CS 6220 Data Mining — Assignment 2
DUE DATE: January 25, 2022

YOUR NAME
YOUR LDAP

1 Map Reduce in Spark

Write a Spark program that implements a simple "People You Might Know" social network friendship recommendation algorithm. The key idea is that if two people have a lot of mutual friends, then the system should recommend that they connect with each other.

1.1 Data

- Associated data file is soc-LiveJournal1Adj.txt in q1/data.

- The file contains the adjacency list and has multiple lines in the following format:
  `<User><TAB><Friends>`

- Here, `<User>` is a unique integer ID corresponding to a unique user and `<Friends>` is a comma separated list of unique IDs corresponding to the friends of the user with the unique ID `<User>`. Note that the friendships are mutual (i.e., edges are undirected): if A is friend with B then B is also friend with A. The data provided is consistent with that rule as there is an explicit entry for each side of each edge.

1.2 Algorithm

Let us use a simple algorithm such that, for each user U, the algorithm recommends N = 10 users who are not already friends with U, but have the most number of mutual friends in common with U.

1.3 Output

- The output should contain one line per user in the following format:
  `<User><TAB><Recommendations>`

- Here, `<User>` is a unique ID corresponding to a user and `<Recommendations>` is a comma separated list of unique IDs corresponding to the algorithm's recommendation of people that ¡User¿ might know, ordered in decreasing number of mutual friends.

- **Note**: The exact number of recommendations per user could be less than 10. If a user has less than 10 second-degree friends, output all of them in decreasing order of the number of mutual friends. If a user has no friends, you can provide an empty list of recommendations. If there are recommended users with the same number of mutual friends, then output those user IDs in numerically ascending order.

1.4 Pipeline sketch

Please provide a description of how you used Spark to solve this problem. Don't write more than 3 to 4 sentences for this: we only want a very high-level description of your strategy to tackle this problem.

1.5 Tips

Use Google Colab to use Spark seamlessly, e.g., copy and adapt the setup cells from Colab 0.

- Before submitting a complete application to Spark, you may go line by line, checking the outputs of each step. Command .take(X) should be helpful, if you want to check the first X elements in the RDD.

- For sanity check, your top 10 recommendations for user ID 11 should be:
27552, 7785, 27573, 27574, 27589, 27590, 27600, 27617, 27620, 27667.

- The execution may take a while. Our implementations took around 10 minutes.

2 Submission Instructions

- Commit your code to main in Github, and provide the link. We will snapshot and download it. If you created your code in Colab, you can download your Colab as an iPython Notebook by going to `Download` and selecting `Download .ipynb`.

- Your documentation and code's legibility is part of our grading criterion, so please make sure it's readable.

- Include your writeup, including a pipeline description with a diagram.

- Include in your writeup the recommendations for the users with following user IDs: 924, 8941, 8942, 9019, 9020, 9021, 9022, 9990, 9992, 9993.