

**Gebze Technical University
Computer Engineering**

CSE 222 - 2019 Spring

HOMEWORK 8 REPORT

**STUDENT NAME Sıla Bengisu Yüksekli
STUDENT NUMBER 1801042877**

Course Assistant: Doç. Dr. Erchan APTOULA

1 INTRODUCTION

1.1 Problem Definition

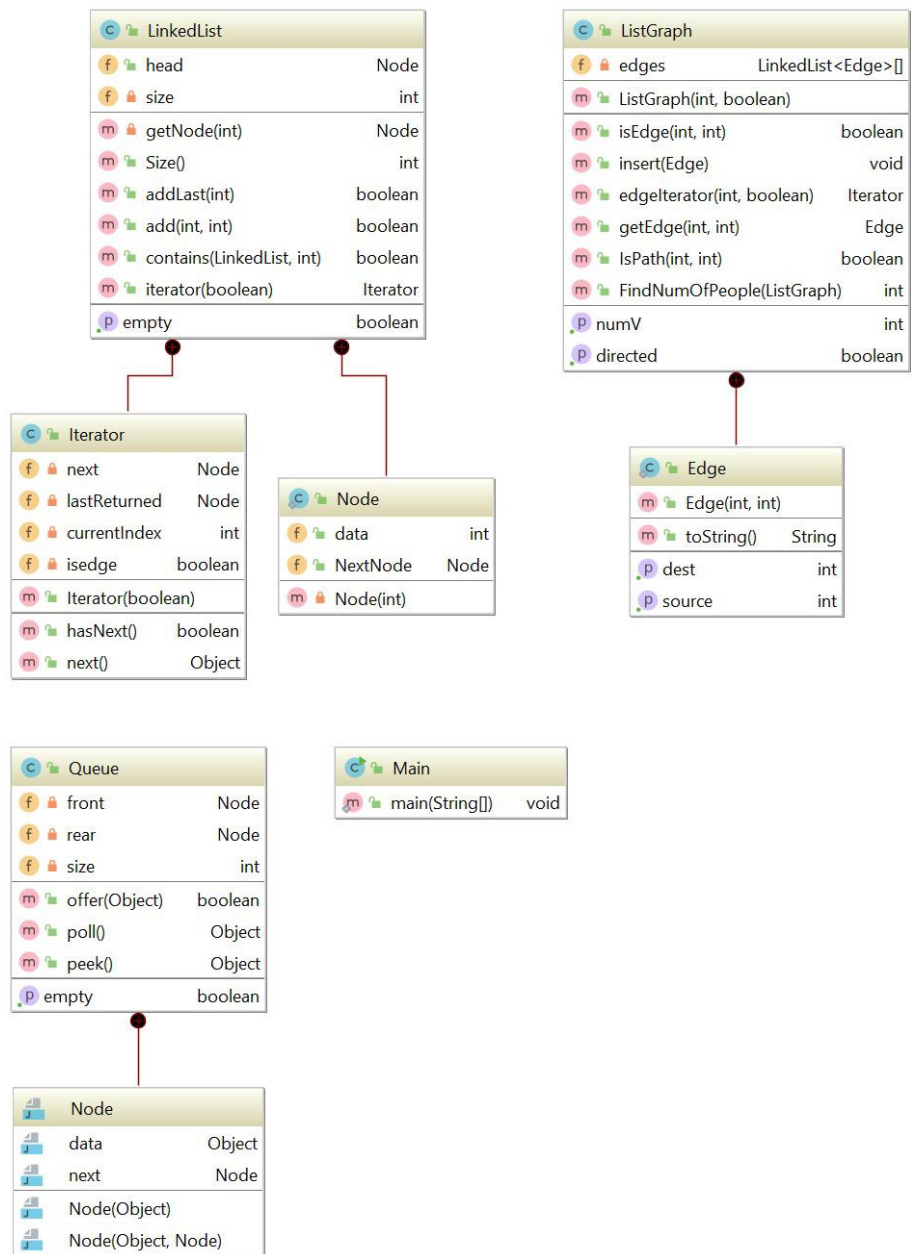
In this problem, I must find the number of the people who is considered popular by every other person, but the popularity is transitive. For instance if X thinks Y is popular and Y thinks Z is popular, then X also thinks Z is popular. So in this problem there are two situations for popularity: first case is direct and second case is indirect. For both cases, I must find person/people who is/are considered popular in direct way or indirect way and print the number.

1.2 System Requirements

You can run this program everywhere which has java virtual machine. But if you want to use IntelliJ like me, maybe you need some requirements like minimum 1 GB RAM, 300 MB hard disk space, minimum 1 GB for caches, 1024x768 minimum screen resolutionSystem requirements.

2 METHOD

2.1 Class Diagrams



2.2 Use Case Diagrams

The user can open the file in IntelliJ. First click the File button which is located at the left-up edge and chose open. After some window opened, choose the project. Then setup the SDK. Run the project from the green button which is located at right-up edge, but if user want to run in some other place, if there is java virtual machine, he/she can do.

