**Project Description**:

The final project is your opportunity to demonstrate you programming skills in a context of your choosing. You are tasked with developing an idea, which can be recreating an existing game, developing a database, or creating a program to make planning easier. Refer to the list at the end of this document for options, and choose a topic that you know well and you will enjoy working on. To further develop your idea, you should plan out the flow of your program through algorithms, models, flow charts, tree logic maps, mock up sketches, and/or some other type of planning tool. Once you develop an idea, you must implement your idea through coding and debugging. Follow the project submissions, project requirements, and coding requirements. ***NOTE:*** *If you have taken this course and started a final project before, talk to your professor about your previous project immediately. If you submit the same or similar final project that you submitted in a previous semester, this will be considered fraud per our academic integrity policy and will result in being submitted for an academic integrity violation.*

**Project Submissions**:

Keep in mind your final project is NOT something you can complete in one weekend. To help you spread your work out and be more successful, this project will be submitted across the five following submissions. Keep in mind deadlines come up fast, so I recommend putting them in your calendar now.

* **Reserve Project Idea/Topic** due: 10/4/19 percentage: 2.5%
  + For this submission you must go over the provided choices and choose a topic that you would like to work on for the final project. You must write a few paragraph explaining why you choose that topic and describing what you plan to do in the project.
* **Project Proposal/Analysis** due: 10/11/20 percentage: 4%
  + For this submission you must write out more details about your final project idea. Use the project proposal or beginnings template, if one is provided. If not, write out a long form description of your project. The purpose of this submission is to help ensure you understand the goals of your project before you start coding. ***NOTE:*** *Upon completion of this assignment you MAY NOT change your final project idea without your professor’s explicitly written approval. Change will result in significant loss of points. You may also not ask to change your topic within a week of a draft submission or the final submission.*
* **First Draft of Project** due: 10/18/20 percentage: 8.5%
  + For this submission you must submit a draft of your final project. You are expected to have a working code. It does not have to be complete, but it must demonstrate the Program Techniques from the Rubric. The rubric will be a slightly modified and shortened version of the Final Project Rubric.
* **Second Draft of Project** due: 11/8/20 percentage: 8.5%
  + For this submission you must submit a draft of your final project. You are expected to have a working code. It does not have to be complete, but it must demonstrate the Program Techniques from the Rubric. The rubric will be a slightly modified and shortened version of the Final Project Rubric.
* **Third Draft of Project** due: 11/22/20 percentage: 8.5%
  + For this submission you must submit a draft of your final project. You are expected to have a working code. It does not have to be complete, but it must demonstrate the Program Techniques from the Rubric. The rubric will be a slightly modified and shortened version of the Final Project Rubric.
* **Peer Review on a Draft** due: 11/15/20 percentage: 3%
  + For this submission you will review a peer’s final project and they will review yours. Use the peer review template, if one is provided. If not, use the final project rubric to help provide each other meaningful, constructive feedback and document what feedback you gave each other.
* **Final Project** due: 12/4/20 percentage: 15%
  + For this submission you will submit your final project in a zipped folder. You must include all relevant files. This submission will be graded using the final version of the Final Project Rubric. ***NOTE:*** *This is the only submission that late submissions will be accepted, but you will lose 5% on your submission for every hour late and the time late will be rounded up. (Examples: If you submit a final project that is assessed to be a 90% 1 hour and 20 minutes late, you will lose 10% and receive an 81% for this submission. If you submit a final project over 19 hours and 1 minute late, it will be considered 20 hours late or later and you will lose 100% and receive a 0 on this submission.)*

**Project Requirements**:

**Comments in Code:** You must include comments in every script file submitted that cites any resources outside of course materials used (e.g., hyperlinks for blogs, hyperlinks for MathWorks), acknowledges any additional help you received from anyone other than your own professor with the person’s name (e.g., Professor X, Taylor the tutor, my roommate Jordan), and an explanation of how the material was used and/or the person helped with the code. If you are using lines of code substantially written by others to enhance your code, you must clearly comment these lines of code and their source; these may be used in your code, but they will NOT count for any points in the assessment of your final project (e.g., if your only programmer-defined function is one that someone else wrote, you will receive a 0 for “create at least one programmer-defined function”).

