**COMP491 Assignment P2: Open-source contributions and checkpoint presentation**

This assignment is worth a substantial portion of the final grade (see syllabus) and is expected to require about three weeks’ work, with the usual expectation of 5-8 hours per week spent on the capstone project. Record your activity in real time on the Slack live log.

Students pursuing a research or honors project complete Part B of this assignment, but not Part A. Instead of Part A, please submit your Research Report to the instructor by email. For Part B, please consult with the instructor to determine appropriate content for the checkpoint presentation in your research project.

# Part A: Make progress on open-source contributions (60 points)

**Introduction**

In the previous bug gardening assignment, you became more familiar with your product and project by investigating bugs or issues that appeared in the issue tracker. In this activity, you will attempt to:

* fix bugs or address issues in the issue tracker; and/or
* complete the basic milestones listed on the “Basic milestones for the capstone project” webpage.

Your attempts will be documented on the course wiki, on Slack live logs, and in 5-15 reports. The 5-15 reports are separate homework assignments such as TR1, IR1, etc., but they overlap with this assignment by reporting on the progress you are making.

Work on Part A can be done in any combination of individuals, subteams, or the entire team. It is also perfectly acceptable to change configurations for addressing different issues. For example, an individual may work on issue or milestone X individually for one week, and then work in a subteam with a partner on a different issue or milestone Y for the next two weeks. Feel free to assign work within your team as appropriate.

**Identifying bugs or issues**

In the rest of this assignment, the word *issue* refers to bugs or issues that appear in your project’s issue tracker. Most H/FOSS projects have a collection of issues that have been tagged as “good first issue”, “beginner”, “introductory”, “easy”, etc. The developers have identified these as issues that have, in their opinion, a reasonably low barrier to entry and make good places to start. If your project does not have such a list, you should search the issue tracker and find an issue that is well specified and seems approachable. If there is an appropriate communication channel for your project, you might also consider asking the community if they have any suggestions for issues that might be good for a newcomer to the project.

Many communities use specific procedures for assigning issues to individuals. Other communities do not assign issues at all. Be sure to identify how your community handles the assignment of issues and follow their procedures. In particular, once you identify an issue to work on, be sure to inform the community via the appropriate mechanism (if there is one) that you are working on it. If you ultimately fix the issue, or abandon work on it, again be sure to inform the community via the appropriate mechanisms.

**Documenting Your Efforts**

You will document your efforts and results for this activity in three ways. Each team will collaboratively maintain an Issues Addressed wiki page that provides an overview of the work done by the entire team. You will continue to live-log your individual and sub-team work sessions in Slack to provide real-time information about when you have worked and the details of what you have done. Finally, you will write some 5-15 reports that summarize what you have accomplished, set goals for future work, and reflect on your effectiveness and ways you can improve (as described in separate assignments TR1, IR1, etc.).

**The Issues Addressed wiki page**

Create a new wiki page named “Issues Addressed”, linked from your team’s main wiki page. Create a table on the new wiki page similar to the one shown below:

| **Bug ID** | **Team Members** | **Description** | **Confirmed** | **Actions** |
| --- | --- | --- | --- | --- |
| *The identifier for the bug as an active link into the issue tracker* | *Initials of team member(s) working on the issue* | *A short description of the bug* | *Yes/No/Unknown - does the bug exist in the current development version* | *Chronological list of each action taken on the bug with dates and active links whenever possible.* |
| [7890](http://example.com/link7890.html) | GB/XY | Withdraw field allows zero amount. | No. | 11/1/22 - Appears to have already been fixed - [Commented on the ticket suggesting it be closed](http://example.com/7890-close.html). |
| [AB392c](http://example.com/linkAB392c.html) | GB/XY | Program crashes on dates before 1970. | Yes | 11/2/22 - [Claimed in issue tracker](http://example.com/claimAB392c.html) 11/5/22 - [Released in issue tracker](http://example.com/releaseAB392c.html) - Appears to be too complex for our current level of familiarity. |
| [X5432b](http://example.com/linkX5432b.html) | JP/PQ | UI unresponsive when sorting data. | Yes | 11/5/22 - [Assigned to self in issue tracker](http://example.com/claim5432b.html) 11/6/22 - [Proposed using Quicksort as a fix](http://example.com/comment5432b.html) 11/9/22 - [Made pull request](http://example.com/pr5432b.html) 11/11/22 - [Received request for changes on PR](http://example.com/prcomment5432b.html) 11/13/22 - [Pushed changes to PR](http://example.com/prupdate5432b.html) 11/15/22 - [Pull request merged!](http://example.com/prmerge5432b.html) |

The table should:

* Document every issue that your team attempts to work on, regardless of the final outcome.
* Include a list item for each *significant* event that occurs when working on the issue. Significant events would be things such as:
  + Starting work on the issue or claiming the issue.
  + Any communications with the community on the issue tracker or otherwise. These should include live direct links to the communication whenever possible.
  + Completion of work on the issue, regardless of the final outcome.

**Rubric**

Grading for Part A will be based largely on effort rather than achievement. However, achievement is also taken into consideration. For most capstone projects, completion of the basic milestones and substantial progress on one or more issues would be expected by the due date of this assignment. But the reality of open open-source contributions is that unexpected difficulties can create barriers to progress. If this occurs, it is still possible to obtain an excellent grade, but it is very important to document the efforts made to address difficulties. It is also important to communicate with the instructor and seek advice on alternative ways to make progress.

Individuals and subteams may receive different grades for Part A.

# Part B: prepare and deliver a checkpoint presentation (40 points)

The objective of Part B is to give a presentation explaining and demonstrating your open-source contributions to our class. Each capstone team (not subteam) will give one presentation. That is, each team will combine their subteams’ work into a single presentation.

Your presentation should be about 20 minutes in length, and there will be about 5 minutes of questions from the audience afterwards. (Teams with only one or two people can give a shorter presentation; the suggested duration is 10 minutes.)

You must submit your slides, in any reasonable format, to Moodle before the start of the class in which you are delivering your presentation.

To achieve an excellent grade, your presentation should excel on the following aspects. It should:

* meet the target length of 20 minutes (or 10 minutes for small teams), plus or minus two minutes;
* clearly explain the issues and milestones you attempted to address;
* clearly explain the progress your team has made on these goals;
* clearly explain any significant challenges your team has encountered;
* use clear and engaging visual materials (slides and/or whiteboard);
* employ large fonts that are very easy for the audience to read and use only a small number of words on every slide so that every word can be read by the audience;
* be delivered in a clear, engaging, and fluent voice;
* be delivered without reading verbatim from notes or slides (it is a good idea to use notes—just don’t read from them word for word);
* demonstrate understanding of technical details from the project and the ability to explain these details to the audience;
* convince the audience that substantial effort has been invested in the project, equivalent to 5-8 hours of work per week by each team member.

Usually, all members of the team will receive the same grade for Part B.