CMPUT 275 - Tangible Computing II **Winter 2020**

Final Project

The following are general expectations for the final project in CMPUT 275.

For your final project you will work in teams of two to four on a project of your own design. We want you to come up with a project that you are excited to work on and is of sufficient difficulty and complexity. This should involve some combination of implementing algorithms, devising your own wiring schemes on the Arduino and/or using the TFT display or parts provided in the Arduino kit. We do not require you to incorporate every one of these aspects into your project, but your project must be sufficiently advanced (see more discussion below and the sample proposal posted to eClass). A project that does not conform to this expectation will not be successful.

Specifically our expectations for a good project are:

- A project done by a team of two should be at least the size of one of the major assignments, and a project done by a team of three or four must be larger in size than the major assignments. This should be something you can reasonably complete before the deadline..
- It should be "demo-able".
- You must write your final project in C++ and may either:
  - Work entirely on the desktop using the Standard Template Library
  - Use the Arduino along with any desired components.
  - Implement a combination of both using Serial communication as in Assignment 2.
- It should demonstrate your command of the material covered in class. It should demonstrate your ability to write clean, efficient code using good style, and your ability to solve algorithmically challenging problems. You should be mindful of the time complexity of your code and the data structures you use.

Pitfalls to Avoid:

You should try to avoid projects like the following:

- Projects that involve a lot of simple "busywork". That is, the underlying task is incredibly simple (like playing a simple chiptune song through the buzzer) and it is made complicated by slough features that are not interesting from the perspective of the class (like adding a lot of random buttons and dials that just tweak the output from the buzzer). Think of a project that really demonstrates the coding, problem solving, and/or algorithmic concepts we taught beyond very basic programming.
- Projects that involve you using hardware that you do not understand very well. Maybe you want to interface the Arduino with some "shield" you picked up, but the shield does not have much support from the Arduino community. You might spend a lot of time getting frustrated and not making any progress because you can't get the external hardware to do what you want it to do.

Note that we are not forbidding this. But in the past we have seen some students not make much progress with their project because they were trying to figure out how to make some hardware work. This is just a strong caution.

Project Proposal

Each team must submit a project proposal. The instructors will provide you with feedback on the feasibility and appropriateness of your project ideas, based on the project proposal. Please be succinct and clear about what you intend to do. You must show that you have thoughtfully broken the task down into well-defined milestones with expected completion dates.

Each milestone should be up to a week worth of work. You should have something to demo at the end of each milestone. Note we do NOT want to see milestones that each implements a separate, non-demonstrateable piece of code and the final milestone is to integrate them all together. This is bad planning and can leave you without a demo-able project if you fail to reach the integration milestone.

Your project proposal should contain the following:

- Project title
- The full name of each team member
- Brief description of what your project idea is and what the demonstration at the end will look like.
- Include 3-6 milestones for your project and an expected completion date for each one.
- Brief description (2-3 sentences) of what you expect to achieve by the end of each milestone and what needs to be done to achieve it. Be explicit about what the demo for each milestone would be.

Proposal Examples and Ideas

We provide one example for your reference, which can be found on eClass. You are not limited to this project; it is simply an example of a good proposal write-up. Additionally, below you will find a list of project ideas, some of which have been implemented successfully as final projects in the past.

Note that you may choose to use any of these ideas as your final project. However, be aware that many have already been done well in the past. If you choose a past project, we will expect you to live up to our expectations.

Project Ideas

Here is a general collection of project ideas. These are not full project proposals, but you may find inspiration from some of these keywords. Note that some of the projects classified as "Arduino" or "C++" could be adapted for either or both.

Arduino Projects:

- A 3D model viewer that allows rotating, viewing, and rendering objects of different sizes.

- A game of ice hockey: two screens pushed together to make a larger board, using Serial communication to pass the puck between screens.
- Snake, the game: implemented efficiently with constant time updates using a queue.

C++ Projects

- A graph visualizer, which takes input graphs in the style of CMPUT 275 and displays the vertices and edges in a pleasing way. You would need to find a way to extend this into a large enough project (*i.e.*, a nice user interface, efficient code, a sophisticated parser for the input graphs, etc).
- A text editor which supports spell-checking and autocompletion.
- A parser and interpreter for a simple language that supports several different types of operations. This could be done using an AST.
- A visualizer for the Fourier transform algorithm. An extremely ambitious project could do this for fast Fourier transform (FFT) instead.

