THE UNIVERSITY OF AUCKLAND

EXAMINATION FOR BA BSc ETC 1994

COMPUTER SCIENCE

Operating Systems

(Time allowed : THREE hours)

NOTES:

Answer SIX questions. The total mark for each question is 20; the total for the paper is 120 marks.
The marks for each question are divided equally between the parts of the question, and the same rule is applied recursively to the parts.

QUESTION 1.

(a) Three areas of concern in many computer systems are user authentication, protection, and security. For each of these areas, explain the nature of the danger which causes concern, and briefly describe two ways in which the operating system can provide safeguards.

(b) It is required that each subject using the software package Q1 must be individually invoiced for each session. Q1 performs a simple clerical operation on local files, for which it must read and write files in the local directory. The Q1 software is supplied as a machine code file which cannot be changed, and which contains no provision for accounting procedures.

Describe a way of providing secure access to Q1 while keeping a secure log file containing start and stop times, and the identification of the subject concerned, for each use of Q1. Describe your proposed method in terms of subjects and objects, and show clearly how the required access can be securely provided. State clearly any assumptions you make about the structure of the operating system, and say why they are necessary.

Is your proposed method still secure if programmes have free access to all of the computer's memory ? Explain your answer.

CONTINUED

QUESTION 2.

(a) Describe the two basic data structures used for data transfer and for administrative transactions in the interface between a running programme and the file system.

(b) Describe the actions taken by an operating system when a programme opens a disc file. Under what circumstances would you expect the **open** operation to fail ?

(c) A file contains information which is available for public use. It may be opened by any programme at any time, so it may sometimes be in use by several processes simultaneously. The file may not be changed by any ordinary process, but it is occasionally necessary to bring its information up to date.

How is it possible for a programme to change the information in the file without delaying any process which requires access to the data ? Your answer should be based on the assumptions listed below. List the significant initial properties of the file, and all actions needed to complete the change, describing the actions as they would be taken by the process which performs the change. Explain how each of the assumptions affects your answer.

Assumptions :

1 : Processes which have already opened the file can safely continue to use the old version.
2 : A process must use the name of a file to open it, but can close it without explicit reference to the file name.
3 : Processes requesting access to the file must be served immediately, and therefore have priority over the process changing the file.
4 : The new information should (subject to those constraints) be made available to open requests as soon as possible.
5 : The process executing the change has operating system privileges.

QUESTION 3.

(a) (i) Explain the functions of the system configuration and personal configuration files in shared computer systems. When and how are the files executed ?

(ii) Comment on these items of the configuration, explaining what they are, when they are set, and why they are used : disc layout; search path.

(b) List the scheduling operations on different time scales which are needed for effective management and control of a large shared computer system with a multiprogramming operating system. Comment on the similarities and differences between batch and interactive systems.

(c) What are process priorities for in scheduling ? How can they be implemented in a multiprogramming system ? Explain why it is common to increase the priorities of waiting processes with time.

CONTINUED

QUESTION 4.

(a) What is a page fault ? List the actions taken by an operating system after a page fault occurs.

(b) Two processes, P and Q, run concurrently, executing programmes PrP and PrQ. The programme code is locked in memory, and can be ignored for purposes of memory management. Four pages of memory are available for the data. Data areas are not shared. Initially, both programmes are of the same form :

```
programme Pr*;

A, B, C : onepageofdata; { Data areas, each occupying exactly one
                                  memory page. }

repeat
   use A;     { Short operation on variables only in A. }
   wait;      { Await interrupt. }
   use B;     { Short operation on variables only in B. }
   wait;      { Await interrupt. }
   use C;     { Short operation on variables only in C. }
   wait;      { Await interrupt. }
forever;

end of programme.
```

At each wait instruction, a programme waits for an interrupt; each programme has its own series of interrupts which never arrive at the same time. The period between successive interrupts is much greater than the time taken for a page fault, which is in turn much greater than the time taken to execute the use operations.

Comment on the frequency of page faults with both LRU and cyclic page replacement strategies under the three sets of conditions :

- the frequencies of P and Q interrupts are equal;
- P interrupts are twice as frequent as Q interrupts;
- P interrupts are three times as frequent as Q interrupts.

(HINT : In each case, write down the data pages in the order in which they will be required for a few cycles of the programmes, then allocate them to memory using the page replacement strategies. Note that the order of the page requests is the same for both page replacement strategies.)

CONTINUED

QUESTION 5.

(a) Explain what is meant by *consistency* and *mode* in user interface design. Show why consistency is important, and how consistency is related to the idea of modeless systems. Comment on the Macintosh features listed below, and assess their contribution to the consistency of the Macintosh interface :

- Double-clicking;
- The menu bar;
- "Trash".

(b) A characteristic of current operating system design is that the order of events within the system is much more dependent on the decisions of a person using the system than it was in earlier designs. It is now commonly possible, typically by clicking on different graphical objects displayed on a graphical user interface, to switch at will between different processes or between different activities of the same process. Describe what sort of signals must be produced by the user interface management system in response to the clicks, and how they can be used to cause the events required.

QUESTION 6.

(a) Explain the differences between heavyweight processes and threads (lightweight processes). What advantages do threads have over heavyweight processes ?

(b) Threads can be used successfully within server processes. The server process starts a thread to handle each request to the server. This solves the problem of the server blocking and stopping all client requests until the blocked request has completed. Describe how something similar can be done with the UNIX `fork` system call. In what ways is this different from the thread solution ?

(c) Migrating processes from one processor to another was discussed as one aspect of providing transparency in a distributed computing environment. What would it mean for a heavyweight process to migrate in such an environment ? What would it mean for a thread to migrate in such an environment ? Assume homogeneous processors (all of the same type).

CONTINUED

QUESTION 7.

The following client/server processes use an asynchronous (non-blocking on send) message passing system with the primitives send(toProcess, message) and receive(fromProcess, message). If the fromProcess parameter of receive is whichProcess, the operating system replaces the parameter with the identification of the process sending the message.

```
process Server;
begin
   while true do begin
      receive( whichProcess, requestMessage );
      clientProcess := whichProcess;
      doTaskForClient( requestMessage, reply );
      send( clientProcess, reply )
   end
end.

process Client;
begin
   send( Server, requestMessage );
   receive( Server, reply );
   carry on ...
end.
```

(a) (i) What would happen to the server if the client failed after executing send and before executing receive ?

(ii) What would happen to the client if the server failed after executing receive and before executing send ?

(iii) Can the code be simplified if the send primitive is synchronous (blocking) ? If your answer is yes, show how; if your answer is no, explain why not. What disadvantage would a synchronous send have in this case.

(b) Write procedures Lock and Unlock which can be called by the following program to enforce mutual exclusion to a resource using the send and receive primitives. Also write the process to which Lock and Unlock send messages. Make sure that if another process calls Unlock the resource is not released.

```
program UseMessageLock;

begin
   Lock;
   Use the resource;
   Unlock
end.
```

CONTINUED

QUESTION 8.

(a) Explain briefly what is meant by these terms :

- Device table;
- Device descriptor;
- Input-output request block;
- Device driver;
- Interrupt handler.

(b) Describe the steps required to withdraw a device from service and install a replacement (which may be different in some details) without interrupting the running of the rest of the system. What sort of system structure is needed to ensure that this operation is possible ?

(c) When a process has initiated a transaction on some device, the process may either wait for the transaction to be completed before continuing, or it may continue processing in parallel with the transaction. Briefly describe the advantages and disadvantages of each of these alternatives.

CONTINUED