Question/Answer Sheet COMPSCI 340 & SOFTENG 370

Surname: ..... Forenames:.....

ID: .....

THE UNIVERSITY OF AUCKLAND

SECOND SEMESTER, 2009
Campus: City

COMPUTER SCIENCE & SOFTWARE ENGINEERING

Operating Systems

(Time allowed: TWO hours)

NOTE:

Attempt ALL questions.

Answer the questions in the spaces provided.

Marks for each question are shown and total **100**.

For markers only:

Question 1	*/10*	*Question 5*	*/9*
Question 2	*/12*	*Question 6*	*/14*
Question 3	*/14*	*Question 7*	*/*
Question 4	*/12*	*Question 8*	*/*
		Total	

CONTINUED

ID: .....

1. Concurrency [10 marks]

(a) What is a spinlock (or busy wait)?

A spinlock is a lock which repeatedly checks to see if the lock is available until it becomes available.

(2 marks)

(b) What is the major disadvantage of a spinlock?

It consumes CPU cycles preventing other threads from running on that CPU. No useful work is carried out on the CPU until the lock becomes available.

(2 marks)

(c) Most operating systems use an adaptive lock. An adaptive lock uses a spinlock when trying to access a resource locked by a currently-running thread, but the adaptive lock will sleep if the thread holding the resource is not currently running.
What advantage does an adaptive lock have over a simple spin lock?

If the current holder of the lock is not running, it will not unnecessarily consume CPU cycles.

(2 marks)

(d) What advantage does an adaptive lock have over a lock which always sleeps if the lock is unavailable?

If the lock is held for a short period of time (and it must be on another CPU) then there is no context switch and the overall waiting time is minimized.

(2 marks)

(e) On a uniprocessor (a machine with only one processor/core) an adaptive lock is the same as a lock which always sleeps if the lock is unavailable. Explain why this is true.

On a uniprocessor, only one thread can run at a time. If a thread calls an adaptive lock, the thread holding the lock cannot be running. Hence the requesting thread will immediately go to sleep.

(2 marks)

ID: .....

2. Process Priorities [12 marks]

(a) It is common for processes with better priorities to be scheduled before processes with worse priorities. Describe another way the priority of a process can be used when scheduling processes?

Processes with better priorities could get selected more frequently than those with worse priorities.
Processes with better priorities run for longer than those with worse priorities.....
.....

(2 marks)

(b) Explain how process priorities can lead to starvation or indefinite postponement.

If processes with better priorities are always selected before processes with worse priorities and there are always processes with better priorities then the worse priority processes will never run. This is indefinite postponement. .....

(2 marks)

(c) Describe a solution to indefinite postponement.

Priority aging, where priorities improve over time. So no process stays with a poor priority for ever. .....
.....

(2 marks)

(d) There are three processes: Process A with priority 1, Process B with priority 2, and Process C with priority 3. (A higher number means a better priority.) There is also one resource X that can only be used by one process at a time.
Describe how priority inversion could happen with these processes.

A runs and gets X. B runs and prevents A running because of its better priority. C runs and waits for X. X cannot be released by A because B is running. .....
.....
.....
.....

(4 marks)

(e) Explain how you could prevent the priority inversion from occurring.

When C requests X, if its priority is higher than the priority of the process currently using X, that process has its priority improved until it releases the resource. It is given the same priority as C. .....
.....

(2 marks)

CONTINUED

ID: .....

3. Deadlock [14 marks]

(a) What are Havender's four conditions for deadlock?

There is a cycle of processes waiting for resources held by other processes in the cycle.
The resources cannot be shared.
Only the owner can relinquish the resource.
A process can hold a resource while waiting for another one. .....
.....

(4 marks)

(b) How does resource ordering prevent deadlock?

It stops a cycle of waiting processes from being formed. A process can't wait for a resource earlier in the order if it already holds a resource later in the order. .....
.....

(2 marks)

(c) Deadlock can also be prevented by allowing some processes to wait and forcing others to restart when they request a busy resource. Describe an algorithm that uses this technique.

Wait-die forces younger processes to restart if the resource they want is held by an older process. Older processes are allowed to wait. .....
.....
.....
.....

(3 marks)

(d) Dijkstra's Banker's Algorithm is a technique to avoid deadlock. Requests are not allowed if it moves the system to a non-safe state.
What does it mean for a state to not be safe?

A state is not safe if it can't be proven that all processes will eventually finish from that state. In other words deadlock is possible (not definite). .....
.....

(2 marks)

ID: .....