2.3 Problem Solution Approach

In this problem some input file is given and I have to read the numbers from this file and I must represent this numbers as a graph. This graph actually represents popularity relation and I must find the number of people/person who are/is considered popular by every other person. I used Adjacency List as a data structure to implement Graph, because I think, LinkedList structure for this problem is more useful due to usage of iterator and simpler to understand. First of all, I implemented my own graph and LinkedList class. After I had tested all the functionality of my methods, I implemented a new method called IsPath to find the indirect way between vertexes. While I was implementing this method, I searched about BreadthFirstSearch, because I used this approach to find if there is a path in indirect way. I implemented a Queue structure which is used in IsPath method. I needed that structure because algorithm is like that: a start vertex(source) is given, add it in queue, search the neighbors of this vertex, if you cant find the expected value, remove it from the queue and do the same things for first neighbor of the given vertex and so on. Then I implemented a method called FindNumOfPeople to find number of the people. In this method, Firstly I researched if there is a edge between other numbers and a given number in direct way. If not, I looked for if there is a indirect way.I did this for every number. Due to this design, I accomplished to solve this problem.

Time complexity of isEdge method is $O(V)$

Time complexity of IsPath method is $O(V+E)$

Time complexity of insert method is $O(1)$

Time complexity of getEdge method is $O(E)$

Space complexity of Graph is $O(V+E)$

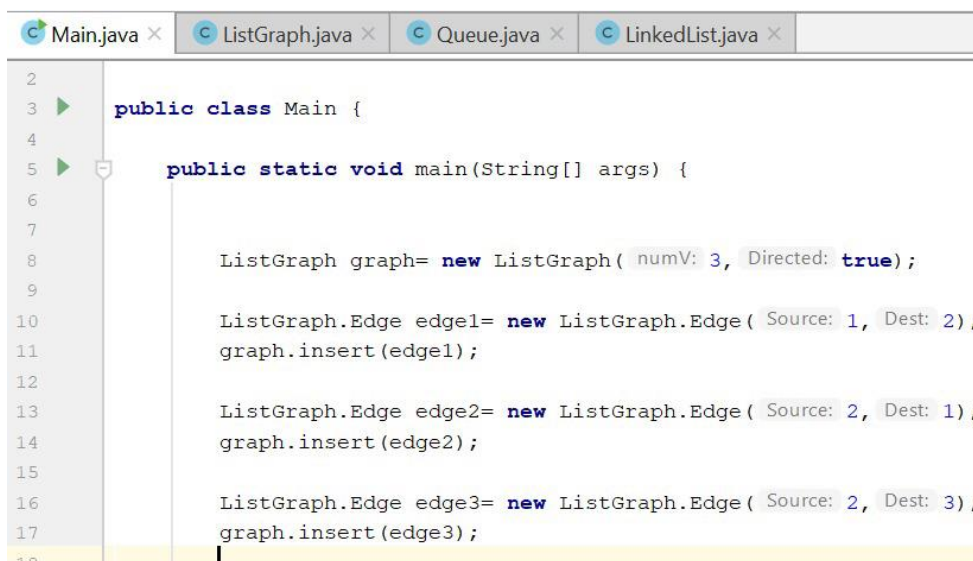
3 RESULT

3.1 Test Cases

First, I created some arbitrary edge objects to test graph and LinkedList structure. I inserted these objects in the graph and tested all the methods with different values in main repeatedly as you can see in the screenshots. After I had implemented `IsPath` method, I checked if my Queue structure works fine. I tested all the methods of Queue class by using some integer values. After the small parts, I tested my program by checking the result of compilation. After the compile, I look the results and checked on the paper, then I compare with the result that was found by computer. So I could test whether the program is working like that. By this methods, I decided to my program works successfully.

3.2 Running Results

Test-1



```
1  public class Main {
2
3  public static void main(String[] args) {
4
5      ListGraph graph= new ListGraph( numV: 3, Directed: true);
6
7      ListGraph.Edge edge1= new ListGraph.Edge( Source: 1, Dest: 2);
8      graph.insert(edge1);
9
10     ListGraph.Edge edge2= new ListGraph.Edge( Source: 2, Dest: 1);
11     graph.insert(edge2);
12
13     ListGraph.Edge edge3= new ListGraph.Edge( Source: 2, Dest: 3);
14     graph.insert(edge3);
15
16
17
18 }
```

Result-1

```
"C:\Program Files\Java\jdk-11.0.2\bin\java.exe" "-javaagent:C:\Program Files\JetBrains\IntelliJ IDEA 2018.3.4\lib\idea_rt.jar=57312:C:\Program Files\JetBrains\
1
Process finished with exit code 0
```

