Part I: Multiple Choice

Each question is worth 2 points (there are 40 questions). Indicate your choice by putting an X in the appropriate circle.

1. (2 points) In the process state transition diagram, the transition from the READY state to the RUNNING state indicates that:
   ( ) A. A process was preempted by another process.
   ( ) B. A process has blocked for a semaphore or other operation.
   ( ) C. A process is done waiting for an I/O operation.
   ( ) D. A process was just created.

2. (2 points) Which of the following is shared between all of the threads in a process? Assume a kernel-level thread implementation.
   ( ) A. Register values
   ( ) B. File descriptors
   ( ) C. Scheduler priority
   ( ) D. Local variables

3. (2 points) Which of the following is not true?
   ( ) A. Shortest Remaining Time Next is the best preemptive scheduling algorithm in terms of turnaround time.
   ( ) B. Priority scheduling can suffer from starvation.
   ( ) C. Lottery scheduling is preemptive.
   ( ) D. Multi-level feedback queues guarantee equal time to all processes.

4. (2 points) A critical region is:
   ( ) A. The part of a program in which shared data is accessed.
   ( ) B. The most important part of a program.
   ( ) C. The part of the kernel that interfaces directly to the device controllers.
   ( ) D. The part of a program in which a bug would cause the program to exit.

5. (2 points) Which of the following is not used for synchronization?
   ( ) A. The banker’s algorithm
   ( ) B. The bakery algorithm
   ( ) C. Busy waiting with test and set
   ( ) D. Monitors

6. (2 points) Which of the following is not true of virtual memory?
   ( ) A. It allows more efficient use of memory.
   ( ) B. It requires hardware support.
   ( ) C. It reduces the need for relocatable code.
   ( ) D. It requires the use of a disk or other secondary storage.

7. (2 points) Which of the following is not usually stored in a two-level page table?
   ( ) A. Virtual page number
   ( ) B. Physical page number
   ( ) C. Dirty bit
   ( ) D. Reference bit
8. (2 points) Which of the following paging algorithms is most likely to be used in a virtual memory system?
   O A. FIFO
   O B. Second chance
   O C. Least Recently Used
   O D. Least Frequently Used

9. (2 points) The purpose of a TLB is:
   O A. To cache page translation information
   O B. To cache frequently used data
   O C. To hold register values while a process is waiting to be run
   O D. To hold the start and length of the page table

10. (2 points) Which of the following is not true about segmented memory management?
    O A. Segment length must be a multiple of the page size.
    O B. Segmentation allows multiple linear address spaces in one process.
    O C. Segmentation can be used with paging to keep segments partially resident in memory.
    O D. A segment can be read-only for one process and read-write for another.

11. (2 points) System calls:
    O A. Provide a rich and flexible API for software developers.
    O B. Often change dramatically between different releases of an operating system.
    O C. Protect kernel data structures from user code.
    O D. Allow the operating system to optimize performance.

12. (2 points) What is the main difference between traps and interrupts?
    O A. How they are initiated
    O B. The kind of code that's used to handle them
    O C. Whether or not the scheduler is called
    O D. How the operating system returns from them

13. (2 points) Buffering is useful because
    O A. It makes it seem like there's more memory in the computer.
    O B. It reduces the number of memory copies required.
    O C. It allows all device drivers to use the same code.
    O D. It allows devices and the CPU to operate asynchronously.

14. (2 points) The main advantage of DMA is that it
    O A. Increases system performance by increasing concurrency.
    O B. Allows the CPU to run faster.
    O C. Reduces the traffic on the data bus.
    O D. Removes the requirement that transfers be properly aligned.

15. (2 points) Which of the following disk seek algorithms would be the best choice to implement in a system that services an average of 5 disk requests per second?
    O A. First Come First Served
    O B. Shortest Seek Time First
    O C. SCAN
    O D. C-SCAN
16. (2 points) Which of the following disk seek algorithms has the most variability in response time?
   - A. First Come First Served
   - B. Shortest Seek Time First
   - C. SCAN
   - D. C-SCAN

17. (2 points) A typical hard drive has a peak throughput of about
   - A. $2 \times 10^3$ bytes per second.
   - B. $2 \times 10^6$ bytes per second.
   - C. $2 \times 10^7$ bytes per second.
   - D. $2 \times 10^8$ bytes per second.

