Gebze Technical University
Computer Engineering

CSE 222 - 2018 Spring

HOMEWORK 7 REPORT

MESUT BUNALDI
111044077

Course Assistant: Fatma Nur Esirci

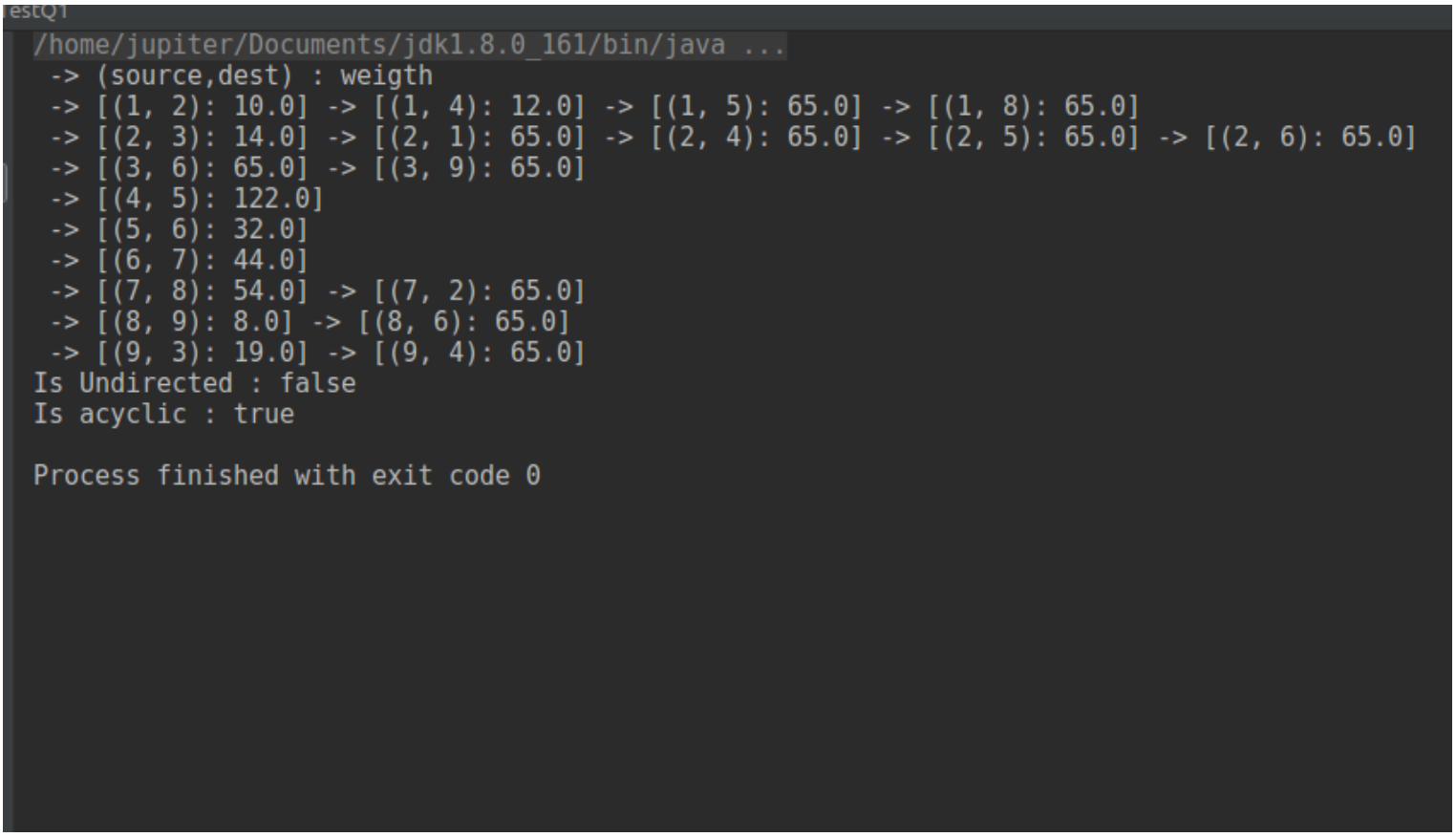
1 Q1

This part about Question1 in HW7

1.1 Problem Solution Approach

Acyclic ve directed Vertex = 10 , Edge=20 olan ListGraph objesi oluşturularak içerisine ağırlıkları rasgele olan edge objeleri eklenmiştir. Bu ListGraph objesi üzerinde birinci kısımda istenen kontrol fonksiyonları çalıştırılmıştır.

1.2 Test Cases



2 Q2

This part about Question2 in HW7

2.1 Problem Solution Approach

Acyclic ve undirected Vertex = 15 olan ListGraph objesi oluşturularak içerisine ağırlıkları olmayan edge objeleri eklenmiştir. Bu ListGraph objesi üzerinde ikinci kısımda istenen kontrol fonksiyonları çalıştırılmıştır.

