Midterm Exam
CS 111, Principles of Operating Systems
Fall 2016

Name: ____

Student ID Number: ____

This is a closed book, closed note test. Answer all questions.

Each question should be answered in 1-3 paragraphs. DO NOT simply write everything you remember about the topic of the question. Answer the question that was asked. Extraneous information not related to the answer to the question will not improve your grade and may make it difficult to determine if the pertinent part of your answer is correct.

1. What is the difference between the invalid bit in a page table entry and the invalid bit in a translation lookaside buffer entry? What happens in each case if an address translation attempts to use that entry? Is it possible for both to be set? Why?

2. There are many difficult issues that arise due to uncontrolled concurrent executions. Why do we not simply turn off interrupts to prevent such problems from arising? Why not always? Why not just for all critical sections of code?

3. What issues arise in management of thread stacks that do not arise in management of process stacks? Why do these issues arise and what is typically done to handle them? What are the disadvantages of this approach and why is the approach used despite those disadvantages?

4. In MLFQ scheduling, processes are moved from one scheduler queue to another based on their behavior. Each such queue has a particular length of time slice for all processes in that queue. Why might a process be moved from a queue with a short time slice to a queue with a long time slice? How can the operating system tell that the process should be moved?

5. Will binary buddy allocation suffer from internal fragmentation, external fragmentation, both, or neither? If it does suffer from a form of fragmentation, how badly and why? If it does not suffer from a form of fragmentation, why not?

6. What is the purpose of a trap table in an operating system? What does it contain? When is it consulted? When is it loaded?

7. Assuming a correct implementation, do spin locks provide correct mutual exclusion? Are they fair? Do they have good performance characteristics? Explain why for all three of these evaluation criteria (correctness, fairness, performance).

8. What is the purpose of using a clock algorithm to handle page replacement in a virtual memory system? How does it solve the problem it is intended to address?

9. What are two fundamental problems faced by a user level thread implementation that are not faced by a kernel level thread implementation? Why do these problems arise in user level implementations? Why don't they arise in kernel level implementations?

10. What is the difference for a virtual memory system between segmentation and paging? Why might both be used in a single system?