(S2-18_DSECFZG519)
(Data Structures and Algorithms Design)
Academic Year 2018-2019

Assignment 1 – PS2 - [Big Screen Graph] - [Weightage 13%]

1. Problem Statement

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

1. A movie can have at most 2 actors associated with it.
2. A particular actor can act in more than 2 movies.

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 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. T(A, B) is true if
   a. T(A, B) = R(A , B)
      (or)
   b. 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) exists / is True?

6. Perform an analysis for the questions above and give the running time.

The basic structure of the graph will be:

```
class BSGraph:
    ActMov=[]      #list containing actors and movies
    edges=[[],[]]  #integer matrix indicating edges between actors and movies

     # add all relevant functions to execute the above-mentioned queries
```

Functions:

1. **def readActMovfile(self, inputfile)**: This function reads the input file **inputPS2.txt** containing the name of the movie and its associated actors in one line. The name of the movie and actors should be separated by a slash.
   eg. Kesari / Akshay Kumar / Parineeti Chopra
   The function should create relevant nodes for the movies and actors and 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. **def displayActMov(self):** This function displays the total number (count) of unique movies and actors entered through the input file. It should also list out the movies and actors stored. The output of this function should be pushed into **outputPS2.txt** file. The output format should be as mentioned below.

```
--------Function displayActMov--------
Total no. of movies: 10
Total no. of actors: 15

List of movies:
Dangal
Sanju
PK
Munna Bhai MBBS
Zindagi Na Milegi Dobara
ADHM
Krish
Kesari
Gully Boy
Highway

List of actors:
Aamir Khan
Fatima Sana
Ranbir Kapoor
Dia Mirza
Anushka Sharma
Sanjay Dutt
Arshad Warsi
Farhan Akhtar
Katrina Kaif
Hritik
```

Akshay Kumar
Parineeti Chopra
Alia Bhat
Ranveer Singh
Randeep Hooda
---

3. **def displayMoviesOfActor(self, actor):** This function displays all the movies a particular actor has acted in. The function reads the input actor name from the file **promptsPS2.txt** where the search id is mentioned with the tag as shown below.

   searchActor: Aamir Khan

   searchActor: Sanjay Dutt

   The output of this function should be appended into **outputPS2.txt** file. If an actor is not found, an appropriate message should be output to the file. The output format should be as mentioned below.

   --------Function displayMoviesOfActor--------
   Actor name: Aamir Khan
   List of Movies:
   Dangal (if actor is not found display appropriate message)
   PK
   ---

4. **def displayActorsOfMovie(self, movie):** This function displays all the actors that have performed in a particular movie. The function reads the input movie name from the file **promptsPS2.txt** where the search id is mentioned with the tag as shown below.

   searchMovie: Gully Boy

   searchMovie: PK

   The output of this function should be appended into **outputPS2.txt** file. If a movie is not found, an appropriate message should be output to the file. The output format should be as mentioned below.

   --------Function displayActorsOfMovie --------
   Movie name: Gully Boy
   List of Actors:
   Alia Bhat (if movie is not found, display appropriate message)
   Ranveer Singh
   ---

5. **def findMovieRelation(self, movA, 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.). The function reads the input movie names from the file **promptsPS2.txt** where the search id is mentioned with the tag as shown below.

   RMovies: Dangal : PK

   RMovies: Kesari : Highway

The output of this function should be appended into **outputPS2.txt** file. If a relation is not found, an appropriate message should be output to the file. The output format should be as mentioned below.

--------Function findMovieRelation --------
Movie A: Dangal
Movie B: PK
Related: Yes, Aamir Khan (if no, display appropriate message)
---

6. **def findMovieTransRelation(self, movA, 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). The function reads the input movie names from the file **promptsPS2.txt** where the search id is mentioned with the tag as shown below.

   TMovies: Dangal : ADHM

   TMovies: Sanju : Krish

   Display the entire relation that links movie A and movie B. The output of this function should be appended into **outputPS2.txt** file. If a relation is not found, an appropriate message should be output to the file. The output format should be as mentioned below.