18. (2 points) RAID is a way to:
   - A. Increase hard drive latency and performance.
   - B. Increase hard drive performance and decrease cost.
   - C. Increase hard drive reliability and performance.
   - D. Increase hard drive reliability and decrease cost.

19. (2 points) Which of these would not be a good way for the OS to improve battery lifetime in a laptop?
   - A. Shut down the hard drive until it's needed.
   - B. Reduce the processor speed while it's idle.
   - C. Turn off power to the memory.
   - D. Shut down the modem when it's not connected.

20. (2 points) Which of the following is not included in an inode in Linux?
   - A. File owner
   - B. File name
   - C. File modification date
   - D. Pointer to the first data block

21. (2 points) Which of the following is not true of a file system using a file allocation table (FAT)?
   - A. A bitmap is used to track free blocks.
   - B. Index nodes (inodes) are unnecessary.
   - C. File size is limited only by the amount of free space.
   - D. Block $n$ of a file can be read without first reading blocks 0 through $n-1$.

22. (2 points) Which of the following file system types requires the fewest 4 KB file blocks to store a 16 KB file?

23. (2 points) Given a block size of 4 KB and a disk that holds 100 GB of data (using 32 bit disk addresses), how big can a file be if you have a single index block in an indexed file management scheme?
24. (2 points) What purpose does a journal serve for a file system?
   - A. Contains all of the metadata structures for the file system.
   - B. Records all reads and writes performed on the file system.
   - C. Ensures that on-disk structures remain consistent.
   - D. Guarantees that data written to disk is never lost.

25. (2 points) Which of the following is true of memory-mapped files?
   - A. Memory-mapped files can be used for process synchronization.
   - B. It’s impossible to increase the size of a memory-mapped file.
   - C. They eliminate the need to transfer file data to and from disk.
   - D. Programs need not make read and write system calls for them.

26. (2 points) Which of the following is not true about the file system directory hierarchy in Linux?
   - A. Directories are stored in the same way as files.
   - B. Directory entries are a fixed size.
   - C. It can be represented as a directed graph.
   - D. A single file can have multiple names in the same directory.

27. (2 points) Which of the following is not a benefit of using larger file blocks in a file system using indexed allocation (such as Linux)?
   - A. Better disk space utilization.
   - B. Faster access to a randomly-selected block of a large file.
   - C. Higher average transfer rate for file data.
   - D. Less disk space consumed for metadata structures.

28. (2 points) Which of the following is an advantage for having disk quotas?
   - A. It ensures that the free space on a disk is distributed equitably to all users.
   - B. It stops a malicious process from filling the disk.
   - C. It prevents a process from consuming too much disk bandwidth.
   - D. It makes it easier to group a user’s files together.

29. (2 points) In UNIX (or DLXOS), how many system calls must a process make to create a new file, write three blocks to it, and close it?
   - A. One
   - B. Three
   - C. Four
   - D. Five

30. (2 points) The three main goals of computer security are:
   - A. Detection, response, and correction
   - B. Confidentiality, performance, and reliability
   - C. Confidentiality, integrity, and availability
   - D. Detection, protection, and access control

31. (2 points) Which of these statements about public-key encryption is false?
   - A. Public-key encryption doesn’t require a secret key
   - B. Public-key encryption can be used to authenticate the sender of a message.
   - C. Public-key encryption is slower than symmetric-key (shared-key) encryption.
   - D. Decryption and encryption are the same function for public-key encryption.
32. (2 points) Moving from a 64-bit key to a 128-bit key for symmetric-key encryption makes brute-force decryption (guessing the key) how much more difficult? 
   - A. 2\times hard
   - B. 64\times hard
   - C. 64^2\times hard
   - D. 2^{64}\times hard

33. (2 points) One-way functions are used in:
   - A. Public-key encryption
   - B. Symmetric-key encryption
   - C. Authentication
   - D. Permission revocation

34. (2 points) The Linux file system uses a form of which protection mechanism?
   - A. Access control lists
   - B. Capabilities
   - C. Protection domains
   - D. Security by obscurity

35. (2 points) A trojan horse is:
   - A. A program that does something beyond its apparent purpose.
   - B. A program that shuts down if certain conditions are not met.
   - C. A special password that's built into the system software.
   - D. A computer virus that's spread by email.

36. (2 points) Which of the following is not a good technique for ensuring that passwords are not compromised?
   - A. Storing a hashed version of passwords.
   - B. Forcing users to pick passwords that are not in the dictionary or other lists of common words.
   - C. Adding salt to the password file.
   - D. Storing the password file where only the operating system can read it.