Test-2

```
Main.java × ListGraph.java × Queue.java × LinkedList.java ×
3  ▶ public class Main {
4
5  ▶  ▶ public static void main(String[] args) {
6
7
8      ListGraph graph= new ListGraph( numV: 4, Directed: true);
9
10     ListGraph.Edge edge1= new ListGraph.Edge( Source: 1, Dest: 2);
11     graph.insert(edge1);
12
13     ListGraph.Edge edge2= new ListGraph.Edge( Source: 2, Dest: 1);
14     graph.insert(edge2);
15
16     ListGraph.Edge edge3= new ListGraph.Edge( Source: 2, Dest: 3);
17     graph.insert(edge3);
18
19     ListGraph.Edge edge4= new ListGraph.Edge( Source: 4, Dest: 3);
20     graph.insert(edge4);
21
```

Result-2

```
Main ×
1  "C:\Program Files\Java\jdk-11.0.2\bin\java.exe" "-javaagent:C:\Program Files\JetBrains\IntelliJ IDEA 2018.3.4\lib\idea_rt.jar=57273:C:\Program Files\JetBrains
Process finished with exit code 0
```

Test-3

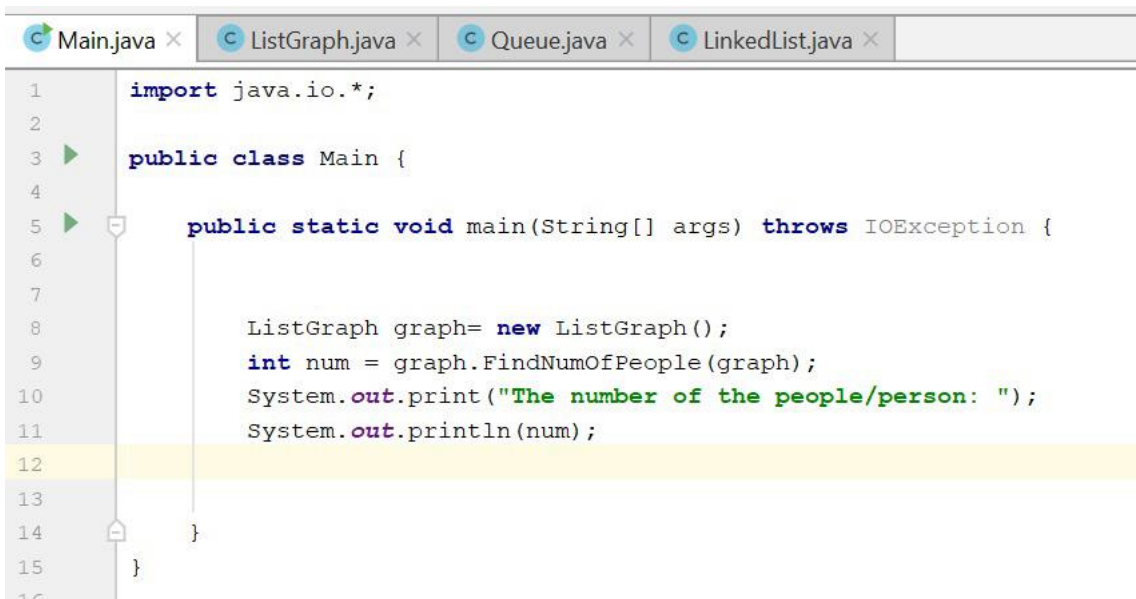
```
Main.java × ListGraph.java × Queue.java × LinkedList.java ×
6
7
8      ListGraph graph= new ListGraph( numV: 5, Directed: true);
9
10     ListGraph.Edge edge1= new ListGraph.Edge( Source: 1, Dest: 2);
11     graph.insert(edge1);
12
13     ListGraph.Edge edge2= new ListGraph.Edge( Source: 2, Dest: 1);
14     graph.insert(edge2);
15
16     ListGraph.Edge edge3= new ListGraph.Edge( Source: 2, Dest: 3);
17     graph.insert(edge3);
18
19     ListGraph.Edge edge4= new ListGraph.Edge( Source: 4, Dest: 5);
20     graph.insert(edge4);
21
22     ListGraph.Edge edge5= new ListGraph.Edge( Source: 3, Dest: 4);
23     graph.insert(edge5);
24
```

Result-3

```
"C:\Program Files\Java\jdk-11.0.2\bin\java.exe" "-javaagent:C:\Program Files\JetBrains\IntelliJ IDEA 2018.3.4\lib\idea_rt.jar=57312:C:\Program Files\JetBrains\1
```

Process finished with exit code 0

Test-4



The screenshot shows the IntelliJ IDEA code editor with four tabs: Main.java, ListGraph.java, Queue.java, and LinkedList.java. The Main.java tab is active, displaying the following code:

```
1  import java.io.*;
2
3  public class Main {
4
5      public static void main(String[] args) throws IOException {
6
7
8          ListGraph graph= new ListGraph();
9          int num = graph.FindNumOfPeople(graph);
10         System.out.print("The number of the people/person: ");
11         System.out.println(num);
12
13     }
14 }
15
16 }
```

Result-4

```
"C:\Program Files\Java\jdk-11.0.2\bin\java.exe" "-javaagent:C:\Program Files\JetBrains\IntelliJ IDEA 2018.3.4\lib\idea_rt.jar=60982:C:\Program Files\JetBrains\
The number of the people/person: 1

Process finished with exit code 0
```