Requirements for the class project

This document describes the requirements for the class project, including requirements for the presentation/demo (in class on May 28) and the final deliverables (due on June 4).

Overall, the goal of the class project is to produce a working prototype of a program analysis (broadly construed), and to demonstrate that it improves over what is already available.

Requirements during development
We do not require a particular tool chain, but each project should use the following:

- Version control (we expect the final deliverables to be available in a public repository, but you can develop in private until the final deliverables are due).
- Lightweight project management (e.g., GitHub projects/issues, Trello, Slack, ...).

Please make sure that the instructors have access to both of the above. The class project is a group assignment, but we reserve the right to change individual scores based on contributions.

Presentation format
The in-class presentations will be science-fair style. You should prepare a live demo and some printed material that describes your project. The demo must be performed live (i.e., it cannot be pre-recorded). A script is permitted, but expect that the instructors (and possibly other students) may ask you to go off-script to demonstrate the capabilities of your prototype. The demo should show off the most interesting features of your system.

The printed material can be either of the following:

- A poster describing your project (printed, letter-size sheets arranged on a posterboard are also fine).
- Info sheets describing your project. You should print enough info sheets for other students and the instructors (i.e., 20 or more copies).

The printed material should include information about your project that is not conveyed by your live demo. Examples include (but are not limited to):

- Motivation (why is this project worth doing?)
- Design decisions (why did you implement the prototype the way you did?)
- Testing and debugging (what testing/debugging approaches did you use? what unique challenges did you face?)
- Evaluation (how did you evaluate whether your prototype improves over existing tools?)

Each instructor will spend 10-12 minutes with each group. When your group is not talking to an instructor, you should move around to learn about what others have built; have one or two students available and rotate so that everyone gets a chance to see the other projects.

Final deliverables
The final submission on Canvas is a link to a publicly available code repository. Your repository must contain all artifacts and a README with links to the three kinds of documentation described below. Overall, your repository should provide the following:
● A polished **working prototype implementation** of the project.
● Evidence of following **testing and debugging best practices** during development.
● **User-facing documentation** that explains what the project does, how to use it, and why a potential user might care (e.g., use cases).
● **Developer-facing documentation** that explains how to build, test, and run the project. It should also describe:
  ○ major design decisions
  ○ style guidelines etc.
  ○ high-level design of the testing infrastructure and how to add new tests.
● An **evaluation document** explaining the methodology of your evaluation, what the results were, and how you interpreted them.