

A3Q5	Level 4	Level 3	Level 2	Level 1
communication	Nothing assumed, catching errors, anticipating users needs - beyond Level 3	Comments - useful succinct, (block OK), not sparse, block at top (modifications, start date, programmer name, problem description, Notes on anything special or unusual, citations) Readability - code is easy to follow. enough white space	lacking explanation, inline comments, or hard to read, or spelling mistakes on output or Programmer missing or modifications	Any two or more details missing or Citations/borrowed work not acknowledged OR more than 25% of code borrowed
	Excellent	user interface, clear language, no grammar or spelling errors	Adequate	Needs improvement
	User instructions that are clear and can be accessed at any time or Additional options, like premature quit		2 or 3 grammar or spelling errors	User interface not clear, poor language and or spelling and grammar errors
	Advanced class concepts or similar more advanced coding - not required or not taken in class OR new or better implementation of class work. Sprite beyond basics given.	Proper class usages and storing of data - Basic sprite implemented properly		hard coded words and forgot to fix errors and/or basic sprite not working
Knowledge Thinking & Inquiry	Description at top to clarify	IPO proper chart (headings), steps, proper terminology	Unclear steps or incorrect terminology	missing information in columns or steps missing
Thinking & Inquiry	Screen sizes to user systems Make it interactive Colour or design helps user know what stage they are at	Graphics are appropriate and professional looking for Hangman - word list appropriate components - Standard screen size 1280 x 1024	Small graphics, features not intuitive	No graphics, just console that user would not necessarily have. Lost count on number of guesses
		Trace statements - variable name, steps and changes (Table/Chart)	Variable names don't match or missed a step	more confusing, complicate the existing error or do not recognize it
Application	Could be class based	All functions (including user) are efficient and work as expected	Functions work, but confusing or inefficient	Hard coded where a user defined function or built in would suffice
	Highly efficient, best possible, perhaps found a new function or way to complete the task	Code efficiency - nested if's better into function or class.	Nested if's or lack of proper loops - does not take advantage of built in functions	
Flowchart/ IPO or Table (Application)	colouring to add to understanding, using one colour/output another	Flowchart - organized, proper spacing, correct symbols, proper language	Poor spacing, some incorrect symbols, typing errors.	No functions or classes poor organizations, many symbols used incorrectly for the copy/paste attempting to correct, errors in the wrong section or incorrect pointer.

Very Cute! Nice work.