ENGR131 – Final Project

One team member must submit the following:

Phase	**Points**	**Due at midnight at end of the day (except *)**	**Submit** (on Canvas) **or Use Online Form**
Choose partner	0	*Due in lab Nov 6–9	Included in Part A of Lab #9
Proposal	10	Tue, Nov 20	Complete online form on website
Alpha Version	20	Tue, Dec 4	M-files, input file, and any other necessary support files
Presentation	extra: 10	**Optional:* Thu, Dec 6	Complete online form on website
Beta Version	30	Thu, Dec 6	M-files, input file, and any other necessary support files
Final Version	40	Mon, Dec 10	M-files, input file, and any other necessary support files

Introduction: The final project is worth 10% of your grade for the course. You will write an application of your own choosing that uses various techniques learned throughout the semester. You will create a software development team by choosing one or two partners. The project will follow a professional software life cycle (http://en.wikipedia.org/wiki/Software_release_life_cycle).

General Requirements: Teams must have 2 – 3 people (no more, no less). List your partner(s) names as explained in Lab #9, Part A. You may request to have a partner assigned to you as explained in Lab #9. All content must adhere to a PG-13 rating. Each student should contribute the same amount of effort to the project so as to maximize his or her learning. The scope and difficulty of the overall project should be large enough that an "average" student would require at least 12 person-hours to complete the project. This will be determined by the ENGR 131 teaching staff based on a submitted proposal.

Technical Requirements: Below are the *minimum* requirements. You may use code written by someone else (including ENGR 131 assignments), but your team must create new, original code that meets each of these requirements.

1. You must design a MATLAB event-driven GUI. The interface may consist of text, graphics, or a combination of both.
2. You must use all of the following programming techniques:
   a. Either an array of cells or an array of structs.
   b. A function that accepts a variable number of input arguments
   c. A data file that is read using either "fscanf" or "textscan" and is modified by the program using "fprintf".

Development Phases (see due dates above)

1. *Proposal (10 pts)*: The purpose of the proposal is to make sure you have a design that meets the general requirements. Using the online form, explain the following about your program: (a) a name (for example: *Big Adventure*), (b) what you plan for it to do, (c) your idea for an array of cells or array of structs, (d) your idea for a variable-input function, and (e) your idea for file I/O. It will be graded based on clarity and completeness. The design of your application can be changed later, but any significant changes should be re-approved by your TA.

2. *Alpha Version (20 pts)*: In the software industry, "alpha" indicates the first version that is available for testing within the company. Yours is allowed to be quite worse than a typical alpha and does not need to be functional. The intent is to provide a clear and complete framework of the entire application by providing a GUI and showing where the cell array (or array of structs), variable-input function, and file input will be used. Include a progress report as a comment in your .m file describing the current state of your program. Submit .m files, the input file, and any other support files (e.g. graphics, audio).

   The cell array (or array of structs), variable-input function, and input file do not actually need to be used exactly as they will be in the final version. However, some example data should be stored in the cell array (or array of structs) and the data file, and the variable-input function should be called. The GUI must

respond in some way to all of the possible user input and provide some indication of what action would be performed for that input. You should be able to run the program without errors. The GUI does not need to have the final graphics (if there are supposed to be any), and responses to user inputs do not need to be realistic. For example, clicking on a button can simply cause a message to be displayed to explain what is supposed to happen (e.g. "The hero would move to the square above").

Grading rubric (4 points each): (1) overall GUI and program, (2) progress report, (3) cell/struct array, (4) variable-input function, (5) file input.

3. *Beta Version (30 pts)*: In the software industry, "beta" indicates a version that is ready to be tested outside of the company. Yours is allowed to be quite worse than a typical beta. The intent is to have a demonstration version with at least minimal functionality for the cell array (or array of structs), variable-input function, and file input, even if the program output is incomplete. Include a progress report as a comment in your main .m file describing the current state of your program. Submit .m files, the input file, and any other support files (e.g. graphics, audio). *Note: do not use large data files (>5 MB) in your submissions because these are difficult for teaching staff to work with.*

   Grading rubric (6 points each): (1) overall GUI and program, (2) progress report, (3) cell/struct array, (4) variable-input function, (5) file input.

4. *Final version (40 pts)*: In the software industry, the final or "release" version should be flawless. While yours should be fully functional, minor flaws in the algorithms might not be penalized, depending on their severity. It should be free of errors. Be sure to simplify the application if your original ideas are not working. Include a progress report as a comment in your main .m file describing any known issues with your program. Submit .m files, the input file, and any other support files (e.g. graphics, audio, etc.). *Note: do not use large data files (>5 MB) in your submissions because these are difficult for teaching staff to work with.*

   Grading rubric (8 points each): (1) overall GUI and program, (2) progress report, (3) cell/struct array, (4) variable-input function, (5) file input.

5. *Presentation (extra credit: 10 pts)*: **This part is optional and is worth 1% of your grade for the course.** Either demonstrate your program or create an advertisement describing its exciting features and showing at least one screen shot (actual or hypothetical). Presenters will be required to register in advance using an online form. (Note that you should *not* send any files to Chris or your TA.) Presentations should be short (2 – 3 minutes is ideal). Only students who are present will receive credit.