(S1-19_DSECLZG519)
(Data Structures and Algorithms Design)
Academic Year 2019-2020

Assignment 1 – PS7 - [Interpreters] - [Weightage 12%]

1. Problem Statement

In this Problem, you have to write an application in Python 3.7 that maps language interpreters with the languages they can translate.

People from all over India call up the Indian consulate to get their visa and immigration queries clarified. At the consulate there is a requirement to hire interpreters and translators who can cover all languages spoken in India so that they can provide better service to all citizens.

The consulate has received a number of resumes from different interpreters who know a certain set of languages. Our job is to hire the minimum number of candidates that can address all Indian languages.

For the sake of this exercise, let us assume the following limitations.

1. An interpreter can speak at most 5 languages

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 names of candidates and the unique list of languages covered by all of them combined.
2. What is the least number of candidates that need to be hired to cover all languages? List the names of the candidates.
3. Generate a list of candidates who can speak a particular language.
4. If the consulate needs to translate some documents from one language to another, is there one single candidate who can do that? If yes, who?
5. If the consulate needs to translate a document from language A to language B, but there is no single person who knows both languages, can there be an intermediary language C which can be used such that the document is first translated from language A to language C and then from language C to language B? If so, draw the path that enables this translation.

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

The basic structure of the graph will be:

```
class interPretr:
    vertices=[] # list containing interpreters and languages
    edges=[[],[]] # adjacency matrix of edges linking interpreters to languages
```

Functions:

1. **def readApplications(self, inputfile)**: This function reads the input file **inputPS7.txt** containing the name of the candidate and the languages they know separated by a slash.
   eg. Manasa / English / Hindi / Punjabi

   The function should create relevant nodes for the candidates and languages relevant edges to indicate the connection between the candidate and the languages they know. Ensure that the nodes are unique and there are no duplicates.

2. **def showAll(self):** This function displays the total number (count) of unique candidates and languages that have been fed into the system. It should also list out the candidate names and unique list of languages stored. The output of this function should be pushed into **outputPS7.txt** file. The output format should be as mentioned below.

   --------Function showAll--------
   Total no. of candidates: 10
   Total no. of languages: 15

   List of candidates:
   Manasa
   Harish
   Paul
   Amjad
   Parul
   Nisha
   Rohan
   Rahul
   Sanjay
   Surya
   Venkat
   Zubin

   List of languages:
   English
   Hindi

Kannada
Malayalam
Tamil
Punjabi
Gujarati
Bengali
Marathi
Telugu

***Note**: This is only an indicative output and not the actual output of the program.*

3. **def displayHireList(self):** This function displays the minimum number of candidates that need to be hired to cover all the languages that are fed into the system. The function also displays the names of the candidates and the languages they speak individually to the **outputPS7** file. The function is triggered when the following tag is found in the file **promptsPS7.txt** file.

   showMinList

   The output of this function should be appended into **outputPS7.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 displayHireList--------
   No of candidates required to cover all languages: 5
   Manasa / English / Hindi / Punjabi
   Venkat / English / Kannada / Tamil / Telugu
   Paul / Malayalam / Marathi / Tamil / Telugu
   Harish / Punjabi / Gujarati / English / Hindi
   Nisha / Bengali / English / Marathi / Hindi

   ***Note**: This is only an indicative output and not the actual output of the program.*

4. **def displayCandidates(self, movie):** This function displays all the candidates who can speak a particular language. The function reads the input language from the file **promptsPS7.txt** with the tag as shown below.

   searchLanguage: Hindi

   searchLanguage: Tamil

   The output of this function should be appended into **outputPS7.txt** file.

   --------Function displayCandidates --------
   List of Candidates who can speak Hindi:
   Manasa,
   Harish
   Nisha
   Sanjay
   Zubin

   ***Note**: This is only an indicative output and not the actual output of the program.*

5. **def findDirectTranslator(self, langA, langB):** Use one of the traversal techniques to find out if one candidate can directly translate from language A to language B. The function reads the input languages from the file **promptsPS7.txt** with the tag as shown below.