(e) There are three copies of a resource R in a system. There are two processes P and Q with the following maximum resource requirements for completion:

P needs 2 Rs
Q needs 3 Rs

Using the Banker's algorithm explain why the third request in the following table is not safe:

command	**state (allocation of Rs)**	**safe?**
P requests R	P1, Q0	safe
Q requests R	P1, Q1	safe
Q requests R	P1, Q2	not safe

P may not be able to complete because it may need a second R which is currently not available. Similarly Q may not be able to complete because it may need a third R. Therefore we can not guarantee that eventually all requests will be met and all processes can finish.

.....

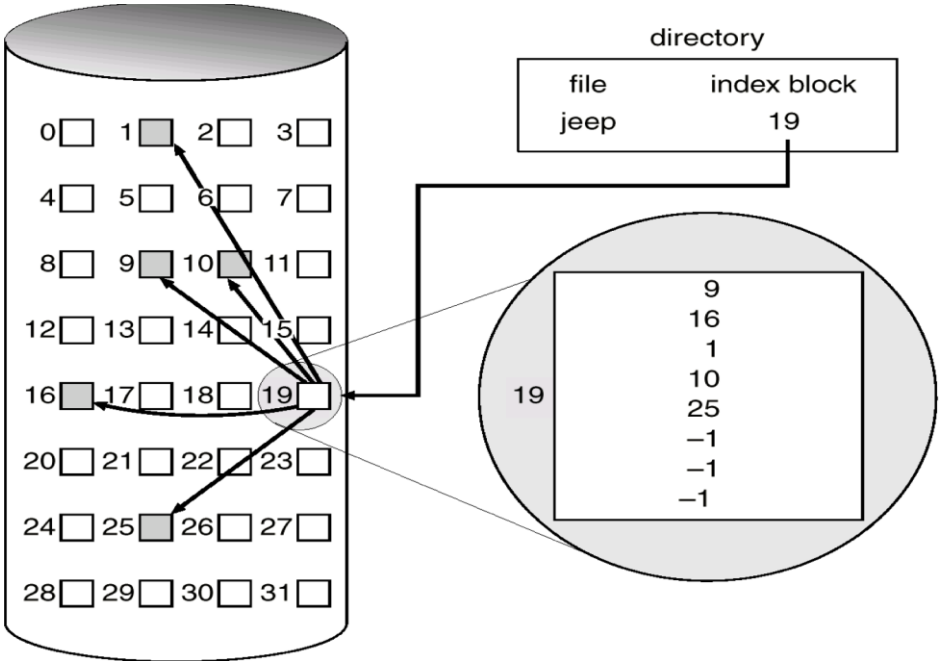
.....

(3 marks)

ID: .....

4. File Systems [12 marks]

(a) There are different ways of keeping track of the blocks allocated to a file on a disk. This diagram from the lecture notes and the textbook shows one of them.



Explain this diagram. Also explain how the third block in the `jeep` file is found after the `jeep` file has been opened.

This is indexed allocation. Each file has an index block. The index block stores the numbers of the blocks allocated to the file in order. After the jeep file is opened its index block (19) is read into memory. The third number (1) corresponds to the third block of the file. So block 1 on this device holds the data for the third block of the jeep file.

.....
.....
.....
.....
.....
.....
.....
.....
.....

(6 marks)

CONTINUED

ID: .....

(b) Calculate the maximum file size in a file system which uses three levels of index blocks. The third level points to actual blocks. The second level points to third level index blocks and the first level points to second level index blocks. There is one first level index block for each file. Each block is 4K bytes in size and each block address occupies 8 bytes.

Show your working. You may leave your answer as a power of 2 bytes.

```
Each index block can hold 2^12 / 2^3 = 2^9 or 512 block numbers.
So the maximum number of addressable blocks per file is:
2^9 * 2^9 * 2^9 = 2^27 blocks
So the maximum file size is:
2^27 * 2^12 = 2^39 bytes or 512 GB.
```

(6 marks)

ID: .....

5. Distributed File Systems [9 marks]

(a) Distributed file systems can be either stateless or stateful. Which type of system deals better with server crashes and restarts? Explain why.

The stateless system deals better with server crashes and restarts because no state information is lost by the crash. When a server restarts any new requests carry all required information to carry out the request. Similarly the server does not need to recreate a list of clients accessing its files.....
.....
.....
.....

(4 marks)

(b) Messages are not sent across the network to the file server for each read and write access when using AFS. How does AFS deal with most read or write requests? How does this affect the performance of the file system?

