

This is the Final Project Rubric that will be used to assess your final project submission. It is out of 85 points. Be sure to note there is a multiplier that impacts your final grade for this assignment.

Final Score Multiplier <i>total points are multiplied by this number</i>	Multiplier = 1 + (Project Topic/Goals) + (Project Difficulty) + (MATLAB Errors)			RANGE: 0.6 to 1.1
(Multiplier) Project Topic/Goals	Unsatisfactory (- .10)	Completed proposed project or written approval for change/reduction of goals (+/- 0)		
(Multiplier) Project Difficulty <i>(use at professor's discretion)</i>	Overly simplified/poor (- .15)	Everything else (+/- 0)	Complex (above & beyond) (+ .10)	
(Multiplier) MATLAB Errors	Lots of errors (- .15)	More errors (- .10)	Some errors (- .05)	No errors (+/- 0)
Inappropriate Codes/Techniques <i>(refer to final project document)</i>	Total number of categories checked: <b>(times 10 points for each category)</b>			Total points lost:
<i>NOTE:</i> Based on the use of forbidden codes, you can get a 0 on your final project, but you cannot get below a 0. <i>REMINDER:</i> GUIDE or App-Designer – automatic 0 on final project.				
global, persistent, or static variables or use of setappdata or getappdata (e.g., evalin, assignin)		quit(), exit(), or error()		evaluation of a string input as numeric
calling program/s (script files – not functions)		variable/file names (already exist) (e.g., ans, i, min, input)		.mat files
a programmer defined-function defined in another programmer-defined function		inappropriate flag variables		data type: no “strings” only ‘char’
break, continue, return, or run		display, table, print, unsuppressed outputs, etc.		
<b>Program Characteristics (out of 25)</b> <b>(Formatting, Presented Work)</b>	No Attempt (0 pts)	Poor Attempt (1.5 pts)	Fair Attempt (3 pts)	Meets Expectations (5 pts)
Clear user-interface with clear instructions and organized code <i>(with comments)</i>				
One interface used (command window or pop-up windows/GUIs)				
User inputs properly error checked				
Effective use of programmer-defined functions <i>(avoid exceptionally long files)</i>				
No/minimal redundant and hardcoded information <i>(refer to coding guidelines)</i>				
<b>Programming Techniques (out of 60)</b> <b>(Use of Learned Coding...)</b> (comment in your code ONE example where each of these codes are used with the scoring markers shown below – %<SM:___>)	Not present (0 pts)	Present, but used incorrect syntax – errors (or not all required parts present) (1.5 pts)	Present and correct, but poorly used or no clear purpose (or no score marker) (3 pts)	Present, correct, proper use, and serves purpose in program (4 pts)
FOR loop %<SM:FOR>				
WHILE loop %<SM:WHILE>				
Conditionals (i.e., IF statement, SWITCH/CASE) (only one type required) %<SM:IF> %<SM:SWITCH>				
Relational operators %<SM:ROP>				
Boolean operators %<SM:BOP>				
Nesting (for, while, if, switch) %<SM:NEST>				
Create at least one programmer-defined function %<SM:PDF_CALL>				
Input values to programmer-defined function through input parameters and Collection of return values from programmer-defined functions %<SM:PDF_PARAM> %<SM:PDF_RETURN>				
String functions (e.g., sprintf, strfind, strcmp, str2double) %<SM:STRING>				
Array manipulation (referencing, slicing, augmentation, diminution) (two required) %<SM:REF> %<SM:SLICE> %<SM:AUG> %<SM:DIM>				
Data handling (running totals, sorting, searching) (one required) %<SM:RTOTAL> %<SM:SORT> %<SM:SEARCH>				
Random numbers (randomly generate number/s, use random numbers) (both required) %<SM:RANDGEN> %<SM:RANDUSE>				
Create/display a plot (e.g., plot, bar, pie) (one required) %<SM:PLOT>				
File input (e.g., xlsread, dlmread) or file output (e.g., xlsxwrite, dlmwrite) (one required) %<SM:READ> %<SM:WRITE>				
Built-in function not discussed in class %<SM:NEWFUN>				
<b>Number of mislabeled scoring markers:</b>	<b>(times 3 points each)</b>			<b>Total points lost:</b>