**Programming Techniques:** All of the programming techniques in the table below must be demonstrated in your code for reasonable uses that make sense for your project. *(refer to final project rubric for more details)*

|  |
| --- |
| **INPUTS:** |
| User inputs/interface (e.g., input, dialog boxes, GUI) |
| File input (e.g., xlsread, dlmread, getcsv) |
| Random numbers (e.g., rand, randi) |
| **DATA TYPES:** |
| Numeric data and strings |
| Arrays (i.e., vectors, matrices, cell arrays) |
| **DATA PROCESSING:** |
| Rounding (i.e., round, ceil, floor) |
| Counting/calculating (e.g., running total, sum, mean, multiplication) |
| Organizing/analyzing (e.g., sorting, searching) |
| Array manipulation (i.e., referencing, slicing, augmenting, and diminution) |
| **LOGIC:** |
| Conditions (i.e., relational operators and Boolean operators) |
| Conditionals (i.e., if statement, switch/case) |
| Loops (i.e., while loop, for loop) |
| Nesting (loops and conditionals) |
| Error checking (all inputs appropriately error checked) |
| Programmer-defined functions (input parameters and return values) |
| String functions (e.g., sprintf, strcat, strfind, strcmp) |
| Other built-in function not discussed in class |
| **OUTPUTS/DISPLAY:** |
| Display relevant outputs to user/interface (e.g., fprintf, dialog boxes, GUI) |
| File output (e.g., xlsread, dlmread, putcsv) |
| Plotting (with appropriate data, formatting, and labeling) |

**Scoring Markers in Code:** You must add score markers (%<SM:\_\_\_>) as defined in the final project rubric for one occurrence of each programming technique. You will lose 1 point for each missing scoring marker. You will lose 3 points for each occurrence of mislabeled scoring markers (e.g., a for loop labeled as a while loop).

**Overall Coding:** You must also make your interface user-friendly, strive for efficiency in your code, and format your code neatly. Refer to the final project rubric for more details about grading for the final project submission.

**Coding Requirements**:

The coding guidelines discusses best practices that must be followed; points will be lost for all of these not addressed in your final project. The list of inappropriate code presents banned code; the use of these will result in the loss of significant points on your final project. Ask your professor for additional examples of how to avoid these codes.

***Coding Guidelines:***

These guidelines refer to coding practices that you must implement in your code. Not following any of the coding guidelines will result in the loss of points on your final project (*refer to final project rubric*).

* Clear user interface with clear instructions and organized code (with comments). Please comment your code in a way no one has to read the code to understand the ideas. Example of user inputs:

heightM=input('Enter value: '); %HORRIBLE

heightM=input('Enter height: '); %better

heightM=input('Enter height in meters: '); %much better

heightM=input('Enter height in meters (0.5 to 2.5m): '); %perfect

dateYMD = input('Enter date: ','s'); %HORRIBLE

dateYMD = input('Enter date (year,month,day): ','s'); %ok...but...

dateYMD = input('Enter date (for example 01162020 for jan.16.2020): ','s'); %perfect

dateYMD = input('Enter date (for example dd-mm-yyyy): ','s'); %also perfect

* One interface used (command window OR pop-up windows/GUIs). Note that a figure window popping up to display a figure or plot is acceptable with code that only uses command window – this will not count as multiple interfaces, since the user will not have to click or input information on these windows.
* User inputs properly error-checked. *(Refer to the standards of your class.)*
* Minimize Hardcoding. *Use data files to contain large amounts of what would otherwise be hard-coded information.*
* Avoid repetitious switch/if statements. *Many times there are ways to minimize these through loops, programmer-defined functions, and/or using arrays and slicing.*

