THE UNIVERSITY OF AUCKLAND

EXAMINATION FOR BA BSc ETC 1993

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.

QUESTION 1.

(a) In a conventional computer system, memory is dealt with in two different ways : the system's memory manager *allocates* primary memory to processes and implements the virtual memory service, while the processes themselves *use* the allocated memory. Describe the information which must be maintained to support these two distinct views of memory, and point out similarities and differences between the two views.

(7 marks)

(b) The algorithm below transposes a matrix A. Regarding each row of the matrix (row i includes the elements { A[i, j], $1 \leq j \leq N$ }) as a segment, discuss the behaviour of the algorithm if run in a multiprogrammed segmented virtual memory system.

```
var A : array [ 1 .. N, 1 .. N ] of integer;

i, j, t : integer;

for i := 1 to N
do   for j := i to N
     do   begin
          t := A[ i, j ];
          A[ i, j ] := A[ j, i ];
          A[ j, i ] := t;
          end;
```

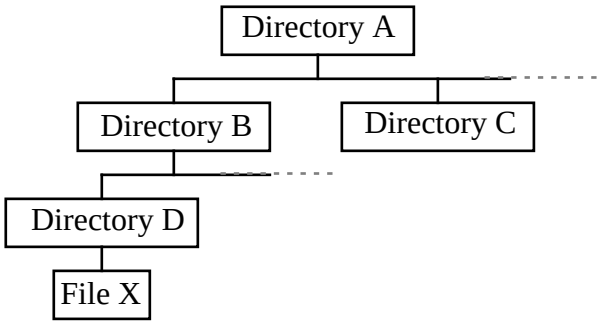
(7 marks)

(c) Discuss any changes in behaviour which might be observed if the same algorithm were run in a system with paged virtual memory.

(6 marks)

QUESTION 2.

The Unix and Macintosh file systems share a very similar structure, but are customarily used in quite different ways. Consider this structure of directories A, B, C, and D, and the file X :



The statements in the "scenario" below describe an initial state and a sequence of operations, using the Unix shell language where appropriate. Working from this scenario :

- (for each line) state the closest equivalent Macintosh description or action;
- (for lines 2 and 3) give a brief account of how the Unix system interprets the signals it receives from its user interface and obeys the instruction;
- (for lines 2 and 3) give a brief account of how the Macintosh system interprets the signals it receives from its user interface and obeys the instruction.
- (for each line) point out the significant differences between the two systems. (For lines 2 and 3, include both the differences in the system operation and the differences as seen by the people using the systems.)

The "scenario" :

1. Initially, the working directory is A.
2. The instruction `cd B/D` is issued.
3. The instruction `mv X ../../C` is issued.

NOTES : `cd` is a shell instruction to change the working directory;
`mv` is a shell instruction to move or rename a file;
`..` is a shell convention to denote the parent directory.

Low-level details (such as how to parse the input string or handle individual mouse interrupts) are **not** required. For the Unix system, assume that the user interface generates a stream of tokens, and that system procedures to find and search directories, etc. are available. For the Macintosh system, assume that the user interface generates a stream of events such as mouse clicks, double clicks, etc., and that system procedures to identify screen positions and active windows, as well as to find and search directories, etc. are available.

(Notice that there are ten equally weighted parts to this question.) (20 marks)

QUESTION 3.

(a) Explain the nature of a supervisor call (otherwise called system call), and show why it is useful in implementing protection and security systems.

(5 marks)

(b) Describe the Unix file protection system as seen by someone using Unix. Explain (briefly) *what* the operating system must do to implement this file protection, and how it gets the information necessary for its computation.

(5 marks)

(c) Show *how* the supervisor call mechanism can be used in implementing a Unix-like file protection system. Explain how the protection is ensured during the checking procedure, and say what sort of protection is needed for sources of information used.

(5 marks)

(d) In implementing Unix it is customary to store file system directories as files, but with different access rules. Assuming that the protection mechanisms as described so far are enforced, what degree of direct access to directories (read or write; user or all) can be permitted if the protection is not to be compromised ? Explain your answer.

(5 marks)

QUESTION 4.