AFS tries to cache the entire file locally so that all read/write requests go to the local copy. Caching should speed up access to the file. .....
.....
.....
.....

(3 marks)

(c) When talking about file system consistency semantics what is session semantics?

The process gets a copy of the file when it is opened and changes are not visible to other processes until a file is closed. .....

(2 marks)

ID: .....

6. Assignment 2 [14 marks]

(a) Assignment 2 asked you to complete a user-level thread system.
What is a user-level thread?

A thread implemented in user-level code. In the case of assignment 2 the operating system had no knowledge of the threads running inside the example processes. The operating system only saw one thread per process and scheduled that thread, all further thread switching was done within the process itself.

(2 marks)

(b) Briefly explain why each thread needs its own stack.

Each thread will be making its own function calls and hence needs space for return addresses, parameters and local variables. .....
.....

(2 marks)

(c) A thread was defined with the following struct in C:

```
typedef struct thread {
    int tid;             // thread identifier
    void (*start)();     // the start function
    jmp_buf environment; // saved registers
    enum state_t state;  // the state
    struct thread *prev; // pointer to the previous thread
    struct thread *next; // pointer to the next thread
} *Thread;
```

What was the `environment` field and how was it used in the assignment?

The environment field held the registers when a thread wasn't running. The setjmp function stored the current values of the registers into environment when the thread was being suspended and the longjmp function restored the values from environment back into the registers when the thread was about to continue running. .....
.....
.....
.....
.....

(4 marks)

CONTINUED

ID: .....

(d) Two different signal types were used in the assignment. The signals were the SIGUSR1 and the SIGVTALRM signals. Briefly describe how the signals were used and what they were used for.

To associate a stack with each thread. When a new thread was created, memory for a stack was allocated and the sigaltstack function was used to associate that stack with the SIGUSR1 signal handler. When the signal was received, the handler ran on the new stack and then the registers were saved with a call to setjmp which connected the new stack to the new thread.
To handle clock interrupts. The SIGVTALRM signal was called repeatedly every 20 milliseconds. This signal was used to invoke the scheduler and pass control to the next thread. .....
.....
.....
.....

(6 marks)

ID: .....

7. Access Control [9 marks]

(a) A kernel process P1 is the reference monitor for user process P2.
User process P2 has requested, and has obtained, read access to file F1.
User process P2 also has obtained write access to file F2.
In the space below, draw an access matrix for this system. Briefly discuss the non-blank entries in your matrix. Be sure to indicate the domains that P1 and P2 are running in.

P1 is in domain D1, P2 is in domain D2. I have made it the owner of files F1 and F2, because it's pretty clear that P2 can't be the owner (since it had to obtain access from P1 for these files). The kernel P1 has control rights over all domains, including its own. The user process P2 is able to read F1, and it is able to write F2, but it has no other rights.

D\O	F1	F2	D1	D2
D1	o	o	c	c
D2	r	w	-	-

(5 marks)

(b) You are a systems administrator for a very large bank. The bank president has asked you, urgently, to revoke all access rights for a formerly-trusted employee -- a systems operator -- who is suspected of fraud. The immediate worry is that this employee might overwrite entire file systems, causing large losses to the bank, in an attempt to obliterate all traces of their prior fraudulent activity. Authentications for many of your bank's systems are handled through Kerberos. In the space below, explain the relevant TOCTTOU vulnerability of Kerberos.

.....
.....
.....

(4 mark)

ID: .....

8. Devices [14 marks]

(a) Buffers are used in a variety of ways in operating systems. Describe two ways in which buffers are used when dealing with devices. Explain how the buffers help in each case.

Buffers can be used to cope with differences in speed between devices or between a process and a device. The buffer helps by receiving data at the speed of production (by the device or process) and making it available
They can also be used to cope with differences in data format sizes. E.g. When transmitting data from a disk over a network, one buffer can be of the disk block size and data is moved from this buffer to another buffer which could be packet size. .....
Buffers can also be used to preserve copy semantics so that data is moved to a buffer when it is written to a device, and even if the original data is modified the data sent to the device is the actual data intended by the write call. .....
.....
.....
.....
.....
.....

(8 marks)

(b) Unix rather simplistically separates devices into two main categories, block devices and character devices. Give an example of each type of device and explain the major difference between the two categories.

A disk is usually a block device. Data transfer between disks and memory is done in blocks. On Unix a block device uses the block/buffer cache. .....
A keyboard is an example of a character device. The device produces a small number of bytes on each interaction. But the most important thing is that it doesn't use the block/buffer cache. .....
.....
.....

(6 marks)