1) Recommendation friends within 2 hops in social networks

Item		Description
Goal and purpose	Purpose	• What is a similarity join and why can it be applied to many application in real world? • How do we evaluate the 'friendship' between two users in social network? • How do we compute similarity joins using Spark system in parallel?
	Goal	• Understand the MapReduce paradigm and implement a parallel program in Spark • Understand similarity joins and apply it to find k nearest friend-pairs (i.e., pair of users) among all possible pair of users
Openness		• Choose appropriate methods (i.e., RDD operations) in Spark • Find the best similarity join algorithm
Virtual role		An ERICA student who starts his/her internship in an SNS provider
Scenario		A CTO of an SNS provider has a trouble. They already have a efficient metric to evaluate the similarity between two users and confirmed that it works well for friend recommendation. However, computing the metric between every possible pair of users in their 100m+ users takes too long time. Thus, the CTO gives it to an intern from ERICA majoring computer science.

Item		Description
Problem definition	Problem	Among all users, find all similar pairs who are not connected yet
	Input dataset	**1. Friends relations file (friend-relations doesn't have a direction)** format (in each line): <user ID1> <tab> <user ID2> **2. A threshold in Jaccard coefficient:** t
	Ouput	format (in each line): <user ID1> <tab> <user ID2> Note: order: ID1 < ID 2
Submission		1. README.txt - a command line that runs your program e.g., $ ./exp.sh edge.txt 0.85 output.txt or $ python3 exp.py edge.txt 0.85 output.txt - a command line that compiles your code e.g., ant or "None" 2. Source file and all scripts for running your problem
Evaluate		1. First, you must pass correctness 2. Compete with running time
Misc.		Jaccard coefficient Jaccard(A, B) = | A ∩ B | / | A ∪ B |