BITS Pilani
Pilani | Dubai | Goa | Hyderabad

Work Integrated Learning Programmes Division
M.Tech (Data Science and Engineering)

(S1-18_DSEABZG519, S1-18_DSEADZG519, S1-18_DSEAHZG519) –
(Data Structures and Algorithms Design)
First Semester, 2018 -19

Assignment 1 – PS2 - [Big Screen Graph] - [Weightage 10 %]

1. Problem Statement

In this Problem, you have to write an application in JAVA that implements the Big Screen Graph.

Assume that you are a managing an entertainment website for which you have to represent information about movies and the actors who played role in this. You are given with a list of N actors and M movies with the following two constraints.

1. At most 2 actors can be associated with a movie
2. A movie can have at most 2 performers in it

Model the following problem as a graph based problem. Clearly state how the vertices and edges can be modelled such that this graph can be used to answer the following queries efficiently.

The queries are:

1. List the number of movies and the actors represented in the graph
2. List the names of the movies in which performer A has acted.
3. List the names of the performers in the movie X
4. Consider the following relation R on the movies
   "*Movie A is related to Movie B if there is at least one actor common in the movies A and B. In this case, we write R(A, B)*".
   Given any two movies A and B, verify if R(A, B)?

5. Consider the following relation T on the movies "
   a. *T (A,B) if R (A , B)*
   (or)
   b. *T (A, B) if there is a movie C such that R(A,C) and also R (C,B).* ".
   Given any two movies A and B, verify if T(A, B)?

6. **Perform an analysis for the questions above and give the running time in terms of input size,n.**

The basic structure of the node v will be:

```
class BSGraph
{

    String ActMov[]=new String[20];
    int edges[][];
    static int size;
```

```
    /*
     *  add all relevant functions to execute the above-mentioned queries
     */
}
```

Functions:

1. **void readActMovfile(String inputfile)**: This function reads the input file containing the name of the movie and its associated actors in one line. The name of the movie and actors can be separated by a predefined delimiter (eg. Dangal / Aamir Khan / Fatima Sana). The function should create relevant edges to indicate the connection of a movie and its actors. Ensure that none of the movies or actors get repeated while creating the vertices of the graph.
2. **void displayActMov():** This function displays the total number of unique movies and actors entered through the input file. It should also list out the movies and actors stored.
3. **void displayMoviesOfActor(String actor):** This function displays all the movies a particular actor has acted in.
4. **void displayActorsOfMovie(String movie):** This function displays all the actors that have performed in a particular movie.
5. **void findMovieRelation(String movA,String movB):** Use one of the traversal techniques to find out if two movies are related to each other with the relation R(A,B) (as defined in the problem statement query no 4.). Display the actor that is common between the two movies.
6. **void findMovieTransRelation(String movA,String movB):** Use one of the traversal techniques to find out if two movies are related to each other with the relation T(A,B) (as defined in the problem statement query no 5). Display the entire relation that links movie A and movie B.
7. Add other functions that are required to perform the above minimum requirement

Note for entering inputs:
You may enter all the movies and actors in the file **input.txt**.
Example :
Dangal / Aamir Khan / Fatima Sana
Sanju / Ranbir Kapoor / Dia Mirza
PK / Aamir Khan / Anushka Sharma
Munna Bhai MBBS / Sanjay Dutt / Arshad Warsi
Zindagi Na Milegi Dobara / Farhan Akhtar / Katrina Kaif

2. Deliverables

a. Graph.java file containing the basic graph definition and associated functions
b. BigScreenGraph.java file containing the public static void main function, Operation menu and switch cases with input requests where relevant.
c. Stack.java or Queue.java depending on which traversal technique is being used in the code.
d. Word document showing a successful run and corresponding output for each of the operations.

e. A .txt file with answers of question no:6(Running times).

3. Instructions

- It is compulsory to make use of the data structure/s mentioned in the problem statement.
- It is compulsory to use JAVA for implementation.
- Ensure that all data structure insert and delete operations throw appropriate messages when their capacity is empty or full.
- For the purposes of testing, you may implement some functions to print the data structures or other test data. But all such functions must be commented before submission.
- Make sure that your read, understand, and follow all the instructions

4. Deadline

- The strict deadline for submission of the assignment is **Jan 5, 2019 eod.**
- Late submissions won't be evaluated.

5. How to submit

- This is a group assignment.
- Each group consists of 8-11 members from each batch. All members of the group will work on the same problem statement.

  Ex: DSAD_[BLR/HYD/DLH]_GROUP[1-5]

  The group details can be found here.
- Each group is further divided into subgroups.

  Ex: [BLR/HYD/DLH]_[B1-B4]_[G1-G5]_SUBGROUP [1]
- Each subgroup consists of 2-4 members.
- If your group has 11 students, there will be 3 subgroups of 3 students each and 1subgroup with 2 students. If your group has 10 students, there will be 2 subgroups of 3 students and 1 subgroup of 4 students.
- Each subgroup has to make one submission (only one, no resubmission) of solutions.
- Each sub group should zip the deliverables into "ASSIGNMENT1_[BLR/HYD/DLH]_[B1-B4]_[G1-G5]_SUBGROUP [1-4].zip" and upload in CANVAS in respective location under ASSIGNMENT Tab.
- Assignment submitted via means other than through CANVAS will not be graded

6. Evaluation

- The assignment carries 10 Marks
- Grading will depend on
  - Fully executable code with all functionality
  - Well-structured and commented code
- Every bug in the functionality will cost you 0.5 mark each.

- Every segmentation fault/memory violation will cost you 1 mark each.
- Source code files which contain compilation errors will get at most 25% of the value of that question.

7. Readings

- **Section 6:** Algorithms Design: Foundations, Analysis and Internet Examples Michael T. Goodrich, Roberto Tamassia, 2006, Wiley (Students Edition)