



University of the West of England

Computing, Engineering and Mathematical Sciences

Academic Year: 2007 - 2008
Examination Period: January
Module Leader: Ian Johnson
New Module No: UFEEHJ-30-2
Old Module No: UFSEHJ-30-2
Title of Module: Operating Systems & System Administration

Examination Date:
Examination Start time: 09:00
Duration of Examination: 2 Hour(s) 00 Minutes

Instructions to Students: Answer all of section one and one question from section two

Materials supplied to the student will be:

Number of Examination Booklets (+ any continuation booklets as required) per Examination	1
Number of Pre-printed OMR (Multiple Choice Answer Sheet)	0
Number of sheets of Graph Paper size G3 (Normal)	0

Additional Instructions to Invigilators:

Calculators may be used subject to University regulations	Yes
Students allowed to keep Examination Question Paper	No
Material supplied by student allowed :	None
Additional Specialised Material:	

Treasury tags & adhesive triangles will be supplied as standard

This page blank

- 1) Any reasonable discussion of interface between o/s and user eg interactive use, customization of unix session, programming (2). built-ins and execs (1). & = backgrounding, | piping (1), implied redirection of input and output(1).
- 2) Can only change the directory of the currently executing process (1) therefore it must be built into the shell(1). Kill is an optional built-in (1), typically built-in as if process table full(1) then cannot execute system /bin/kill(1) as no space.
- 3) 1 mark for reference to errno as global var. 1 mark for explaining that it only applies after using system calls . 1 mark for each of perror() & strerror(), mark for strerror_r being thread safe. 1 for errno.h. Any reasonable exp.
- 4) a) non preemptive, process runs till complete or blocks on I/O (2)
b) preemptive, process swapped out on time slice, or interrupt or block on syscall or I/O (2)
c) cooperative, as non but checks at intervals to see if higher priority proc. wants to run, if yes then release cpu. (1)
- 5) expect discussion of the number of context switches (2). 2 for unix, up to 6 for XP (2), need for message block with XP (1)

ANSWERS

ANSWERS

ANSWERS

6)

marks from:

- i) pipes: unnamed pipes, between related processes only, kernel provides mutex, one way only, blocking
mkfifo: named pipes, rel or unrelated processes, subject to access permissions, one way, kernel mutex, blocking
messages: one way, can have type, block or non-block, kernel mutex, access perms, rel/unrel but need key id
shared mem: shm no mutex, related threads/processes, dangerous. mmap unrelated processes, no mutex
- ii) int of specified size, atomic inc & dec, handled by kernel, sem_wait on access, access will decrement flag, if not available then place process on q. sem_signal on exit from access, if q then wake next else inc semaphore. etc. posix provides sem_init, sem_wait, sem_trywait, sem_post sem_getvalue, sem_destroy

7)

- ii) marks for explaining FIFO (first-in first-out) and NRU (Not recently used) (2).
- FIFO: simplest, allocate pages sequentially, always replace oldest. (1), discussion of risk, eg loops and replacing page about to be used. (2)
 - Modified FIFO: keep temporary list of removed pages and write block of pages out at intervals. AKA a cache of pages so check this list before swapping in a page. (2)
 - NRU: Circular search of page list, clear reference bit first time around, if still clear on second pass then page out else leave. (2)
- see attached diagrams.

iii) Indication as to how the 32 bit address is split up into 10 bits for index into page directory + 10 bits for index into page table + 12 bits to identify byte within page. (4). There should also be discussion of the entries within the page table along the lines of ... high order 20 bits give page frame address (1) and low order bits are used for information about the page (1), present bit, read/write bit, privilege bit, accessed bit, dirty bit etc (4)

8)

i) see attached diagram

ii) Correct identification of parent & child, 1 mark. Explanation of pid value 2 marks.

up to (7) for explanation of execvp call, v = array of pointers(1), so need to build the argument list before calling(1); p = will do path search(1) as per shell's \$PATH (1), first arg is pointer to filename(1), second arg is pointer to array of args(1), shouldn't ever (in principle) return to execute the perror. If it does then it will call exit(1).

marks for blocking wait on specific child(pid)(1), get the exit status(1), no options, WIFEXITED macro to test if child exited normally, via exit() or _exit() or return from main()(2). WEXITSTATUS gets the return value that may have been set by child(1). Can only be evaluated if WIFEXITED true(1).