|  |  |
| --- | --- |
|  |  |

***List of Inappropriate Code/Techniques:***

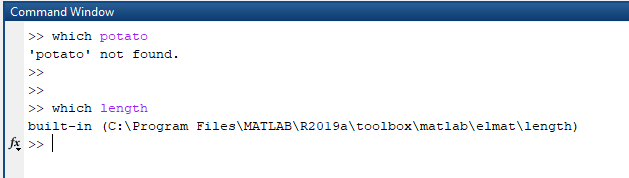
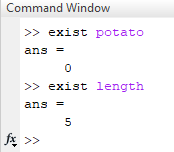
These guidelines refer to coding practices that you must NOT implement in your code. Use of any of these will result in 10 point reduction for each category unless explicitly stated elsewhere (example: use of GUIDE is an automatic 100% reduction and results in a 0, as stated below). For example, if there is a variable called i, a variable called ans, 4 uses of break, and one use of exit in your final project, you will lose 30 points on your final project.

Most of these are tools which can be useful (and not necessarily detrimental) when used in special situations. New programmers will generally use them instead to bypass good design practices, or the tools allow students to avoid using the techniques we are attempting to teach. As such, they are BANNED.

* GUIDE and App-Designer for code creation will result in a 0 on the final project. This includes all .fig files.

|  |  |  |
| --- | --- | --- |
|  | \* If you are going down this path, you will get a **ZERO**. **DO NOT** use GUIDE or App Designer. |  |

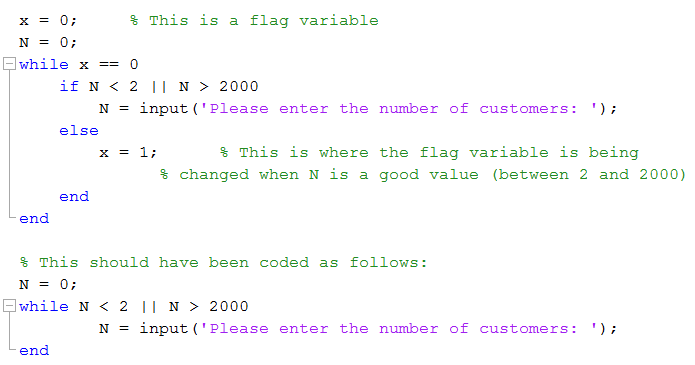
* global, persistent, and static variables are NOT allowed. (This includes the use of evalin and assignin.) setappdata and getappdata are also NOT allowed. For levels of coding in this course, this is a lazy way to avoid transferring variables/data. *For other methods, consider using struct (structures) for GUIs instead. Refer to* *https://www.mathworks.com/help/matlab/matlab\_prog/create-a-structure-array.html*
* Calling a program. *(This is not referring to calling a programmer-defined function or other built-in function.).* This means typing the name of the file to execute it, but this file is not a function file. To fix this, turn the file into a function without return-values or parameters then call the function!
* Defining a programmer-defined function inside another programmer-defined function.
* break, continue, return, and run functions are NOT allowed. For levels of coding in this course, this is a lazy way to avoid writing a well-though-out algorithm.
* quit( ), exit( ), and error( ) are NOT allowed. These will terminate the MATLAB program.
* You many not create a variable or function with a name that is already used in MATLAB (e.g., i, j, ans, edit, help*). To check and see if a name exists in MATLAB, use either method below:*
  + *the* ***which*** *function will tell you ‘not found’ or a path to an existing file.*



* + *the* ***exists*** *function will tell you 0 if*  name does not exist or cannot

be found for other reasons”, or a number if it exists somewhere

* Inappropriate flag variables. *Flag variables are variables used to track events. In very complex situations they can be useful – in most of our class, they are not needed. An example of an inappropriate flag variable is below, as well as how to remove it:*



* The use of disp, display, table, summary, celldisp, and other print functions are NOT allowed. You should use fprintf() to format any data that you want to display to your user in the command window. *(Avoiding these by omitting semi-colons to display information will result in this same deduction; this deduction applies to any unsuppressed lines of code in your final project.)*
* Evaluating string inputs as numeric inputs is a NO. *An example of this is shown here:*

