415.340SC
415.341ST

THE UNIVERSITY OF AUCKLAND

EXAMINATION FOR BA BSc ETC 1997

COMPUTER SCIENCE

Operating Systems

(Time allowed: TWO hours)

INSTRUCTIONS:

Answer any SIX questions. The total mark for each question is 20; the total for the paper is 120 marks.

QUESTION 1.

(a) A user interface management system (UIMS) in a computer system with a graphical user interface (GUI) is required to control and coordinate the transfer of information between the system and the person using it. Consider an interface showing several windows, each associated with a different process running in the computer. State briefly how the UIMS must handle the input signals it receives from the person and the output signals it receives from the processes in order to satisfy this requirement.

(b) When a window displayed on a GUI is moved, many systems require that processes which own windows wholly or partly revealed by the move must redraw their windows.

(i) What information must the UIMS maintain in order to determine which processes must be notified to redraw their windows, and in which order ?

(ii) What is the *simplest* way of using this information to redraw the screen ?

(iii) What must the UIMS do to maintain the correct screen display after it has identified a mouse click ? (NOTE : Only actions needed to maintain the display are required.)

(c) Compare the use of command files in textual systems with the use of script files (such as Applescript) in graphical systems. Note similarities and differences in the structure, preparation, and use of the files, and explain why the special difficulties found in implementing the script files in graphical systems cannot be solved by the operating system alone.

CONTINUED

QUESTION 2.

(a) Explain the nature of a supervisor call (otherwise called a system call), identifying two of its properties which are essential to its function as a security mechanism.

(b) Explain how a virtual memory system can guarantee that a process is confined to its own memory space, and cannot address any memory outside that space. Point out any differences between paged and segmented systems.

(c) Explain how the combination of supervisor call mechanism and memory protection can be used to ensure secure access to devices, and show that neither is, in general, sufficient by itself.

QUESTION 3.

(a) Explain how separate arguments based on hardware and software considerations can be used to define different ways of dividing a process's requirements for memory into chunks which are significant in the operation of the memory management system.

(b) A programme runs in a system which uses virtual memory. The programme consists of three segments of equal size, each occupying exactly one page in memory : these are the main programme (M) and two procedures (A and B). The available memory is large enough to contain only two segments; any memory required by the system or other software can be ignored. The memory manager uses the least-recently-used (LRU) memory management algorithm, and a "dirty" bit, set when the segment is altered in memory.

(i) At some point in the execution, M is in memory and the other memory segment is vacant; both A and B are swapped out. Draw the process's segment table for this state.

(ii) The process executes these steps :

```
1 : M calls procedure A;
2 : A changes its internal memory;
3 : A returns to M;
4 : M calls procedure B;
5 : B returns without change;
6 : M calls procedure A.
```

Identify all operations in those steps which change the segment table, describe them (briefly !), and explain how the segment table has changed at each step.

QUESTION 4.

(a) (i) Describe a directory organisation which implements a system in which file names may have arbitrarily many components, and explain how the directory is used to find a file with a name of three components.

(ii) List the types of information which must be managed in the directory, and explain how they can be accommodated in the organisation you described in part (i).

(b) A conventional file directory tree can be implemented in (at least) two ways : by keeping each subdirectory in a separate file, or by collecting all directory information into a single volume directory on the disc.

(i) Discuss the advantage of using a volume directory.

(ii) Suggest an appropriate structure for a volume directory.

QUESTION5.

(a) Explain what a spin lock is and why it is generally not regarded as being the best way to implement locks over resources. Explain why the effect of spin locks in a single-processor system is different from their effect in a multiple-processor system.

(b) Some multiprocessor systems use adaptive locks. If the process holding the resource is currently running, the lock functions as a spin lock, otherwise the lock puts the requesting process to sleep until the resource becomes available. How is this preferable to an ordinary lock, and in what circumstances?

(c) Here is some pseudocode attempting to implement an adaptive lock. There are some race conditions in this code. Describe two of them.

```
lock:
    while ( resource is busy )
        if ( holdingProcess is not running )
            put currentProcess to sleep on a queue for
                      this resource
            break out of the while
    mark resource as busy

unlock:
    mark the resource as available
    if ( anything on the queue for this resource )
        wake up one process on the queue
```

QUESTION 6.

It is common to want to refer to the same file information with more than one name. e.g. Two users want to access the same data from their own home directories. This can be achieved in several different ways, three of which are outlined below. Explain the advantages and disadvantages of each. Consider both centralised and distributed systems and include any special requirements the methods depend on.

(a) The file can be duplicated. In this case we have two copies of the file data and two different directory entries in the home directories.

(b) The directory entries concerning the file are duplicated. We have only one copy of the file, which both directory entries refer to directly.

(c) There is a table of file information which both directories point to. The real file information such as location and ownership is stored in this table.

QUESTION 7.

(a) (i) Briefly describe a possible algorithm for the short-term scheduler on a single-processor multi-user time-sharing system. The scheduler should aim to minimise response time to user actions, and be fair to all users. State the conditions which could cause your algorithm to fail in its objectives of fairness or short response time.

(ii) Explain how your algorithm could be implemented on a multiprocessor. You may assume all memory is shared and the processors are identical. Along with the requirements for fairness and response include the need to balance the load over the processors.

(b) Imagine an interactive single-user time-sharing system. The scheduling algorithm on this system always gives priority to the current process being used in the foreground by the person. All remaining processes are scheduled on a first come first served basis, with an unlimited time slice, when the foreground process is waiting. (Of course these processes are preempted whenever the foreground task needs to run. Processes preempted in this way are returned to the end of the ready queue.)

(i) What sort of behaviour would the user notice from the use of this scheduling algorithm? What sorts of processes are penalised and rewarded by this algorithm?

(ii) Would this algorithm be suitable in a multi-user environment? Explain your answer.

QUESTION 8.

(a) Message passing systems can usually be classified as either direct communication from one process to another or indirect communication via a mailbox. Briefly outline the advantages an indirect communication method has over a direct one.

(b) A process wants to wait for messages from two mailboxes, A and B. It can do this by executing the following commands.

```
receive( A, message );
```

CONTINUED

```
                receive( B, message );
```

This forces the process to wait for data from mailbox A before it waits for mailbox B.

Describe two different approaches which allow the process to continue as soon as it receives data from either mailbox. Data from the second mailbox is to be ignored.

(c) The same problem becomes trivial with a direct communication system where messages are received by a command of the form

```
                receive( fromProcess, message )
```

which fills the fromProcess field with the sending process id when a receive is made.Why is it so simple in this case?