CS410/510 Languages and Low-Level Programming Spring 2016

Portfolio Assessment Guidelines (Provisional)

This course is defined by a set of twelve learning objectives that specify the knowledge and skills that are expected of students upon successful completion of the course. These objectives are organized in three groups: Bare Metal (1-5), Microkernel (6-8), and Language (9-12). To earn a passing grade for this course, students will be required to construct a *portfolio* of work to provide evidence that they have met each of these objectives. A single piece of work (e.g., the results of a programming project, a paper, or other documentation as appropriate) may be used as evidence for multiple objectives. When a student submits a piece of work for assessment, they should include a narrative statement that identifies the specific objectives that they are targetting and provides, if it is not immediately apparent, a clear and brief explanation of how the work addresses each of those objectives. Each submission will be reviewed and, for each objective identified, scored in to one of the follow categories:

- **N/A**: The submitted work does not address the identified objective.
- **Below**: The submitted work addresses some aspects of the objective, but lacks depth and is generally below the level of the material and examples demonstrated and discussed in class and/or labs.
- **Meets**: The submitted work addresses the objective in an appropriate manner at the level of the material and examples demonstrated and discussed in class and/or labs.
- **Exceeds**: The submitted work addresses the objective in ways that demonstrate depth, insight, originality, or creativity that is very clearly beyond the level of the material and examples demonstrated and discussed in class and/or labs.

Each student should submit work, via the D2L dropbox, for no more than two objectives in any given week, starting in the second week of term and finishing at the end of finals week. Each week is considered to end, and the next to begin, at 11:59pm on Friday. If there are multiple submissions for a given objective, then the submission with the highest score will be used in determining the final grade. If the student has not submitted any work for a given objective by the time of the final, then the student will be assigned a score equivalent to N/A for that objective.

Final grades will be assigned according to the highest applicable rule in the following table:

Grade	CS 410	CS 510
	N, B, M, E are numbers of objectives in the N/A, Below, Meets, Exceeds categories	
A	$E \geq 3 \ \wedge \ (N + B) \leq 2$	$E \geq 4 \ \wedge \ (N + B) \leq 1$
	(Requires at least one Exceeds in each of the three objective groups)	
B	$(N + B) \leq 4$	$(N + B) \leq 2$
C	$B \leq 4 \ \wedge \ N \leq 2$	$B \leq 2 \ \wedge \ N \leq 2$
D	$(E + M) \geq 3 \ \wedge \ N \leq 2$	$(E + M) \geq 3 \ \wedge \ N \leq 2$
F	Other outcomes not covered by the items above	

Other grades (e.g., A-, B+, D, F, etc.) may be used in a way that is consistent with the examples in the table.

Scores for objectives that, in the instructor's judgement, have not been adequately addressed in lectures and lab sessions will not be considered in the calculation of final grades, and a revised version of this document will be distributed to students. Any student who believes that there has not been sufficient coverage for any remaining objectives, or who believes that their situation is adversely/unfairly impacted by any changes to this document should discuss this with the instructor and document their concerns in a brief statement in their portfolio. The instructor will retain final judgement in such matters.

The instructor will maintain a set of grading information for each student on D2L, recording the assigned category for each objective and additional supporting comments or justification. Students may use the tables on the following pages as an offline record of their progress and/or plans.

Objective:	N/A	Below	Meets	Exceeds
1) Write simple programs that can run in a bare-metal environment using low-level programming languages.				
Comments:				
2) Discuss common challenges in low-level systems software development, including debugging in a bare-metal environment.				
Comments:				
3) Explain how conventional operating system features (multiple address spaces, context switching, protection, etc.) motivate the desire for (and benefit from) hardware support.				
Comments:				
4) Develop code to configure and use programmable hardware components such as a memory management unit (MMU), interrupt controller (PIC), and interval timer (PIT).				
Comments:				

Objective:	N/A	Below	Meets	Exceeds
5) Describe the key steps in a typical boot process, including the role of a bootloader.				
Comments:				
6) Describe the motivation, implementation, and application of microkernel abstractions for managing address spaces, threads, and interprocess communication (IPC).				
Comments:				
7) Explain the use and implementation of capabilities in access control and resource management.				
Comments:				
8) Develop programs using the capability abstraction provided by the seL4 microkernel.				
Comments:				

Objective:	N/A	Below	Meets	Exceeds
9) Illustrate the use of a range of domain specific languages in the development of systems software.				
Comments:				
10) Use practical case studies to evaluate and compare language design proposals.				
Comments:				
11) Describe features of modern, high-level programming languages—including abstract datatypes and higher-order functions—and show how they can be leveraged in the construction of low-level software.				
Comments:				
12) Explain how the requirements of low-level systems programming motivate the desire for (and benefit from) language-based support.				
Comments:				