PRIFYSGOL CYMRU; UNIVERSITY OF WALES

DEGREE EXAMINATIONS JANUARY 2002

SWANSEA

Computer Science

CS 228 Operating Systems

Attempt 2 questions out of 3

Time allowed: 2 hours

Students are permitted to use the dictionaries provided by the University through the invigilators

CS_228 Operating Systems (January 2002)

*(Attempt **2** questions out of 3)*

Question 1.

(a) With the aid of a state transition diagram, illustrate the transitions between process states **running**, **ready**, **blocked**, **suspended-ready** and **suspended-blocked**. The events causing the transitions should be clearly indicated.

An interactive process is currently in the running state. Consider each of the following events independently. Could a state transition possibly be caused by the event, and if so, which state would the transition lead to? Briefly explain your answers.

(i) the user sends an interrupt signal (similar to **`ctrl-c`** in UNIX) to the process from the keyboard;
(ii) the process makes a system call to sleep for 10 seconds (e.g. a C-library call **`sleep(10)`**);
(iii) the process calls a subroutine to read a number from keyboard input (e.g. a C-library call **`scanf("%f",&a)`**);
(iv) the process calls a trigonometric subroutine (e.g. a C-library call **`sin(a)`**);
(v) the process makes a system call for a semaphore P operation.

[**10 marks**]

(b) Consider a simplified Sleeping Barber Problem. A barbershop consists of a waiting room with 4 chairs, and a barber room with a barber chair. If a customer enters the shop and all chairs are occupied, the customer waits by the door. If the barber chair is occupied but a waiting chair is available, the customer sits in a free chair. If the barber chair becomes free, one of the waiting customer takes over the chair and has a haircut. The pseudo-code for describing this problem is given below:

```
1   program Barbershop;
2   shared var free_waiting_chairs: Integer;
3                  free_barber_chair: Boolean;
4   concurrent_procedure Customer;
5   begin
6       while free_waiting_chairs <= 0 do;                    {busy waiting}
7       free_waiting_chairs := free_waiting_chairs-1;
8       while free_barber_chair = False do;                   {busy waiting}
9       free_barber_chair := False;
10      free_waiting_chairs := free_waiting_chairs+1;
11      {has a haircut for a random period}
12      free_barber_chair := True;
13  end;
14  begin {main program}
15      free_waiting_chairs := 4;
16      free_barber_chair := True;
17      concurrent_begin
18          repeat
19              Customer;
20              {wait for a random period}
21          forever
22      concurrent_end
23  end.
```

(i) Indicate all the critical sections in the above code by giving line numbers.
(ii) Provide the program with mutual exclusion by inserting the necessary code. You may use semaphore operations.
(iii) Assume that customer processes cannot share any global variables (such as `free_waiting_chairs`) except semaphores. Using pseudo-code, outline a solution to the problem with appropriate semaphore operations.

[8 marks]

(c) A file containing 2050K bytes of data is to be stored in a UNIX file system with 1K-byte data blocks. Each indirect block can hold 256 disk addresses (i.e. 32-bits per address). What is the total number of data blocks required to store this file (including all indirect blocks but not the i-node itself)? Explain your answer.

Describe, with the aid of a diagram if necessary, the concepts of the Networked File System (NFS, designed by SUN Microsystems).

[7 marks]

Question 2.

(a) In the context of memory management, contrast the following:
(i) internal fragmentation and external fragmentation;
(ii) demand paging and anticipatory paging;
(iii) page faults and segmentation faults.

[6 marks]

(b) With demand paging memory management and the **least recently used (LRU) page replacement strategy**, the current LRU matrix (6□6) is shown on the right, assuming that the memory has only six page frames. If the next three page reference numbers are **3**, **7** and **8** respectively, illustrate the changes to the LRU matrix step by step.

Does the LRU page replacement strategy suffer from Belady's anomaly? Explain Belady's anomaly and your answer.

	5	**3**	**2**	**6**	**9**	**4**
5	0	1	1	1	0	1
3	0	0	0	1	0	0
2	0	1	0	1	0	1
6	0	0	0	0	0	0
9	1	1	1	1	0	1
4	0	1	0	1	0	0

[6 marks]

(c) In the context of demand paging, consider the code as shown on the right for initialising an array with 10 rows and 32 columns. Suppose that

```
1     for col:=1 to 32 do
2        for row:=1 to 10 do
3           A[row,col] := 0;
```

- the rows are stored in contiguous blocks of virtual memory and the elements in each row are stored in contiguous words;
- the memory consists of page frames, each holding a row of 32 elements;
- a maximum of 9 page frames may be allocated to the array;
- initially no part of the array is in memory.

How many page faults would the running of the above code cause if the first-in first-out (FIFO) page replacement strategy is used?

How could you reduce the number of page faults by modifying the program? How many page faults would the new code cause if the FIFO strategy is used?

Describe a page replacement strategy that would reduce the number of page faults in this particular case (without modifying the original program).

[6 marks]

(d) Describe in detail how a virus infects programs, and how an anti-virus detects known viruses.

Briefly describe what is meant by the terms *encrypted virus* and *polymorphic virus*.

[7 marks]

Question 3.

(a) In the context of process management, what is meant by the following?
(i) context switch; (ii) thread; (iii) session.

Explain why a "process" is not a "program".
Explain how child processes are created in UNIX.

[12 marks]

(b) Consider the set of processes shown on the right. We assume that the CPU ready queue is empty at time 0, and the time needed for context switches is negligible.

Process Name	P1	P2	P3
Arrival Time (msec)	0	1	5
Burst-Time (msec)	7	4	2

For each of the following scheduling algorithms, determine the average waiting time (over all three processes):
(i) non-preemptive FIFO (first in first out) scheduling;
(ii) non-preemptive SJF (shortest job first) scheduling;
(iii) preemptive RR (round-robin) scheduling (with quantum = 2 msec).

What is the effect of increasing the time quantum to an arbitrarily large number for the round-robin scheduling?

[7 marks]

(c) Consider a **variation of round-robin scheduling** in which a process that has used x% of its quantum is returned to the ready queue at a place x% of the distance away from the beginning of the queue. For example, a process that has used a full quantum is returned to the end of the ready queue, and one that has used half a quantum to the middle of the queue. Discuss the merits and demerits of this algorithm in comparison with both **simple round-robin** and **multi-level feedback queues** algorithms. *(Hint: you may consider issues such as I/O bound processes, CPU-bound processes, fairness, waiting time, queue management, newly-arrived processes and quantum length.)*

[6 marks]