37. (2 points) Which of the following is not a technique often used to break into computer systems?
   - A. Exploiting buffer overflows
   - B. Replaying login sessions
   - C. Hijacking the compiler
   - D. Sending malicious code via email

38. (2 points) The DLX code uses which scheduling algorithm by default?
   - A. Round robin
   - B. First come, first served
   - C. Lottery
   - D. Priority

39. (2 points) How does the vanilla version of DLXOS (before you made changes) keep track of open files?
   - A. Global open file table, at most one entry per UNIX file.
   - B. Global open file table, multiple entries possible per UNIX file.
   - C. Per-process open file table.
   - D. Dynamically-allocated structures to keep track of open files.
40. (2 points) The DLXOS system call Spawn() returns:
   ○ A. The exit status of the child process.
   ○ B. The current status of the child process.
   ⬜ C. The process ID of the child process.
   ○ D. A pointer to the PCB of the child process.
Part II: Short Answer

Each question is worth 12 points (there are 10 questions). Show all of your work—you can get partial credit for partially correct answers, and may not get full credit for answers without any work.

1. (12 points) Disk Seek Algorithms

A disk request queue has requests for blocks on the following cylinders (ordered by time of arrival):
1773, 5041, 4502, 6881, 872, 4823, 8994, 4478, 9235

The disk has 10,000 cylinders numbered 0 through 9999. The disk head is currently at cylinder 4617 and is moving towards cylinder 9999.

a. (8 pts) Calculate the total seek distance for each of the following disk seek algorithms: FCFS, SSTF, LOOK, C-LOOK.

- **FCFS:**
  \[
  \text{Seek Distance} = (4617 - 1773) + (5041 - 1773) + (4502 - 5041) + (6881 - 4502) + (8994 - 872) + (4478 - 4823) + (9235 - 4997) + (1773 - 872) \]
  \[
  = 15243 \text{ cylinders}
  \]

- **SSTF:**
  \[
  \text{Seek Distance} = (4617 - 1773) + (5041 - 1773) + (4502 - 5041) + (8994 - 6881) + (9235 - 8994) + (1773 - 872) \]
  \[
  = 12978 \text{ cylinders}
  \]

- **LOOK:**
  \[
  \text{Seek Distance} = (4502 - 4617) + (6881 - 5041) + (8994 - 6881) + (9235 - 8994) + (1773 - 872) + (4478 - 4823) \]
  \[
  = 16611 \text{ cylinders}
  \]

- **C-LOOK:**
  \[
  \text{Seek Distance} = (4823 - 4502) + (5041 - 4823) + (6881 - 5041) + (8994 - 6881) + (9235 - 8994) + (1773 - 872) + (4478 - 4823) \]
  \[
  = 16611 \text{ cylinders}
  \]

b. (4 pts) Modern disk drives don't provide an easy way to calculate a cylinder number from a disk block address. How could you adapt the above algorithms to operate without knowing the cylinder on which a given disk block is located?

From facts, the number of cylinders × the number of surfaces on the disk. If you find out the # of blocks in the disk, you can find out the # of surfaces on the disk by using cancellation of conversion units.
2. (12 points) Deadlock
   a. (4 pts) List the four conditions that must hold for deadlock to occur.
      1. Mutual exclusion
      2. Preemption
      3. Hold and wait
      4. Circular waiting

   b. (4 pts) List the four strategies for dealing with deadlock.
      1. Make sure all processes hold no resource for more than 1 resource.
      2. Don’t force a process to give up its resources
      3. Don’t let a process wait too long
      4. Let processes request resources in new ways.

   c. (2 pts) Draw a resource allocation graph with three processes (1–3) and three resources (A–C) where the two processes are deadlocked.

   d. (2 pts) Draw a resource allocation graph with three processes (1–3) and three resources (A–C) where both resources are allocated but the processes are not in deadlock.

END OF THE FIRST HALF OF THE EXAM
3. (12 points) Memory Management
Suppose that your memory management hardware supports neither reference bits nor modified (dirty) bits, but does support valid/invalid bits and the ability to make pages read-only or read-write. Explain how you could use this hardware to implement a software solution that will emulate the functionality of the missing dirty and reference bits. Your solution should be as efficient as possible.

