Updated: 2021-03-26

INFO 6205 Spring 2021
Project Description

Virus transmission simulation

- Your task is to simulate the spread of a virus such as SARS-CoV-2, the pathogen behind COVID-19.
- Your simulations will take into account:
  - The R factor of the disease (bonus points for including the k factor);
  - The usage and effectiveness of masks;
  - The prevalence of testing and contact tracing;
  - The availability and efficacy of the vaccine;
  - Any barriers to entry (including quarantining) into the subject area.

Goals/requirements of the project

- This is a somewhat open-ended project. The goal is for you to understand how to do simulations (you've already done some) and how to draw conclusions from your observations.
- You must describe the worst-case growth of any algorithms you create. Additionally, any data structures you employ must explain any invariant(s). Explain where you are getting entropy from for the simulation.
- In particular, I would like you to compare at least two viruses with different k/R factors: the most obvious one would be SARS-CoV ("SARS").
- You must write (and successfully run) unit tests for all of your formulas, rules, etc.–your code should have at least 60% coverage by line.
- The various parameters that you require must be defined via a configuration file (e.g. *config.ini*) that is easy to update.

Teams

- I expect you to form into teams of two or three (although other cardinalities will be considered).
- One person will be primarily involved in modeling the problem while the other will be concerned with a graphical user interface.
- If there is a third person, that person will be mostly concerned with documenting, charting, and cataloguing the results of the simulations.
- I need to know your team makeups by midnight of the 27th.
  - An editable spreadsheet will be provided for this.

Schedule and Deliverables

- Schedule
  - Teams: midnight 3/27;
  - Project deliverables: midnight on 4/19.
- Deliverables (via Github)
  - Code (Java or Python);
  - Unit tests (with proof of running);
  - GUI;
  - Report:
    - Complexity, invariants, entropy source;
    - Conclusions;
    - Graphs.

Grading

- (40) On-time with minimum requirements satisfied;
- (5) Repository (clean, and with contributions from all team members);
- (20) Conclusions;
- (10) Evidence (graphs, data, etc.);
- (10) Code (Java or Python);
- (5) Unit tests;
- (10) User Interface (video if possible);
- (5) Bonus points for any outstanding efforts, e.g. using k-factor.