(a) Distinguish between system backup, file generations, and file archives. Explain what each of these does (details of how they work are not required), and point out its advantages from the point of view of someone using the computer system.

(7 marks)

(b) Describe two ways of providing a file archive service, one (discretionary) requiring specific action from a file's owner, and the other (automatic) saving files without explicit instructions. Include in your description the services provided, and methods used, but not fine details of implementation. How do archive services of these types interact with the operating system's conventional file management software ?

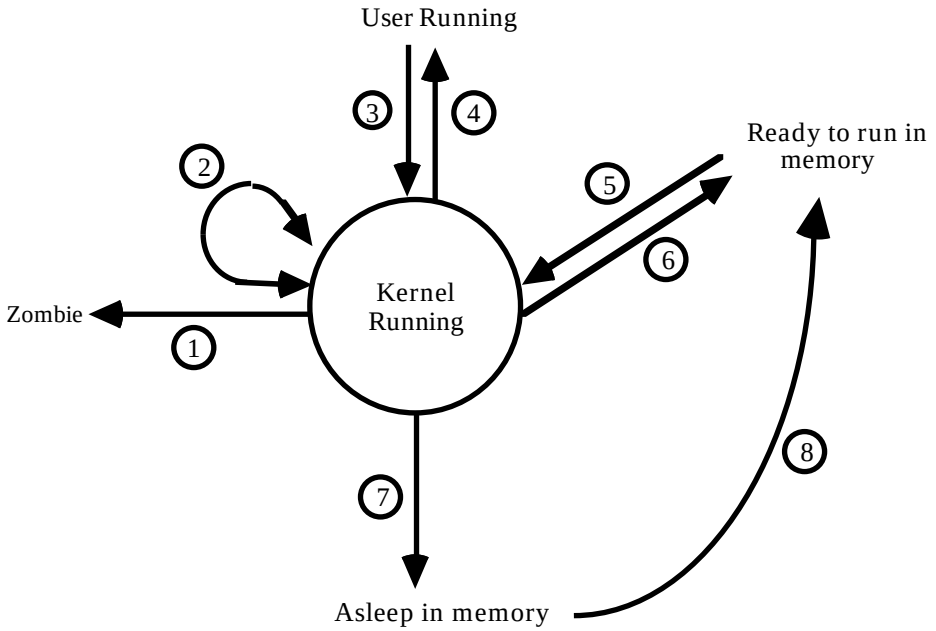
(7 marks)

(c) As costs of storage media decrease, it becomes sensible to think about preserving backups for ever. In an incremental backup system in which a complete backup copy is taken each month, increments are saved each day, and the complete backups saved for many years. Is a file archive necessary ? If no archive were provided, what sort of index to the backup system would you need ?

(6 marks)

QUESTION 5.

(a) Here is a section of the UNIX process state diagram. State briefly what change in the process's behaviour is represented by each of the numbered transitions.



(8 marks)

(b) Traditionally a process has been associated with two distinct concepts: ownership of resources (such as memory and open files) and execution (the value of the registers, especially the pc, and the state). Some operating systems treat these two concepts separately. In these systems a task is that part of a process which is concerned with resources and can contain several threads. A thread is that part of a process that is to do with execution. In this way several threads can be running within one task. They share the same memory, resources, privileges, owner etc, but each thread has its own stack space.

Give similarities and differences between a task with three threads and a UNIX process which forks itself twice giving three processes sharing the same code. Be sure to mention the overhead involved in context switching between processes compared to that between threads.

(5 marks)

(c) Threads are commonly used within server tasks to provide services to a number of different clients. Each request for service from a client process is handled by a different thread within the task. Why is this a good idea?

(3 marks)

(d) One way of providing services across a distributed (loosely-coupled) system is with remote procedure calls. Explain how remote procedure calls work.

(4 marks)

QUESTION 6.

(a) The following program fragment is the basis of a semaphore solution to the producer/consumer problem. As the producer produces integer results they are moved to a shared buffer, the consumer removes them from the shared buffer. No results are to be lost between the producer and the consumer. The shared buffer is a circular queue maintained in an array.