%define Yes and No variables

Yes=1;

No=2;

%ask User to do something

%BAD: this is missing 's' at the end so input() is really scanning for a

%NUMERICAL value. When use enters Yes, MATLAB is forced to search for a

%variable Yes to associate it to the value 1.

response = input('Enter Yes to play, No to quit: ');

%GOOD: do this instead!

response = input('Enter Yes to play, No to quit: ','s');

%or simply:

response = input('Press 1 to play, 2 to Quit: ');

* The use of .mat files are forbidden.
* The use of use of "strings" instead of 'character vectors' (strings taught in course). Always use single quotes/apostrophes.

**List of Sample Project Ideas**:

You will choose one topic from the list below. Keep in mind you will be spending A LOT of time working on your final project so pick something you enjoy. Also know that up to two students may choose the same idea, so be sure to reserve your idea as soon as possible and provide details in why you choose this topic and what you plan to have in the project.

***Games:*** *These projects will typically be based on an existing game, but in some rare cases may be a game that you want to generate. These games may be single player, player vs. artificial intelligence (AI), or player vs. player. These games may have many players and/or many AIs. These may be simplified versions of complex games or complex versions of simple games.*

|  |  |
| --- | --- |
| Title | Category |
| Dining Philosophers Problem | Computer Science |
| Knapsack Problem | Computer Science |
| Eight queens puzzle | Computer Science |
| Traveling Salesman solver | Computer Science |
| Blackjack | Card Game |
| WAR card game | Card Game |
| Texas Hold'em | Card Game |
| Euchre | Card Game |
| Spades | Card Game |
| Hearts | Card Game |
| Yahtzee | Dice Game |
| Quixx | Dice Game |
| Left Center Right | Dice Game |
| Farkle | Dice Game |
| Shut the Box | Dice Game |
| Double Shutter | Dice Game |
| Parcheesi | Game |
| Sorry! | Game |
| Battleship with AI | Game |
| Connect 4 with AI | Game |
| MasterMind with AI | Game |
| Mancala | Game |
| Hangman | Game |
| Othello | Game |
| Checkers | Game |
| Sudoku solver | Game |
| Uno | Game |
| Clue | Game |
| Deal or No Deal | TV Game Show |
| Jeopardy | TV Game Show |
| Wheel of Fortune | TV Game Show |

***Database (either extensible database management system or applied database): Extensible database management system*** *– These projects should allow the user to add and delete new rows and columns of information to the database with extensive error checking and ability to handle newly implemented information. Meaning the user must be able to add/remove new records of information (typically rows in data set) and add/remove new categories of information (typically columns in data set). There must also be sorting, filtering, and viewing features.* ***Applied database*** *– These databases must also have sorting, filtering, and viewing features, but they are not required to have add/delete features (as described for extensible database management system). This databased must have some additional feature that requires user-inputs that have to be analyzed to make some type of conclusion based on a developed algorithm and the data in the database (making it applied).* ***Combination*** *– You may also do a combination of some of the ideas from both. Just make sure your project is complex enough to meet the requirements for this*

*assignment. If you do not meet the requirements for at least one of these categories, your project will be considered “overly simplified/poor” on the project difficulty multiplier (-15%).*

|  |  |
| --- | --- |
| Title | Additional Information |
| Hotel Front Desk |  |
| Music or Movie database | User can add/remove/edit records and can search/sort records based on various criterion |
| Find your ideal pet |  |
| Pokémon Pokédex | User can add/remove/edit records and can search/sort records; include pictures |
| Quiz / Test Generator | Question bank grouped in subjects, topics, formats and so on; avoid hardcoding –questions must have some variability |
| Recipe Keeper | Database of ingredients, variable units, recipes, pictures (help user select a recipe to cook) |
| USAF aircraft | user can add/remove/edit records and can search/sort records; nice to include pictures |
| Small business employee information | Database of past and current employees, performance, HR information, payroll information and so on |
| Daytona Beach restaurants/attractions | Fun idea generator or guide to the area |