CodeX Project Rubric - CSTA Standards

Standard	Basic	Proficient	Mastered
Documentation			
2-AP-10 Use flowcharts and/or pseudocode to address complex problems as algorithms.	Incomplete flowcharts.	Flowcharts provided for each process.	Flowcharts provided for each process. Evidence of revisions and improvements made.
2-AP-13 Decompose problems and subproblems into parts to facilitate the design, implementation, and review of programs.	Code is not organized or readable.	Code is sometimes organized into problems and subproblems in order to make it organized and readable.	Code is decomposed into problems and subproblems, making it easy to follow and read.
2-AP-19 Document programs in order to make them easier to follow, test, and debug.	Incomplete documentation.	Documentation provided for each process.	Documentation provided for each process. Evidence of revisions and improvements made.
Algorithms and Programming			
2-AP-11 Create clearly named variables that represent different data types and perform operations on their values.	No variables; variables not named appropriately.	Variables used and named correctly in most instances.	Variables are used and named correctly in each process as needed.
2-AP-12 Design and iteratively develop programs that combine control structures, including nested loops and compound conditionals.	No loops or conditionals.	Loops or conditionals used correctly in most instances.	Loops or conditionals are used correctly in each process as needed.
2-AP-14 Create procedures with parameters to organize code and make it easier to reuse.	No procedures; procedues not named appropriately.	Procedures used and named correctly in most instances.	Procedures used effieciently to organize code and reused as needed.
2-AP-16 Incorporate existing code, media, and libraries into original programs, and give attribution.	No incorporation of existing code.	Some incorporation of existing code; not attributed properly.	Existing code is incorporated and attributed properly.
Computing Systems			
2-CS-02 Design projects that combine hardware and software components to collect and exchange data.	No hardware used; hardware does not collect or exchange data correctly.	Hardware and software components incorporated; collects and exchanges data inconsistently.	Hardware and software components are incorporated; collects and exchagnes data consistently.
Collaboration			
2-AP-15 Seek and incorporate feedback from team members and users to refine a solution that meets user needs.	Team members did not work together; strengths or suggestions of each member were not incorporated.	Team members ususally worked effectively as a team; strengths and ideas of each member were incorporated somewhat unequally.	Team members worked effectively; the strengths and ideas of each member were incorporated.
2-AP-18 Distribute tasks and maintain a project timeline when collaboratively developing computational artifacts.	Unequal contributions from each team member; project not completed by deadline.	Somewhat equal contributions from each team member. Project completed on time, but may have needed revisions past deadline.	Team members contributed equally; project completed on time.
Debugging			
2-CS-03 Systematically identify and fix problems with computing devices and their components.	Code bugs not identified; little or no documenation of fixes.	Code bugs mostly identified and fixed; adequate documentation of fixes.	Code bugs identified and fixed; extensive documentation of fixes.
Presentation			
1B-AP-17 Describe choices made during program development using code comments, presentations, and demonstrations.	All team members are not able to describe program development and choices.	All team members are able to explain most program development and choices.	All team members are able to extensively explain program development and choices, as well as demonstrate each componenet and line of code.