Department of Computer Science and Engineering

Course Code: CSE221	**Credits: 1.5**
Course Name: Algorithms	**Semester: Fall’18**

Lab 01

Representing graphs

I. Topic Overview:

There are several ways to represent graphs, each with its advantages and disadvantages. In this lab, the students will familiarize themselves with two of the most common representations of graphs- adjacency list and adjacency matrix. They will also learn different ways in which inputs can be read from a file by using built in classes. There are in total 4 major problems in this lab some of which are subdivided into smaller problems.

II. Lesson Fit:

There is prerequisite to this lab. Students must have a basic idea on the following concepts:

a. Arrays (1D and 2D) and Lists
b. Directed and Undirected Graphs

III. Learning Outcome:

After this lecture, the students will be able to:

a. Use different built in classes
b. Take input from files
c. Represent both directed and undirected graphs using adjacency list and adjacency matrix
d. Calculate in degree and out degree for graphs
e. Learn which representation is more efficient in different scenarios

IV. Anticipated Challenges and Possible Solutions

a. Task 2: File may not be found

   Solutions:

   i. Write the path of the file correctly

   ii. May throw an exception so that needs to be handled

b. Task 4: If the file is read line by line, and then tokenized, students may forget to parse the tokens

   Solutions:

   i. Parse to integer if the vertices are represented via numbers

V. Acceptance and Evaluation

Students will show their progress as they complete each problem. They will be marked according to their class performance. There may be students who might not be able to finish all the tasks, they will submit them later and give a viva to get their performance mark. The mark distribution for the lab will be as follows:

Code: 05

Viva: 05

VI. Activity Detail

a. **Hour: 1**

   Discussion:

   Explain how built in classes from different standard libraries can be utilized without having to reinvent the wheel.

   Problem Task:

   i. Task 1- 2 (Page 3-4)

b. **Hour: 2**

   Discussion:

   Explain how an adjacency list and an adjacency matrix can be obtained from a graph using relevant examples.

Problem Task:

i. Task 3 (Page 4- 5)

c. **Hour: 3**

Discussion:

Check task 3 while the students continue with the rest.

Problem Task:

i. Task 4 (Page 5)

VII. Home tasks

a. Task 3-4

b. Unfinished tasks

Lab 1 Activity List

Task 1

Write a Java program that uses the built-in stack class to do the following:
push 10, push 5, push 6, pop, push 9, push 3, push 2, pop, pop
After each push or pop, print the top of stack.

Sample Output:

```
Pushing 10….
Printing top:
10
Pushing 5…
Printing top:
5
Popping…
Printing top:
10
```

Hint/Help: https://docs.oracle.com/javase/7/docs/api/java/util/Stack.html

Task 2

Write a Java Program that reads from a text file **(input.txt)** and prints it:

Sample Output:

```
Printing file:
6
0 3
0 1
0 5
1 4
1 5
2 4
3 4
```

Hint/Help: https://www.geeksforgeeks.org/different-ways-reading-text-file-java/

Task 3

The text file (input.txt) represents a graph. 1st integer is the number of vertices, the rest of the lines show edges. Write a Java program that reads from the file and does the following:

a. **Creates and Prints** the adjacency matrix [assume graph is undirected]
b. **Creates and Prints** the adjacency matrix [assume graph is directed, so 0 1 mean there is an edge 0 → 1]
c. **Creates and Prints** the adjacency list [assume graph is undirected, use built in linked list]
d. **Creates and Prints** the adjacency list [assume graph is directed, so 0 1 mean there is an edge 0 → 1]
e. Print the **out degree** of each node [for undirected graph]
f. Print the **in and out degree** of each node [for directed graph]

Sample Output:

Undirected Graph…..	Directed Graph…..
Adjacency Matrix :-	Adjacency Matrix :-
0 1 2 3 4 5	0 1 2 3 4 5
0 0 1 0 1 0 1	0 0 1 0 1 0 1
1 1 0 0 0 1 1	1 0 0 0 0 1 1
2 0 0 0 0 1 0	2 0 0 0 0 1 0
3 1 0 0 0 1 0	3 0 0 0 0 1 0
4 0 1 1 0 0 0	4 0 0 0 0 0 0
5 1 1 0 0 0 0	5 0 0 0 0 0 0
Adjacency List: -	Adjacency List: -
0 - - > 1 3 4	0 - - > 1 3 4
1 - - > 0 4 5	1 - - > 0 4 5
2 - - > 4 5	2 - - > 4 5
3 - - >0 4	3 - - >0 4
4 - - > 1 2	4 - - > 1 2
5 - - > 0 1	5 - - > 0 1
Out degree: -	Out/IN degree: -
0 - - > 3	0 - - > 3/0
1 - - > 3	1 - - > 2/0
2 - - > 1	2 - - > 1/0
3 - - >2	3 - - >1/0
4 - - > 2	4 - - > 0/3
5 - - > 2	5 - - > 0/2

Hint/Help: https://www.tutorialspoint.com/How-to-create-an-array-of-linked-lists-in-java

Task 4

Study and complete the “Learning Graph” code. This is an object oriented implementation of Graph. You can try the entire semester to finish it. You have to explain what you did when you submit.