2.2 Test Cases

```
/home/jupiter/Documents/jdk1.8.0_161/bin/java ...
 -> (source,dest) : weigth
 -> [(0, 1): 1.0] -> [(0, 1): 1.0]
 -> [(1, 0): 1.0] -> [(1, 0): 1.0] -> [(1, 4): 1.0] -> [(1, 5): 1.0] -> [(1, 6): 1.0] -> [(1, 8): 1.0]
 -> [(3, 6): 1.0]
 -> [(4, 1): 1.0]
 -> [(5, 1): 1.0]
 -> [(6, 3): 1.0] -> [(6, 1): 1.0]
 -> [(7, 9): 1.0] -> [(7, 9): 1.0]
 -> [(8, 1): 1.0]
 -> [(9, 7): 1.0] -> [(9, 7): 1.0] -> [(9, 10): 1.0] -> [(9, 11): 1.0] -> [(9, 12): 1.0] -> [(9, 13): 1.0]
 -> [(10, 9): 1.0]
 -> [(11, 9): 1.0]
 -> [(12, 9): 1.0]
 -> [(13, 9): 1.0] -> [(13, 14): 1.0]
 -> [(14, 13): 1.0]
Is Undirected : true
Is acyclic : true

Process finished with exit code 0
```

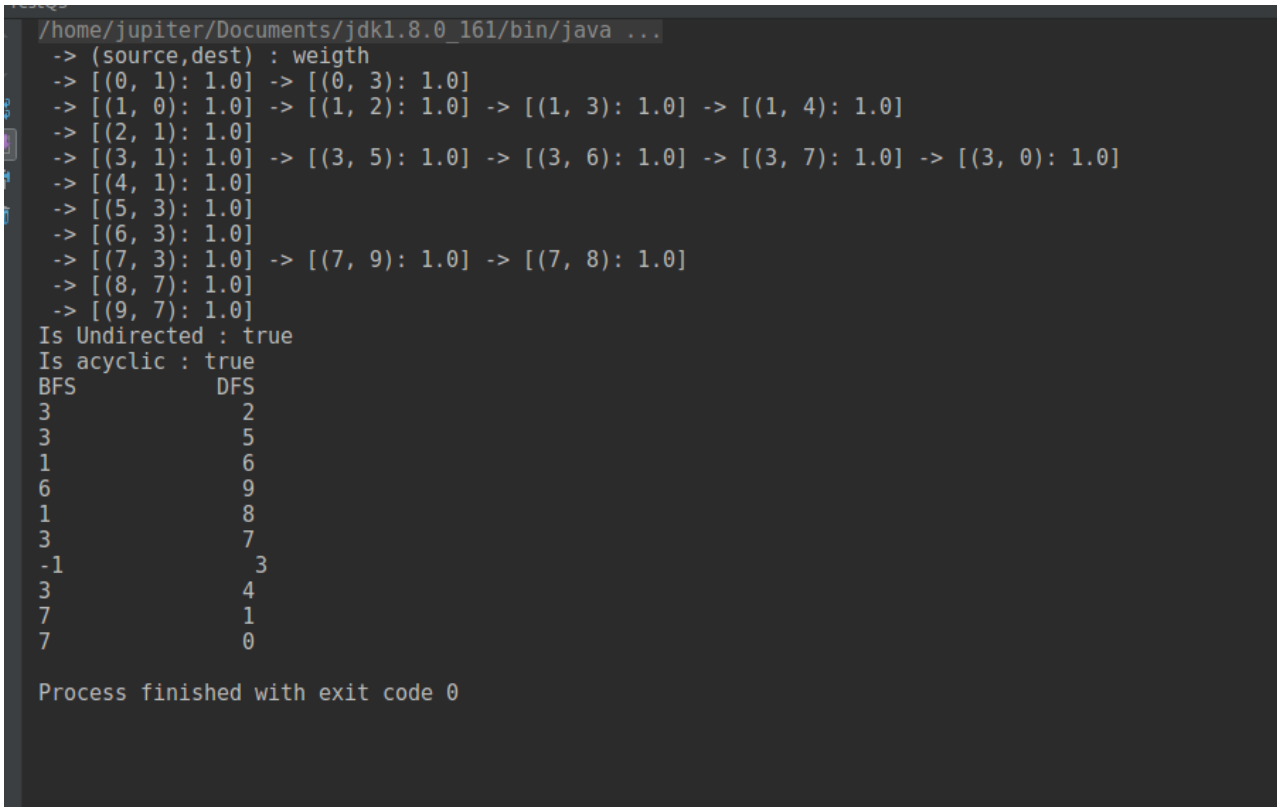
3 Q3

This part about Question3 in HW7

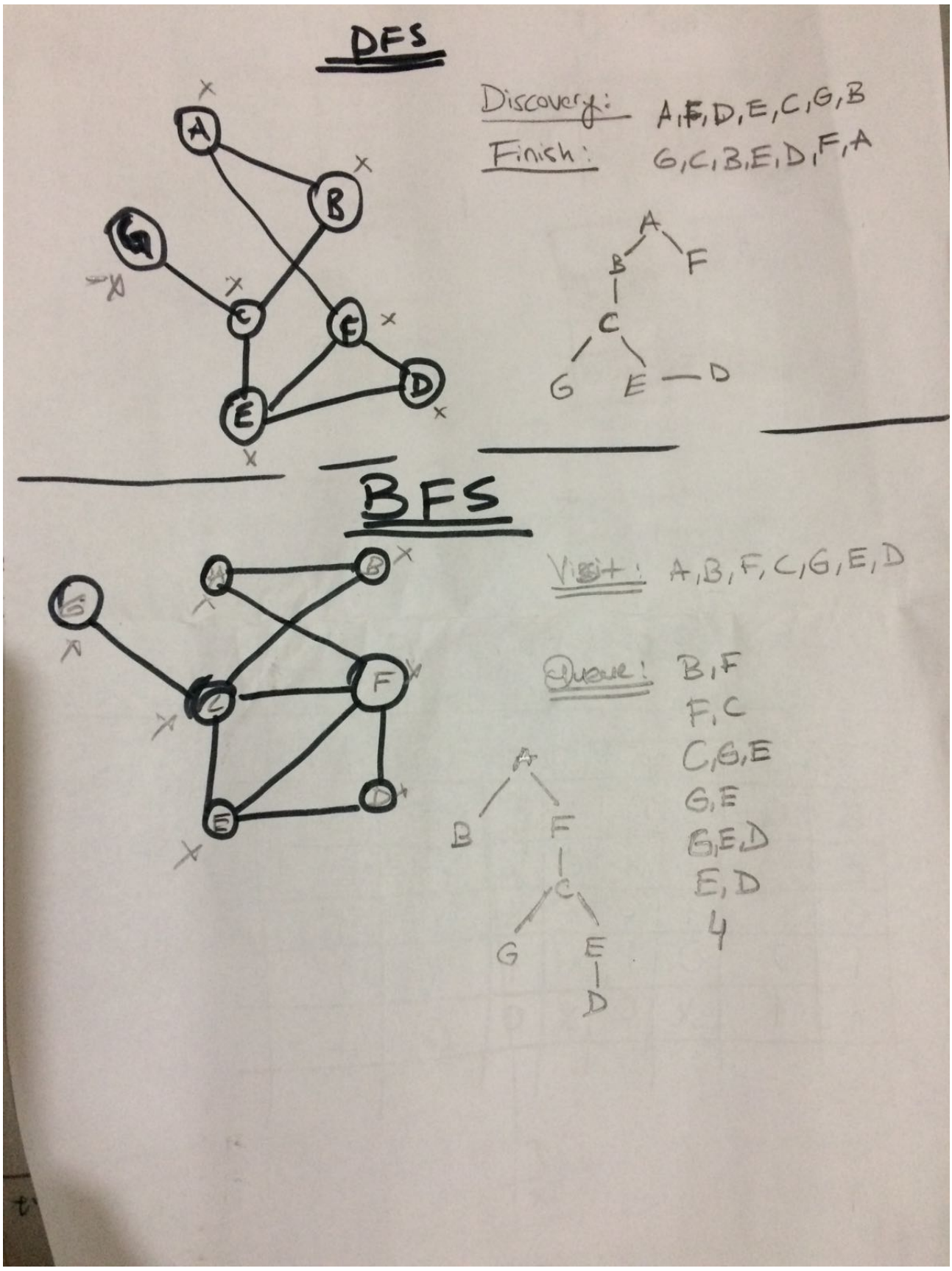
3.1 Problem Solution Approach

Cyclic ve undirected Vertex = 10 ,olan ListGraph objesi oluşturularak içerisine ağırlıkları olmayan edge objeleri eklenmiştir. Bu ListGraph objesi üzerinde üçüncü kısımda istenen fonksiyonlar çalıştırılmıştır.

3.2 Test Cases



4 Q4



DFS ve BFS Arasındaki Farklar :

1- DFS gidebildiği kadar gidip daha sonra geri dönerek Graph üzerinde dolaşmayı hedefler. Yanı Neighbour to Neighbour. BFS ise verilen node üzerinden başlayarak bu nodun tüm komşuları daha sonra yazılan komşuların komşuarı olarak dolaşmayı hedefleyen bir algoritmadır.

2-İkisi de sonuç olarak Spanning Ağacı verir

3- DFS stack yapısı , BFS ise queue yapısını kullanır. Yani sonuç sıralamasını belirlemek için DFS kullanışlıdır.

4-DFS Kullanım Alanları : Topolojikal sıralama, strongly connected graph ,acyclic graph, two edge connected graph , solution of puzzles : maze.

BFS Kullanım Alanları : Finding all connected component in graph shortest path, examine bi pertile graph connectivity.