FACEBOOK LAB

1. The "No" List

- No A"I" for code, no A"I" even for comments. A"I" refers to artificial "intelligence" tools like chatgpt, microsoft copilot, etc. You need to know how to write your code and the comments help you figure that out.
- No looking at code from your classmates.
- No stackoverflow or chegg or other sites like that that haven't been put out of business by chatgpt.

2. How to collaborate

- List your collaborators in the code.
- You can discuss strategies, but no looking at each others' code.

3. Things you should know

- Autograder will only run after the assignment closes. It is your responsibility your code works with the facebook dataset in /datasets/facebook.
- The assignment is in github in the fb folder. You will need to fill in the missing code. If you do not understand what a function should do based on its description, at least make sure it can transform the input specified in the comments into the output specified in the comments.
- Check the comments in the tri.scala file before asking questions.
- You will definitely need to review the scala slides and maybe videos

4. The Assignment

For this assignment, we will be using the /datasets/facebook dataset. This dataset is an **edge list**. That means each line is a pair of node ids (strings) separated by a single **space**. This means that if there is a line that looks like "123 abc" then there is an edge from node "123" to node "abc" in the graph. We want to find out how many triangles there are in the facebook dataset. A **triangle** is a group of 3 nodes that have edges between them. If this is not clear, look at the toyGraph() function in tri.scala. It returns an edge list (as an RDD) that contains 2 triangles.

- The file tri.scala has a main function that runs everything (once the assignment is complete). You don't have to do anything to the main function.
- There is a countTriangles(edgeList) function that counts the number of triangles in the facebook dataset. To do so, it calls functions that you will need to fill in. The only thing you will need to do eventually is to change the last line (those who are up for the challenge of making it efficient will be making additional changes).
- There is a toyGraph() function that returns an edge list RDD that you can use for testing (i.e., you can run it through countTriangles(). You do not need to change anything here.
- There is a function getSC() that you need to fill in to get a SparkContext.
- There is a function getFB() that you have to fill in. It has to read in the facebook dataset and return an RDD where each entry is a pair of strings (node ids). Once your **entire** code is complete, you should be able to run the resulting RDD through countTriangles() to compute the number of triangles in the facebook dataset.
- There is a function makeRedundant() that you need to fill in. It takes in an edge list and makes sure it is redundant. This means that if an edge ("a", "b") appears in the input edge list, then the output edge list should have both ("a", "b") and ("b", "a"). If you are still confused, check the comments already in the function and at least make sure that it converts the input example into the output example. **Your code should also remove duplicate entries from the output RDD.**

- There is a function noSelfEdges() that you need to fill in. If an edge like ("a", "a") appears in the input RDD, that is a self edge. The output RDD should look like a version of edgeList in which all self edges have been removed.
- Fill in the friendsOfFriends() function as specified in the comments in tri.scala
- Fill in the journeyHome() function as specified in the comments in tri.scala
- Read the comments in countTriangles() and fix it so that it now gives the correct answer when the toy graph is given as an input. Then use it to count the number of triangles in the facebook dataset.