UNIVERSITY OF LONDON
IMPERIAL COLLEGE OF SCIENCE, TECHNOLOGY AND MEDICINE

EXAMINATIONS 2003

BEng Honours Degree in Computing Part II
MEng Honours Degrees in Computing Part II
for Internal Students of the Imperial College of Science, Technology and Medicine

This paper is also taken for the relevant examinations for the Associateship of the City and Guilds of London Institute

PAPER C211

OPERATING SYSTEMS II

Friday 16 May 2003, 10:00
Duration: 120 minutes

Answer THREE questions

Paper contains 4 questions
Calculators not required

1) a. Is the code in the OS that does the actual scheduling part of a task or not? Explain your answer.

b. Why could it be useful to introduce the distinction between user processes and system tasks (give at least three reasons).

c. Explain how the fact that messages are not buffered can create problems for the clock and terminal interrupts, and give solutions for both.

d. Describe the flow-of-control of a system call like `READ` that reads a block from disk. Make sure that you discuss the role of the interrupt handler, the system task, the user process and the scheduler; you can ignore the actual block handling.

The four parts carry, respectively, 15%, 15%, 30% and 40% of the mark.

© University of London 2003

2) a. Discuss the differences between *synchronous* and *asynchronous* send, and their implications for the kernel.

b. A car park (garage), with multiple gates that function both as entry and exit, has room for 100 cars and uses a computer to control the gates, using the following processes:

* A `Gate` process for each gate that opens and closes the local gate to let a car in to or out of the garage.
* A `Detector` process (one for each gate) which sends a message to the relevant `Gate` process when a car arrives at the gate and when it has passed through the gate (so that the gate can be closed).
* A single `Controller` process which keeps count of the number of cars in the garage and decides whether to permit entry. The `Gate` processes communicate with the `Controller`.

If the garage is full, cars will queue at a gate waiting for another car to leave.

Using the following message passing primitives

`( send (processname, message))`: asynchronous send, sender is *not* blocked;

`( receive (processname, message))`: receiver blocks waiting for a message from process `processname`, if there isn't a message available already.

`( processname = receiveany (message))`: receiver blocks waiting for a message from *any* process, if there isn't a message available already.

Give a pseudo-code outline implementation for the processes `Gate` and `Controller`.

The two parts carry, respectively, 20%, and 80% of the marks.

3) I/O and disk organisation

a. Describe four different disk head scheduling algorithms. What are the advantages and disadvantages of each scheduling algorithm?

b. Suppose that a disk has 2000 cylinders, 10 tracks per cylinder, 50 sectors per track and 512 bytes per sector. Furthermore, the rotational velocity of the disk is 60 rotations per second. The seek times for this disk are given in ms by

$$t_s = 3 + 0.025d$$

where d is the seek distance, in cylinders. The read/write heads are positioned over cylinder 20. The disk controller receives a request for disk sector 596. (Suppose that the disk sectors are numbered so that all sectors on cylinder i have lower numbers than all sectors on cylinder j, if $i < j$). What is the expected time for the controller to service this request? What is the worst-case time? What is the best-case time?

c. Disk head scheduling, interleaving, and the introduction of a cylinder skew are three techniques for improving the performance of disks.

   i) Briefly explain which, if any, of these three techniques can reduce seek times?

   ii) Briefly explain which, if any, of these three techniques can reduce rotational latency?

   iii) Briefly explain which, if any, of these three techniques can reduce transfer times?

d. Your friend is running a web server at home and is considering to switch from ordinary disk organisation to RAID disk organisation to improve the system performance. He is asking for your expert advice on which RAID disk organisation to choose. Explain in detail how different RAID disk organisations can be used to achieve the following goals:

   i) Increase the I/O request rate.

   ii) Increase the I/O transfer rate.

   iii) Fault tolerance and redundancy (*Note: There are several possible RAID configurations to achieve this. Explain the differences between them and discuss which one is the most efficient*).

The four parts carry, respectively, 20%, 30%, 20% and 30%.

4) File systems

a. Describe, step by step, the steps Minix will take, in terms of inode handing and block search, in order to resolve the path name `/homes/dr/.email` so that it can read the first byte of the file. How many disk reads are required to resolve the path name and read the first byte of the file? You can assume that every time Minix has to retrieve information from the disk, it takes only one disk read operation.

b. Consider the situation where we want to create a link `/homes/guest/.email` to the existing file `/homes/dr/.email`. In this case, `/homes/guest/.email` is the link name and `/homes/dr/.email` is the file to which it is linked.

   i) Describe the steps required to read the first byte of `/homes/dr/.email` starting with the hard link `/homes/guest/.email`. How many disk reads will it require?

   ii) Describe the steps required to read the first byte of `/homes/dr/.email` starting with the soft link `/homes/guest/.email`. How many disk reads will it require?

   iii) If the file `/homes/dr/.email` is removed, is the hard link still valid? Is the soft link still valid?

c. Suppose that you have an inode-based file system in which the disk block size is 1kB, and an inode takes 128 bytes. Disk addresses are four bytes long and the inode contains space for the first 64 bytes of data, 8 direct pointers, one indirect pointer and one double-indirect pointer. An index block is the same size as a disk block. How much space (including overhead) do files require that are:

   i) one (1) byte long

   ii) 1025 bytes long

   iii) 65536 (64KB) bytes long

d. In an operating system like Minix certain events (i.e. power failure or users pressing the reset button) can lead to inconsistencies in the file system. Explain what kind of inconsistencies may arise in these cases and how they could be corrected.

The four parts carry, respectively, 20%, 30%, 20% and 30%.

End of Paper

© University of London 2003