The basic functionality of the dirty bit and reference bit is to tell the OS if the block has been allocated or used. If my hardware only supports valid/invalid and the ability to make pages read-only/read-write only, then I can program the valid/invalid bit to tell the OS: 0 = read-only and 1 = read-write only.

If a block contains valid bit 0, then that bit tells the OS that the block has been referenced, hence imitating the reference bit. If a block contains valid bit 1 (invalid), the block is read-write only, hence imitating the dirty bit.

Diagram:

```
<table>
<thead>
<tr>
<th></th>
<th>read</th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>valid</td>
<td>0</td>
<td></td>
</tr>
<tr>
<td>dirty</td>
<td>1</td>
<td></td>
</tr>
</tbody>
</table>
```

Data
4. (12 points) File Systems
Describe how to implement a file system consistency checker for Linux that verifies that only free blocks are marked free, only allocated blocks are marked allocated, and each allocated block is allocated to exactly one file.

I would implement a global File Allocation Table (FAT) to keep track of allocated and unallocated blocks. By default, as the system starts up, each block in the FAT starting from block 1 (block 0 is reserved for the superblock) has the value "O" in it. The "O" tells the OS that that specific block is free.

When a file is opened, I would implement a simple code that checks if the file size can fit the file allocation table. By default, it would make the block 4KB. That block would contain the start address of the file. If the file is larger than 4KB, then more blocks are needed. The additional blocks would contain the block number of the next block, with the last block containing "-1", signifying the end of the file.
5. (12 points) Security
Based on your knowledge of UNIX and the things we have discussed in this class, describe two realistic ways that someone might be able to get another person's login information (username and password) for one of the UNIX systems on this campus without just being told the information by anyone (the user or someone else). Don't simply list the names of known attacks—explain how they might actually work. Be as specific as possible.

One way is to write a program that bypasses the system call that prompts the login screen. Usually, in most UNIX, the OS asks the user to type in his/her login info. Once you bypass it, you can write a piece of code that indicates how the real OS gets the user's login information. For example, you can program a “fake” login screen or prompt that asks the user for user information. The syntax is exactly the same as the real login process. Once the user thinks it's the real thing and enters his/her information, you can make your program exit and save the information. Once it closes, prompt the user something unrelated and display the real login screen UNIX provides.

Another way is to break into the UNIX system to retrieve the password file in the /etc directory. The only thing that protects the directory is the set permissions in the access list of UNIX. Once you break through the permissions, you have access to the user's directory. Now you can find the file the password is stored in.
6. (12 points) File System Snapshots
Many file systems have the ability to take a snapshot—a copy of the entire file system at a particular point in time. After the snapshot is taken, the file system continues to run, with users reading and writing the most recent version of the file system. Something similar is implemented with the OldFiles directory on unix.ic, which always has a copy of your files as of yesterday.

a. (4 pts) List two distinct uses for snapshots.

- If you accidentally delete a file you didn't want to be removed, you can access that file in the OldFiles directory or through a snapshot of the file.

- Another use of snapshots is the availability to compare old files with new ones. This is useful when programming. If you need older files for reference, snapshots can be useful.

b. (8 pts) Briefly describe how you might implement snapshots efficiently. Keep in mind that you want to take snapshots quickly (a few seconds at most) and don't want to keep multiple copies of files that don't change much.

I would first implement a global table for the filesystem. It looks similar to a buffer, but with different fields. The table would save the start and end address of the files that were created. Once the global table is full, the snapshot process is started. It would copy its fields (addresses of the files) to another table of the same size. That table would be the snapshot of all the files created. The old global table would just be written over with more new files.
7. (12 points) Scheduling
Our operating system is using a scheduling algorithm that gives a higher priority to processes that have used the least CPU time in the recent past.

a. (4 pts) Why will this algorithm favor I/O-bound processes but not completely starve CPU-bound processes?