   DirectTranslate: English : Malayalam

   DirectTranslate: Hindi : Gujarati

   The output of this function should be appended into **outputPS7.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 findDirectTranslator --------
   Language A: English
   Language B: Malayalam
   Direct Translator: No.

   --------Function findDirectTranslator --------
   Language A: Hindi
   Language B: Gujarati
   Direct Translator: Yes, Harish can translate.

   ***Note**: This is only an indicative output and not the actual output of the program.*

6. **def findTransRelation(self, langA, langB):** Use one of the traversal techniques to find out if two languages are related to each other as defined in the problem statement query no 5. The function reads the input languages from the file **promptsPS7.txt** where the search id is mentioned with the tag as shown below.

   TransRelation: English : Malayalam

   TransRelation: English : Gujarati

   Display the entire relation that links language A and language B. The output of this function should be appended into **outputPS7.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 findTransRelation --------
   Language A: English
   Language B: Malayalam
   Related: Yes, English > Venkar > Telugu > Paul > Malayalam

   (if no, display appropriate message)

   ***Note**: This is only an indicative output and not the actual output of the program.*

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

2. Sample file formats

Sample Input file

The input file **inputPS7.txt** contains names of the candidates and the languages they speak in one line separated by a slash (/). There could be cases where one candidate can speak only one language.

Sample inputPS7.txt

Manasa / English / Hindi / Punjabi
Venkat / English / Kannada / Tamil / Telugu
Paul / Malayalam / Marathi / Tamil / Telugu
Harish / Punjabi / Gujarati / English / Hindi
Nisha / Bengali / English / Marathi / Hindi
Amjad / English / Hindi / Gujarati
Parul / Hindi / Punjabi
Rohan / Telugu / Tamil / Malayalam
Rahul / Marathi
Sanjay / Bengali / English
Surya / Malayalam / Tamil
Zubin / Hindi / English

Sample promptsPS7.txt

showMinList

searchLanguage: Hindi

searchLanguage: Tamil

DirectTranslate: English : Malayalam

DirectTranslate: Hindi : Gujarati

TransRelation: English : Malayalam

TransRelation: English : Gujarati

Sample outputPS7.txt

***Note**: This is only an indicative output and not the actual output of the program.*

--------Function showAll--------
Total no. of candidates: 10
Total no. of languages: 15

List of candidates:
Manasa
Harish
Paul
Amjad
Parul

Nisha
Rohan
Rahul
Sanjay
Surya
Venkat
Zubin

List of languages:
English
Hindi
Kannada
Malayalam
Tamil
Punjabi
Gujarati
Bengali
Marathi
Telugu

--------Function displayHireList--------
No of candidates required to cover all languages: 5
Manasa / English / Hindi / Punjabi
Venkat / English / Kannada / Tamil / Telugu
Paul / Malayalam / Marathi / Tamil / Telugu
Harish / Punjabi / Gujarati / English / Hindi
Nisha / Bengali / English / Marathi / Hindi

--------Function displayCandidates --------
List of Candidates who can speak Hindi:
Manasa,
Harish
Nisha
Sanjay
Zubin

--------Function findDirectTranslator --------
Language A: English
Language B: Malayalam
Direct Translator: No.

--------Function findDirectTranslator --------
Language A: Hindi
Language B: Gujarati
Direct Translator: Yes, Harish can translate.

--------Function findTransRelation --------

Language A: English
Language B: Malayalam
Related: Yes, English > Venkar > Telugu > Paul > Malayalam

---

3. Deliverables

a. **Zipped A1_PS7_INT_[Group id].py package folder** containing modules and package files for the entire program code and associated functions
b. **inputPS7.txt** file used for testing
c. **promptsPS7.txt** file used for testing
d. **outputPS7.txt** file generated while testing
e. **analysisPS7.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 3.7 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. The input, prompt and output samples shown here are only a representation of the syntax to be used. Actual files used to test the submissions will be different. Hence, do not hard code any values into the code.
h. 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 20th Dec, 2019.
b. 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 12 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)