   --------Function findMovieTransRelation --------
   Movie A: Dangal
   Movie B: ADHM
   Related: Yes, Dangal > Aamir Khan > PK > Anushka Sharma > ADHM

   (if no, display appropriate message)

   ---

7. Add other functions that are required to perform the above minimum requirement

2. Sample file formats

Sample Input file

The input file **inputPS2.txt** contains names of the movies and its associated actors in one line. The name of the movie and actors should be separated by a slash (/). There could be cases where only one actor is associated with a movie.

Sample inputPS2.txt

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
ADHM / Ranbir Kapoor / Anushka Sharma
Krish / Hritik
Kesari / Akshay Kumar / Parineeti Chopra

Gully Boy / Alia Bhat / Ranveer Singh
Highway / Alia Bhat / Randeep Hooda

Sample promptsPS2.txt

searchActor: Aamir Khan
searchActor: Sanjay Dutt
searchMovie: Gully Boy
searchMovie: PK
RMovies: Dangal : PK
RMovies: Kesari : Highway
TMovies: Dangal : ADHM
TMovies: Sanju : Krish

Sample outputPS2.txt

--------Function displayActMov--------
Total no. of movies: 10
Total no. of actors: 15

List of movies:
Dangal
Sanju
PK
Munna Bhai MBBS
Zindagi Na Milegi Dobara
ADHM
Krish
Kesari
Gully Boy
Highway

List of actors:
Aamir Khan
Fatima Sana
Ranbir Kapoor
Dia Mirza
Anushka Sharma
Sanjay Dutt
Arshad Warsi
Farhan Akhtar
Katrina Kaif
Hritik
Akshay Kumar
Parineeti Chopra
Alia Bhat
Ranveer Singh
Randeep Hooda

--------Function displayMoviesOfActor--------
Actor name: Aamir Khan
List of Movies:
Dangal (if actor is not found display appropriate message)

PK

--------Function displayActorsOfMovie --------
Movie name: Gully Boy
List of Actors:
Alia Bhat (if movie is not found, display appropriate message)
Ranveer Singh

--------Function findMovieRelation --------
Movie A: Dangal
Movie B: PK
Related: Yes, Aamir Khan (if no, display appropriate message)

--------Function findMovieTransRelation --------
Movie A: Dangal
Movie B: ADHM
Related: Yes, Dangal > Aamir Khan > PK > Anushka Sharma > ADHM

(if no, display appropriate message)

---

3. Deliverables

a. **Zipped PS2_BSG_[Group id].py package folder** containing the basic graph definition, stack / queue class for the traversal and the main procedures and functions required for the program.
b. **inputPS2.txt** file used for testing
c. **promptsPS2.txt** file used for testing
d. **outputPS2.txt** file generated while testing
e. **analysisPS2.txt** file containing the running time analysis for the program.

4. Instructions

a. It is compulsory to make use of the data structure/s mentioned in the problem statement.
b. It is compulsory to use Python for implementation.
c. Ensure that all data structure insert and delete operations throw appropriate messages when their capacity is empty or full.
d. 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.
e. Make sure that your read, understand, and follow all the instructions
f. Ensure that the input, prompt and output file guidelines are adhered to. Deviations from the mentioned formats will not be entertained.
g. Run time analysis is provided in asymptotic notations and not timestamp based runtimes in sec or milliseconds.

5. Deadline

a. The strict deadline for submission of the assignment is 6th July, 2019.

b. The deadline is set for a month from the date of rollout to accommodate for the semester exams. No further extension of the deadline will be entertained.

c. Late submissions will not be evaluated.

6. How to submit

a. This is a group assignment.

b. Each group has to make one submission (only one, no resubmission) of solutions.

c. Each group should zip the deliverables and name the zipped file as below

"ASSIGNMENT1_[BLR/HYD/DLH/PUN/CHE]_[B1/B2/…]_[G1/G2/…].zip"

and upload in CANVAS in respective location under ASSIGNMENT Tab.

d. Assignment submitted via means other than through CANVAS will not be graded.

7. Evaluation

a. The assignment carries 13 Marks.

b. Grading will depend on

   a. Fully executable code with all functionality

   b. Well-structured and commented code

   c. Accuracy of the run time analysis

c. Every bug in the functionality will have negative marking.

d. Source code files which contain compilation errors will get at most 25% of the value of that question.

8. Readings

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