INFS7450 Social Media Analytics
Project 1 – Fast Computation of User Centrality Measures
Semester 1, 2021

Marks:	15 marks (15%)
Submission Due:	19 Apr 21 16:00 (Brisbane Time)
Deliverables:	See deliverables part
How to submit:	Electronic submission via Blackboard

Goal: This project aims to implement a number of efficient algorithms to compute various centrality measures for user nodes such as Pagerank, Betweenness and Closeness. Students are required to finish this project individually.

Dataset: In this project, you will be working with the publicly available Facebook social network data. The Facebook data has been anonymized by replacing the Facebook-internal ids for each user with a new value. The data contains 4039 nodes, 88234 edges in total. Each line of the data represents an undirected link starting from one node to another.

The dataset is available from UQ blackboard. See /Assessment/INFS7450 Project One.

Tasks:
1. Calculate the Betweenness Centrality for nodes in the Facebook dataset. (**8 marks**)
   Overview: write code to load the Facebook social network data and construct an undirected and unweighted graph. Based on the constructed graph, you are required to write a program to calculate the betweenness centralities for the graph vertices.
   Input: The provided Facebook social network data.
   Output: The top-10 nodes with the highest betweenness centralities.

2. Calculate PageRank Centrality for nodes in the Facebook dataset. (**7 marks**)
   Overview: write code to load the Facebook social network data and construct an undirected and unweighted graph. Based on the constructed graph, you are required to write a program to calculate the PageRank (with $\alpha = 0.85, \beta = 0.15$) centralities for the graph vertices.
   Input: The provided Facebook social network data.
   Output: The top-10 nodes with the highest PageRank centralities.

Requirements:
You may use third-party libraries, such as NetworkX to read, load and manipulate the Facebook network dataset. However, you must write your own code to implement the function of node centrality calculation rather than using the third-part or built-in functions. (You can use any functions in NetworkX other than the functions for centrality calculation.)

Programming Languages:
Python and NetworkX are recommended. However, you have your own choices of preferred programming languages including, but not limited to, Python, MATLAB, Java, C, C++, etc.

Deliverables (!!VERY IMPORTANT):

1. A report (.pdf). See the given appendix for an example template. **Submit your report in the PDF format, not in the word format!**

2. A source code file. For python users, organize your code in this way:

```
"""
here is your code for the tasks


"""
print("here is your code for the tasks")

if __name__ == "__main__":
    """
    here you callback your code to load data, and generate the results

    Make sure I can run your code only via a click.
    """
    pass
```
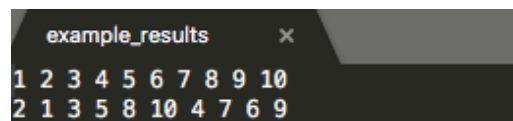
Figure 1. the example of your submitted source code

If you are using other programming languages, please organize your code in one file, and let me know how to run your code only via a simple click.

Only submit the source code in one file, don't submit the data file.

3. A result text file. The file must contain two lines of results. The first line is the top-10 nodes with the highest betweenness centralities. The second line is the top-10 nodes with the highest PageRank centralities. (each node id in each line should be separated by a space.) See the following picture as an example.

```
example_results          ×

1 2 3 4 5 6 7 8 9 10
2 1 3 5 8 10 4 7 6 9
```

Figure 2. the example format of the results

Don't change the format! Your result file will be checked and marked via a script. If you change the format, your submitted results will not be accepted.
Keep in mind the first line is the results of task 1 and the second line is the results of task 2. Don't mix the order.
Don't use other delimiters, only use a space.
Only report the top-10 node ids, don't fill in with node centrality scores!

4. **Name all the submitted files by your student ID**. For example, 41234567.py for the source code, 41234567.txt for your submitted results, and 41234567.pdf for your report.

5. Submit one archive file with your student number as the file name (e.g. 12345678.zip) with all the files mentioned above.

For example:

A student (with id 1234567) can submit his project as follows:

1234567.zip/
-------1234567.py
-------1234567.txt
-------1234567.pdf

Any submitted project which doesn't follow the above guidelines will be desk rejected without marking, which means your will get zero marks for the corresponding parts.

Marking criteria (Total marks: 15):

- Task 1: 8 marks = 3 marks (code) + 3 marks (results) + 2 marks (report)
- Task 2: 7 marks = 2 marks (code) + 3 marks (results) + 2 marks (report)
- Your results should be reproducible and your codes should be readable. If your codes cannot be executed or generate the results as reported, the corresponding marks for the code and results will be deducted.
- We will evaluate your submitted results via calculating the Jaccard Similarity between the submitted results and the ground truth. That means your mark for each task will be calculated by:

  Result Mark = Jaccard Similarity (Submitted Results, Ground Truth) * 3