ENCE260 Embedded Systems Assignment: Demonstration Marking Guide **Group # _____**
Member 1 Name: _____ (usercode): _____

Member 2 Name: _____ (usercode): _____

A. Team ready on time and correct names etc. on source modules:

   Marks [0] Not ready when asked to demonstrate in lab session.

   Marks [1] Ready and group member names in source files.

B. At least one git PUSH:

   Marks [0] No source files on gitLab server.

   Marks [1] Source files have been pushed but object/hex/bin files are also included.

   Marks [2] Only source files and a Makefile are included.

C. git README

   Marks [0] The README is insufficient to be able to understand how to play the game.

   Marks [1] The README is sufficient to be able to understand how to play the game.

D. Program compilation and download to the UCFK4:

   Marks [0] Program fails to compile.

   Marks [1] Program compiles and downloads but issues warning messages.

   Marks [2] Program compiles and downloads without warning messages.

E. Communications:

   Marks [0] No Infrared communications.

   Marks [1] Infrared communications but only in one direction.

   Marks [2] Infrared communications in both directions but intermittent or unreliable operation.

   Marks [3] Reliable infrared communications.

F. Display:

   Marks [0] Flicker, jitter, or slowness in updating.

   Marks [2] Clear display with no flicker or jitter.

G. Game Complexity:

   Marks [0] None: an interactive UCFK4 game has not been created.

   Marks [2] Basic: minimal interaction between players.

   Marks [4] Good: some interaction between players, e.g. Rock-Paper-Scissors game.

   Marks [6] Very good: highly interactive with repeated updates, e.g., a ball/missile based game.

   Marks [8] Excellent: highly interactive, and advanced (eg use of sound) or highly original.

H. Game Operation

   Marks [0] None: an operational UCFK4 game has not been created.

   Marks [2] Basic: some functionality, but the game crashes or enters an unrecoverable state.

   Marks [4] Very good: moderate functionality, user interface with instructions, does not crash or enter an unrecoverable state.

   Marks [6] Excellent: extensive functionality, good user interface with instructions, does not crash or enter an unrecoverable state.

TOTAL: _____/25 Sept. 2018

ENCE260 Embedded Systems Assignment: Source Code Marking Guide

A. Formatting

- Is the program indented by the same amount for each block?
- Is whitespace used consistently? Are braces used consistently?

1. Abysmal.
2. Minimal care taken.
3. A bit sloppy.
4. A tidy effort with only a few mistakes.
5. Meticulous (apart from 1 or 2 whitespace inconsistencies).

B. Commenting

- Is there a banner at the top of the file listing the authors' names and what it does?
- Does each function have a comment explaining its purpose?
- Are the comments well formatted and consistently formatted?
- Are the comments relevant and meaningful?
- Is the program over commented? For example, do most lines have a comment?
- Are there any inappropriate comments? For example, 'add one to i'?

1. No comments.
2. Only a few comments or many inappropriate comments.
3. Good attempt at comments but with poor format.
4. Good, well formatted comments.
5. Excellent, well formatted comments.

C. Naming

- Are the variables named consistently?
- Do the variables have meaningful names?
- Are the functions named consistently?
- Do the functions have meaningful names?
- Are the constants named consistently?
- Do the constants have meaningful names?

1. Random or meaningless names.
2. Some variables, functions, and constants have consistent meaningful names.
3. Most, functions, and constants have consistent meaningful names.
4. Almost all variables, functions, and constants have consistent meaningful names.
5. All variables, functions, and constants have consistent meaningful names.

D. Constants

- Does the program use unnamed constants (magic numbers)? This does not include trivial numbers like 1 for incrementing a loop.
- Are dependent constants defined in terms of an independent constant?

1. No use of named constants.
2. Minimal use of named constants.
3. Good use of named constants.
4. Very good use of named constants.
5. Excellent use of named constants. Dependent constants related to independent constants.

E. Structure

- Can you quickly figure out how to use the module?
- Does each module do one thing well, or is it a mishmash of different things?
- Are things (functions, constants) that should be private but are public?

1. No attempt at using a module.
2. An attempt at using a module but of no use to anyone.
3. The module may be useful but is either trivial or hard to use.
4. The module is not trivial and is easy to use.
5. There are multiple modules that are easy to use.

TOTAL: _____/25 Sept. 2018