We see a very wide range of project ideas, many do not look like one of the ones mentioned here. You get to choose what you want to do! These are just examples to get you thinking.

Proposal Submission Details

We are asking you to submit a project proposal by **March 10 at 11:55 pm**. We will still provide feedback on late submissions, but submitting by this date is best as you will get early feedback. You will not be graded for your project proposal, but we do require that you submit a project proposal. This is your best chance to elicit feedback.

You must submit a document (plain text or pdf files only). Do **NOT** submit a .doc or .docx file.

Final Project Evaluation

Your project will be graded in two phases: the demo and the submitted code. First, you will be asked to demo your project for one of the instructors and/or a TA. After the demo, they will ask you questions about your project. You will be graded both on the quality of your presentation and on your ability to answer these questions. Second, you will be asked to submit your code so it can be graded for style, efficiency and quality.

Demo

The classes on **March 31 and April 2** will be devoted to demos. We will decide which groups will demonstrate their project on each day and your team only needs to show up for one of these days. We will have a few teams of instructors/TAs going around the lab to view your demo. At this time, you will give a short (usually 2-3 minute) demonstration of your project for the evaluators. We may ask some questions about certain features of your project right after the demo. Questions may range from code comprehension, object oriented design, to running time efficiency of the algorithms. These are typical questions, but of course we may ask you anything aspect of the project. Then we will conduct a brief "one-on-one" interview with each team member, asking

about the student's individual contribution to the project, along with a quick code review of what the student views to be their most interesting contribution to the project.

The demo grade will be based on overall impressions of your project (how complete it appeared, did it go smoothly, did it meet our expectations in terms of appropriateness and challenge) and how well the questions were answered by each team member.

Final Project Code Submission

You must submit your code no later than **April 10 at 11:55 pm**. The code you submit will be graded based on organization, style, and correctness.

Anything that is required to build the physical components, run the code, and instructions on how to use the final program itself (i.e. a "user's manual") must be included in the README file. If you use the Arduino, your README should contain wiring instructions, a list of componets, etc.

The code may or may not be run by the grading TAs (how well it ran would have been evaluated in the demo). For Arduino projects: if you use components not in the Arduino kit then a TA may request you bring a fully built project if it helps them complete the evaluation of the code.

Submission Details:

Upload a compressed archive (.tar.gz or .zip) of the directory containing all the bits of your project.

Within this directory, you should include everything a stranger would need to be able to use your project, assuming they had all hardware components. This may include, but is not limited to:

- A README including instructions on how to use your project and all the other details you normally expect in a README. If you have multiple files for your code, you may also want to explain the general layout of your code/files. You may also include any reference materials that could help others to understand (part of) your project.
- Any pictures of your project that help make the setup, use and explanation of your project clear.
- All of your code, including extra libraries.
- **If using the Arduino**: A wiring diagram/details and the Arduino Makefile.
- **If your project uses desktop C++**: A custom Makefile for your project such that the main version can be built using "make".
- A brief list of points that someone building your project needs to know about in order to rebuild it later, such as useful notes on anything usual for the build, special parts, tricky wiring etc.

README Project Notes:

In addition to the usual information you would include in a README (including what was mentioned above, as applicable), you should include a few comments on the project itself. Specifically:

- Give a high-level breakdown of who worked on what. This does not need to be detailed, just briefly explain what aspects each person worked on.

- Explain what was completed in the time between your in-class demo and the final code submission deadline.
- Explain any aspects of your project that were in the proposal but that you failed to complete in time. **Caution**: If you don't complete some parts of your project, then you should submit your most recent *working* code. If your code does not compile or does not run because a last-minute feature was not finished properly, you can expect significant deductions (perhaps even a 0 on the code submission portion of your project, it is not the grader's job to hunt down compile errors or critical issues in the code).

Remember! You want your submission to be easy to use. In particular, you want to make your TA's life easier. Aim to be as clear and as explicit as you can be.