General Programming Assignment Rubric

Attribute	Unacceptable	Meets Requirements	Exemplary		Points
Presentation					
Information	**Missing Information**	**Properly prepared for submission;** includes all names; TA; Lab section; Assignment number; Statement of Originality	Nothing missing – purpose & program highlights obvious.		
Files	**Missing files**	**All included:** required source, data, documentation, script, instruction files	Nothing missing – clear labeling of all parts; readme file includes everything needed to run & test this program		
Submission:	**> 3 min. :**Unable to figure out how to compile & run after 3 minutes.	**2-3 min.** Takes 2-3 minutes to figure out how to compile & run.	**< 2 min.** Takes less than 2 Minutes to compile; run; test		
Demo:	**Major Problems:** Program won't run or crashes unexpectedly. Submitted code seems different from that demonstrated. Programmers seemed confused by the program they were demonstrating. Only one group member participated (out of three).	**Minor Problems:** Program runs reasonably well. Knew what they were doing. Submitted code matches demo. One group member did not participate.	**Good!** Demo went well; major features shown and explained. Demonstrators clearly understood their work. All group members participated.		
				Presentation Total	**/6**
Documentation					
Overall Impression	**Misleading**; confusing; too much or too little	A **few** missing, redundant, or irrelevant parts	**Accurate**; reasonable; easy to read;		
Identifier Names	**Meaningless** or misleading names	**Some** poor choices. Most identifiers explained where appropriate.	**Meaningful** identifier names [some single letter names are OK, such as i,j for indices]. Explanations of identifiers where appropriate.		
Indentation; White Space	**Misleading** indentation; too much or too little white space	**Some** inconsistencies; some inadequate or wasted space	**Consistent** indentation; good use of white space		
External Documentation	**No** external documentation when some is needed. External documentation not useful, confusing, out of date, or misleading	**Adequate** external documentation [javadoc; user manual, as necessary]. Someone else could work with this program with a little help from the original programmer(s).	External doc. as **appropriate**; [javadoc & user documentation where appropriate]. Someone else could work with this program based on code and documentation alone.		
Logical Blocks	**Few** or no logical blocks documented	**Most** logical blocks documented	Documentation for **each** function and loop and logical block		
When Defining Classes:	**Missing** class diagrams	**Reasonable** attempt at class diagrams	Includes some readable form of **class diagram**		
Single Static Class or "C"	**Missing** flowchart	**Reasonable** attempt	Flowchart or other **readable representation** of program flow		
				Documentation Total	**/12**
Design					
Implemented what was asked.	**Hardly** followed specs; no explanations for deviations	Some **deviations**	**Followed specs** (deviations well justified)		
Style & Efficiency	**Poor** choices of code; awkward structures	**Mostly well thought out;** most parts reasonably efficient	**Well thought out;** reasonably efficient (code & data structures)		

Program Subdivisions	**Too many or too few**	**Mostly reasonable** with a few poor choices	**Reasonable** subdivisions (classes/ functions)		
Flow	**Confusing**; hard to follow	**Some awkward** or confusing parts	**Logical**, justifiable structure		
Constants; Magic Numbers	**Magic numbers**; hard-coded values	**Most** constants named & explained	**Appropriate** use of constants		
Globals	**Inappropriate** use of globals.	**Most** globals properly justified.	**All** globals properly justified.		
Initialization & Clean-up	**Uninitialized** values; no clean-up	**Some** uninitialized values; some clean-up **missing**.	**Good!** Reasonable initialization and clean-up		
Structure	**Not clear –** blocks do too much or too little	**Reasonable,** given degree of difficulty of assignment and learner's expected level of experience	**Excellent:** creative, logical, well-delineated with respect to tasks and data		
				Design Total	**/16**
Input / Output / Interface					
Intro / Finish	**No** header or introduction; no obvious end.	**Has start & finish.**	**Good!** Has nice introduction; clear "sign-off";		
Output (or public interface for Class):	Output **unexplained**.	Output **readable**.	**Good!** Output is clean and reasonable.		
Input : Interactive Programs:	**No explanation** of input requirements	**Minimal** explanation of input requirements	**Good!** Input requirements explained as program runs		
Input : "Batch"(or 'user' interface for Class):			Input requirements **well explained** in external documentation		
				Input / Output / Interface Total	**/6**
Testing / Error Detection / Correction					
Choice of test data	**Missing** test data; poorly tested	Only **tested with given data** or only partially tested	**Thorough**; includes some data not given		
Annotation	**Not annotated;** hard to find & follow the testing; confusing	**Reasonably well annotated;** fairly easy to see & follow the testing	**Well annotated**; easy to see & follow the testing		
Endpoints	**Insufficient**: Only one or two endpoints tested	**Most** endpoints tested	**All** endpoints tested as appropriate		
Detection; Correction; Limitations	**Non-existent** error detection/ no attempt at correction/ unreasonable limitations	**Mostly reasonable** error detection/ attempted correction/ mostly reasonable limitations	**Reasonable** error detection/ correction/ limitations		
Debugging Aids	**No** evidence of debugging aids	**Some** evidence of built-in debugging aids	**Reasonable** use of DEBUG flags, program tracing and other debugging aids		
				Testing / Error Detection / Correction Total	**/10**
				[Rubric One] Style & Design Total:	**/50**