: 1.0
4 - 1 w: 1.0
4 - 2 w: 1.0
4 - 6 w: 1.0
4 - 8 w: 1.0

from Source 5
5 - 1 w: 1.0
5 - 2 w: 1.0
5 - 3 w: 1.0
5 - 6 w: 1.0
5 - 9 w: 1.0

from Source 6
6 - 0 w: 1.0
6 - 2 w: 1.0
6 - 4 w: 1.0
6 - 5 w: 1.0
6 - 7 w: 1.0
6 - 8 w: 1.0

from Source 7
7 - 6 w: 1.0
7 - 8 w: 1.0

from Source 8
8 - 4 w: 1.0
8 - 6 w: 1.0
8 - 7 w: 1.0
8 - 9 w: 1.0
8 - 10 w: 1.0
8 - 11 w: 1.0

from Source 9
9 - 3 w: 1.0
9 - 5 w: 1.0
9 - 8 w: 1.0
9 - 12 w: 1.0
9 - 13 w: 1.0
```

```
from Source 10
10 - 8 w: 1.0

from Source 11
11 - 8 w: 1.0
11 - 14 w: 1.0
11 - 13 w: 1.0
11 - 12 w: 1.0

from Source 12
12 - 9 w: 1.0
12 - 11 w: 1.0
12 - 14 w: 1.0
12 - 13 w: 1.0

from Source 13
13 - 9 w: 1.0
13 - 11 w: 1.0
13 - 12 w: 1.0
13 - 14 w: 1.0

from Source 14
14 - 11 w: 1.0
14 - 12 w: 1.0
14 - 13 w: 1.0
```

Show that this func results ->

- plot_grap:



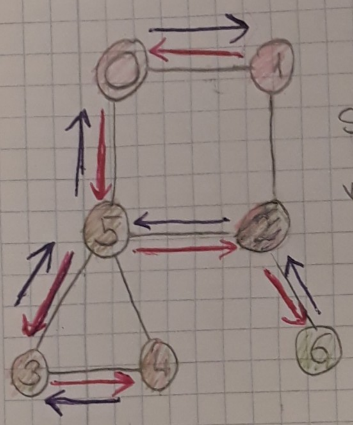
- is_undirected: Graph is undirected
- is_acyclic_graph: Graph is cyclic
- DepthFirstSearch (Show that spanning tree) gösteremedim.
- BreathFirstSearch (Show that spanning tree) gösteremedim.

4 Q4

If you used the handwriting, add this part 1 page pdf include answer of Q4.

DFS'nin implementi daha kolaydır, Search'de stack kullanılır,
BFS aramada queu kullanır.

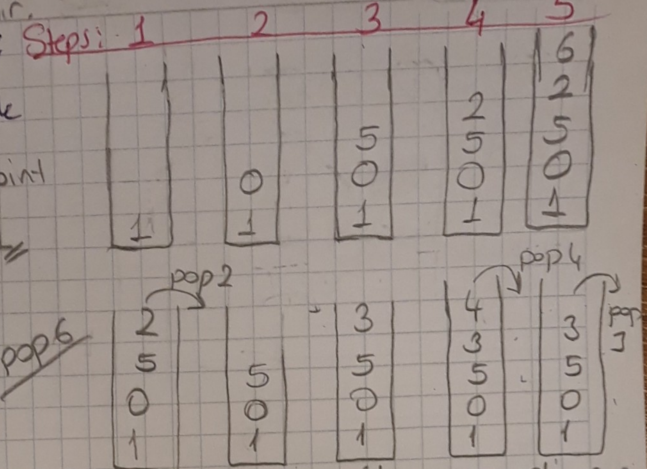
DFS: Steps: 1 2 3 4 5



Stack

Start point

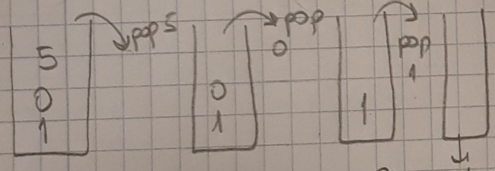
vertex 1



6'dan sonra gidilebilecek yeni vertex olmadığından 6'yı pop ediyoruz.

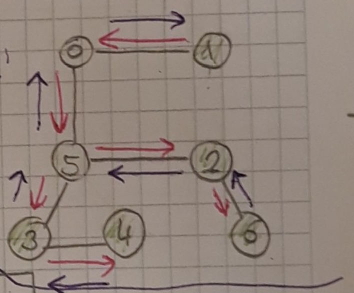
2 den " " " " " " 2'yi " "

4 " " " " " " " " 4'ü " "
3 " " " " " " " " 3'ü " "
5 " " " " " " " " 5'i " "
0 " " " " " " " " 0'ı " "
1 " " " " " " " " 1'i " "

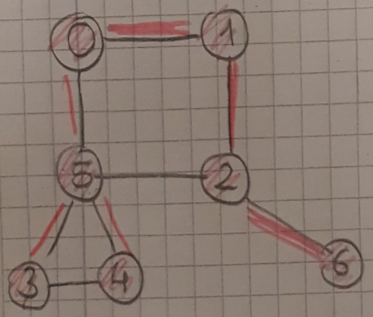


Stack boşaldı

DFS ~~baam~~ ltree:



BFS:



Queu:

1						
	0	2				
		2	5			
			5	6		
				6	3	4
					3	4
						4

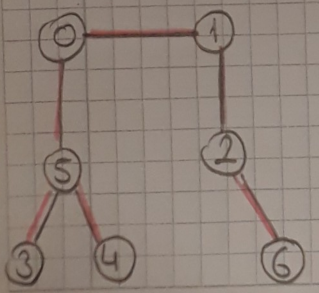
1'den ziyaret edilmemiş komşulara git ve 1'i sil, 0'dan ziyaret edilmemiş komşulara git ve 0'ı sil, 2'den.

5 dan yeni bir komşuya gidilmediğinden 5'i sil
6'yı sil
3'den gidilecek yeni komşu yok, 3'ü sil

4'den gidilecek yeni, ziyaret edilmemiş komşu olmadığından 4'ü sil

→ Queu boşaldı

Final Result:



BFS en yakın komşuları ziyaret etmeyi hedeflerken, |
DFS sürekli olarak komşunun komşusuna gitmeyi amaçlamaktadır.