The scheduling algorithm would favor I/O bound processes because I/O bound processes contain many blocks. It doesn't use much CPU time if the processes frequently stop, due to interrupts, waits, etc. Since it doesn't use much of the CPU time, CPU-bound processes won't completely starve, and will be executed.

b. (5 pts) Give a formula for priority that would result in a good schedule for this algorithm. Your algorithm should allow a process to move from CPU-bound to I/O-bound and back over longer intervals. Assume you have a function \( \text{cpu_usage}(\text{int } i) \) that returns the total CPU usage (up to the current time) of process \( i \). Hint: to get the CPU time used in the interval \( (t, t+\Delta) \), you can call this at \( t \), call it again at \( t + \Delta \), and take the difference.

\[
\begin{align*}
\text{CPU usage} &= \text{cpu_usage}(i) \\
\text{Curri_time} &= (t+\Delta) - t \\
\text{CPU_usage} &= \text{cpu_usage}(i+\Delta) \\
\text{if} (\text{CPU_usage} < \text{Curri_time}) \{ \\
\quad \text{Process priority}++ \\
\} \quad \text{else} \\
\quad \text{process.priority}--
\end{align*}
\]

c. (3 pts) What scheduling algorithm would you choose to actually implement this scheduler given the values generated from the previous function? Justify your choice.

Since the scheduler computes a priority for I/O bound processes and CPU bound processes, usually, an I/O bound process would have a higher priority. A priority scheduler would seem logical, since it favors the I/O processes first, then moves on to CPU bound processes.
8. (12 points) Segmentation
The Intel Pentium uses both segmentation and paging to resolve addresses. It uses segmentation to generate a 32-bit virtual address, and uses two-level page tables to translate the virtual address into a physical address.

a. (8 pts) Show how an address consisting of \textit{(segment\_number, offset\_within\_segment)} is translated into a physical address in such a system.

b. (4 pts) How does the use of both segmentation and paging reduce memory usage requirements? Are there other benefits to the use of segmentation and paging?

Usually the use of both segmentation and paging does not use physical memory. Instead, it uses virtual memory which is embedded in software that mirrors all of the memory usage there. By using segmentation and paging, you have more memory to use since physical memory is rather small compared to virtual memory. By using the two-level page tables, you can reference more blocks, increasing the size of memory.
9. (12 points) I/O System Performance

Your computer system has a PCI bus that runs at 500 MB/sec, and two controllers. One controller, which does not support DMA, interrupts the CPU every time a 32-bit word of data is ready. The other controller supports DMA.

a. (4 pts) Assume that the CPU takes 20 μs to handle an interrupt, including the transfer of the word of data. How many bytes per second can the OS transfer via the non-DMA controller? How long does it take to transfer a 1 KB disk block using this controller?

\[
\frac{32 \text{ bit word}}{\text{sec}} \times \frac{500 \text{MB}}{\text{sec}} \times \frac{1 \text{kb}}{1 \text{MB}} = \frac{1.5 \times 10^6 \text{B}}{\text{sec}} \times \frac{1 \text{B}}{8 \text{bit}} = 1.875 \times 10^5 \text{Byte/sec} = 4 \times 10^5 \text{byte/sec}
\]

b. (4 pts) Assume that it takes 200 μs for the operating system to manage the details associated with a DMA request (including any interrupts, but excluding time spent transferring the data itself). If each word transferred by the DMA controller must travel over the PCI bus twice, how large must a request be to make DMA faster than word-by-word transfer?

\[
\frac{5 \text{MB}}{\text{sec}} \times \frac{1 \text{sec}}{1 \text{MB}} \times \frac{200 \mu \text{s}}{1 \text{sec}} = 10000 \text{ MB/sec}
\]

c. (4 pts) Is there any advantage to allowing the DMA controller to have multiple outstanding requests? Why or why not? How would this affect operating system design?

The only advantage is that multiple words of data would be processed, making the OS more productive. The OS can process more memory, making multiple functions executable. This would slow the performance of the operating system, taking up more CPU usage.
10. (12 points)

a. (6 pts) Explain how a message is sent from Alice to Bob using public-key encryption (as done in PGP). You should describe any encryption and decryption operations that are performed.

Alice would give Bob a public key to encrypt her message to Bob. Once Bob receives the encrypted message, he can use the private key to decrypt her message. Usually Alice would encrypt the message with a function that returns a hash. That hash is sent to Bob, who has the same key. He uses a private key to decrypt the hash with the decrypt function to get the message in text form.

b. (3 pts) What information (other than the message itself) must Alice and Bob possess for this communication?

They both would need the private and public key to encrypt and decrypt the message. They also both need the hash returned by the encrypt and decrypt function.

c. (3 pts) Describe one method by which an intruder (Eve) could compromise this message exchange.

Since the only method to encrypt or decrypt the message, you would need the private key to find the decrypted message. One way is to steal the hash used for the decrypt/encrypt function. From there, Eve can use the key to decrypt the hash to get the message.