

Steve Martin  
CS 162 Fall 2003  
Midterm I Review Topics

1. What is an operating system?
  - a. Batch systems
  - b. Uniprogramming vs. Multiprogramming
2. Function of an operating system
  - a. Coordinator
    - i. Concurrency
    - ii. I/O
    - iii. Memory
    - iv. Files
    - v. Networks
  - b. Extended Machine
    - i. Present nice interface to messy hardware
3. Processes
  - a. What are they?
  - b. How are they made?
  - c. How are they represented in an OS?
  - d. How many processes can run on a CPU at once?
  - e. What are process states?
4. Exceptions
  - a. What is an interrupt?
    - i. What happens when one occurs?
  - b. What is a trap?
    - i. What happens when one occurs?
  - c. What is an exception?
5. Scheduler
  - a. What is a dispatcher/scheduler?
  - b. What are its responsibilities?
  - c. When does it run?
  - d. How does it maintain control of the CPU?
    - i. How does it put a process on the CPU?
6. Scheduling
  - a. What is a Context Switch?
  - b. What are scheduling goals?
    - i. What is flow time?
    - ii. What is throughput?
    - iii. What is response time?
  - c. How are processes scheduled on the CPU?
    - i. FIFO (First in first out)
    - ii. RR (Round robin)
    - iii. SJF (Shortest job first)
    - iv. SRPT (Shortest remaining processing time)
    - v. SET (Shortest elapsed time)

- vi. FB (Foreground/background)
    - vii. MLFB (Multi-level foreground background)
    - viii. EQ, or MLFQ (Multi-level feedback queues)
  - d. What is the difference between a preemptive and non-preemptive scheduling policy?
  - e. What is an adaptive scheduling policy?
- 7. Concurrency
  - a. What is an independent process?
  - b. What is a cooperating process?
  - c. What is an atomic operation?
- 8. Synchronization
  - a. What is synchronization?
  - b. What is a critical section?
  - c. What is mutual exclusion?
  - d. What is busy-waiting? (or 'spinning')
  - e. What is a semaphore?
    - i. What are its properties?
  - f. What is a mutex?
  - g. What is a condition variable?
    - i. How are these different from semaphores?
  - h. What is a monitor?
    - i. What is a Mesa-style monitor?
  - i. What is important about test-and-set and swap?
- 9. Threads
  - a. What is a thread?
  - b. What's the difference between a thread and a process?
  - c. What are the advantages of threads? Disadvantages?
  - d. Can we use both?
- 10. Deadlock
  - a. What is deadlock?
    - i. What are the four conditions of deadlock?
  - b. What are examples of preemptible and non-preemptible resources?
  - c. What are the two approaches to the deadlock problem?
  - d. What are ways to prevent deadlock?
    - i. What is a 'safe state'?
  - e. Bankers Algorithm
  - f. Graph Algorithm
    - i. What is a resource allocation graph?
  - g. What is Rollback? What is a Checkpoint?
- 11. Linkers/Loaders
  - a. What are the components of a program in memory?
  - b. When compiling, what are:
    - i. Segment tables
    - ii. Symbol tables
    - iii. Relocation tables
  - c. What are the 3 steps in a linker/loader?

## 12. Dynamic Storage Allocation

- a. What is a stack?
- b. What is a heap?
- c. Heap allocation schemes
  - i. Best-fit
  - ii. First-fit
  - iii. Next-fit
- d. What is internal fragmentation? What is external fragmentation?
- e. What is garbage collection?
  - i. How does it work? (give one method)
  - ii. Why is it expensive?

## 13. Multiprogramming with Shared Memory

- a. What is a Virtual Address? What is a Physical Address?
- b. What is base and bound relocation?
  - i. MFT – fixed partitions
  - ii. MVT – variable partitions
  - iii. What are the advantages and disadvantages of each?
  - iv. What does base-and-bounds relocation add to context switches? Memory lookups?
- c. Segmentation
  - i. What is a segment table?
  - ii. Who needs one?
  - iii. What is a segment table entry? What does it look like?
    1. What is each field for?
  - iv. What is a segment fault?
  - v. What exactly happens when a program tries to read from memory?
  - vi. What does segmentation add to context switches?
  - vii. What is a core map?
- d. Paging
  - i. What is a page table?
  - ii. Who needs one?
  - iii. What does a page table entry look like?
    1. What is each field for?
  - iv. What is a page fault?
  - v. What exactly happens when a program tries to read from memory?
  - vi. What does paging add to context switches?
  - vii. Can you page out anything in memory?
  - viii. How big is a page table?
  - ix. What is 2-level paging?
    1. How does this work?
- e. How does segmentation compare to paging?
  - i. What are the advantages?
  - ii. What are the disadvantages?
- f. How do these methods provide protection among processes?
- g. How would you combine segmentation with paging?
  - i. Why would you want to?