Show what other variables are needed in the < declarations > section, how these variables are initialised in the < initialisation > section, and complete the Producer and Consumer procedures. All results produced by a single producer must be consumed by a single consumer. You have to add semaphore operations and some code to handle the circular queue.

The semaphore operations are SemaphoreInitialize(semaphore, startingValue); Wait(semaphore); and Signal(semaphore). A semaphore is of type semaphoreType.

(8 marks)

```
program ProducerConsumer;

const
    SLOTS = 100;

var
    buffer : array [0..SLOTS-1] of integer;
    nextSlotToFill, nextSlotToEmpty : integer;
    < declarations >

procedure Producer;
var
    result : integer;
begin
    CalculateNext( result );
    ...
end;

procedure Consumer;
var
    result : integer;
begin
    ...
    UseNext( result );
    ...
end;

begin
    nextSlotToFill := 0;
    nextSlotToEmpty := 0;
    < initialisation >
    parbegin
        while TRUE do
            Producer;
        while TRUE do
            Consumer
    parend
end.
```

(b) Would your solution work correctly with multiple consumers? Explain.

(2 marks)

(c) In a monitor solution to the same problem, there would be two monitor procedures PutInfo and GetInfo. The producer procedure would call PutInfo to place the next result into the buffer and the consumer procedure would call GetInfo to remove the next result from the buffer.

Explain how condition variables would be used to coordinate activity between the producer and consumer.

(4 marks)

(d) Explain why a correct monitor solution to this problem would always work with multiple consumers.

(2 marks)

(e) Explain the advantages and disadvantages of semaphores and monitors in solving problems of this type.

(4 marks)

QUESTION 7.

(a) In many systems, all input and output must be managed through a standard system call of the form

```
DOIO( fileidentifier, action, data, resultdescriptor ).
```

Explain what the DOIO procedure does when a process requests service. Do not describe the operation of the device driver or interrupt handler software.

(10 marks)

(b) A programme is using a file XYZ, which is open for input and output. A single sector buffer is available for the file. The programme executes an instruction which means "WRITE 67 TO BYTE 3376 IN FILE XYZ"

(i) What sequence of operations must be performed to get the byte safely written to the disc ?

(ii) What sort of DOIO calls would be executed to carry out the operations which you listed in part (i) ?

(10 marks)

QUESTION 8.

(a) Describe what happens after a clock interrupt in a system using a single-queue round-robin dispatcher.

(7 marks)

(b) In a multiple-queue dispatcher system, each queue has a name, and its behaviour is described by three parameters :

- The relative frequency at which the queue is selected for dispatching, in the range 1 to 100, with 100 representing the highest frequency.
- The length of the time-slice, in the range 1 to 100, with 100 representing the longest time-slice.
- The name of a "destination queue" - the queue into which processes from this queue will be placed if they are still executing at the end of their time slices.

For example, a single-queue dispatcher might be described as

Initial : { frequency 1, timeslice 10, destination Initial }

Whenever a process is made *ready* it is placed in a queue called **Initial**, which must always exist, and there may be any number of other queues. When a running process completes its time slice, the clock interrupt handler moves it into the destination queue of its current queue. After that, or when a running process releases the processor, the dispatcher selects an occupied queue using a non-random algorithm which takes account of the queue dispatching frequencies, and moves the process at the head of that queue into the *running* state.

Show how you could define dispatcher queues to implement, separately, each of the specifications below :

(i) Any process which is still running at the end of a timeslice is given the same share of the processing time as other processes, but with only one tenth of the context-switching overhead;

(ii) Any process which is still running at the end of a timeslice is penalised by receiving only one tenth of the share of the processing time received by processes in the Initial queue, but also with only one tenth of the administrative overheads associated with context-switching overhead.

In both cases, explain why you believe that your answer satisfies the specification.

(7 marks)

(c) Why is it significant that the dispatcher described above uses a "non-random algorithm which observes the queue dispatching frequencies" ? Compare the behaviour of this system with one in which queues are handled in strict accordance with constant queue priorities, with lower priority queues only inspected if all higher priority queues are empty.

(6 marks)

CONTINUED