VICTORIA UNIVERSITY OF WELLINGTON

Te Whare Wananga o te Upoko o te Ika a Maui

EXAMINATIONS — 2002

MID-YEAR

COMP 305

Operating Systems

Time Allowed: 3 Hours

Instructions: Answer question one.

Answer **five** of questions two through seven.

Each question is worth 25 marks and should take you about 25 minutes.

Paper foreign language dictionaries are allowed.

Electronic dictionaries and calculators are not allowed.

Question 1. Warm Up [25 marks]

A few short questions to get the mind working. Remember to allow approximately one minute for each mark.

(a) [3 marks] State three major goals of OS design.

(b) [2 marks] What is the major difference between Multiprogrammed and Timesharing systems?

(c) [3 marks] Briefly state what happens when a kernel context switches between:

i. processes,

ii. kernel threads within the same process, and

iii. user threads within the same process.

(d) [4 marks] Atomic Transactions

(i) In what ways are atomic transactions and critical sections similar? In what ways do they differ?

(ii) Define a **conflicting operation** in the context of atomic transactions.

(e) [3 marks] A system is designed in such a way that each process has five segments (code, library, stack, heap, global data). If the system is implemented on an architecture with pages consisting of 4096 bytes what will be the format of the 32-bit virtual address?

(f) [4 marks] Suppose that a Unix user has opened a file and the operating system has placed the i-node into the i-node table. Assume that the i-node is the only information in memory to begin with, that the file blocks are numbered from 0, and that each indirect block contains 512 pointers.

(i) If the user makes a request for block 21 of the file, how many disk reads will the operating system have to make in order to load the block into main memory?

(ii) If the user next makes a request for block 530 of the file, how many disk reads will the operating system have to make in order to load the block into main memory?

(g) [2 marks] It has been asserted that a page table provides protection in the capability style, not the access-list style. Do you agree or disagree? Briefly support your answer.

(h) [4 marks] Most modern systems use DMA (direct memory access) for faster I/O devices.

(i) How does DMA increase system concurrency?

(ii) How does DMA complicate the hardware design?

Question 2. Synchronisation [25 marks]

(a) [3 marks] List the three requirements that a solution to the critical-section problem must satisfy.

(b) [2 marks] Which of these requirements does the Test-and-Set hardware synchronisation primitive **fail** to satisfy, why?

(c) [20 marks] **Semaphore Cafeteria Simulation.**

Your task is to simulate a cafeteria using semaphores. The cafeteria is a simple design, with a single queue and serving area. There are three parts to the meal which are: rice, naan and dahl. Each part is served by a different cafeteria worker. Each customer enters the serving area, and starting with the rice, proceeds in order to the naan and last the dahl, until their meal is complete. Because of the limited number of cafeteria workers, only three customers are permitted in the serving area at one time. Any remaining customers wait patiently in a queue, and only once a customer leaves the serving area, may another enter from that queue.

The cafeteria workers are very talkative, talking (but not to anyone in particular) immediately before and after serving a customer. When they are talking they will not serve any food. To obtain each part of the meal, the customer must first get the workers attention, and then wait while the dish is served.

Your answer will be evaluated on correctness, efficiency and clarity and should be written in C++ (or a close pseudo code approximation). You may use the following semaphore class, no other synchronisation primitives are permitted. Also included is a useful enumerated type which you can use.

```
class Semaphore{
  Semaphore(int counting);
  void P();    // wait
  void V();    // signal
};

enum course {rice, naan, dahl, num_courses};
```

Question 3. Deadlock [25 marks]

(a) [5 marks] List the necessary conditions for deadlock to occur. For each of these conditions, state why or why not elimination of the condition is a sensible approach to deadlock prevention.

(b) Consider a system with 3 processes (P_1, P_2, P_3) and 3 resource types (R_1, R_2, R_3). The total number of available resources of each type in the system is $\{R_1 = 1, R_2 = 2$ and $R_3 = 2\}$. The current allocation situation is shown below in table **??**.

	Allocation			**Requesting**		
Process	R_1	R_2	R_3	R_1	R_2	R_3
P_1	1	0	0	0	0	1
P_2	0	0	1	0	0	0
P_3	0	1	1	1	0	0

Table 1: Resource allocation table.

(i) [3 marks] Draw the resource allocation graph corresponding to the state described in table **??**. Is this state deadlocked? Justify you answer.

(ii) [2 marks] Suppose P_2 requests one unit of R_3. Draw the **new** resource allocation graph. Is this new state deadlocked? Justify your answer.

(c) Consider a system with 5 processes and 4 resource types that is currently in the safe state shown in table **??**.

	Allocation				**Maximum**			
Process	R_1	R_2	R_3	R_4	R_1	R_2	R_3	R_4
P_1	0	0	1	1	0	0	1	2
P_2	1	3	3	4	2	3	4	6
P_3	1	0	0	0	1	7	5	0
P_4	0	0	1	4	0	6	5	6
P_5	0	6	3	2	0	6	5	2

Available			
R_1	R_2	R_3	R_4
1	5	3	1

Table 2: Resource allocation table.

For each of the following sequences of requests, indicate if the sequence can be immediately granted, and leave the system in a safe state. Indicate why the new state is safe or not safe after **each** step in the sequence. **Note**: Assume the three sequences are independent, each starting from the state shown above in table **??**.

(i) [5 marks] P_1 requests (0,0,0,1) then P_3 requests (0,1,3,0)

(ii) [5 marks] P_4 requests (0,3,1,1) then P_2 requests (1,0,1,0)

(iii) [5 marks] P_5 requests (0,0,1,0) then P_2 requests (1,0,1,1)

Question 4. CPU Scheduling [25 marks]

(a) [3 marks] State what is meant by short-term, medium-term and long-term schedulers.

(b) [2 marks] State which short-term scheduling algorithm produces optimal allocations. Why is it impossible to implement?

(c) Consider a system with one CPU and four jobs. Each job has an arrival time, burst time and priority as given in table **??**. Priorities are ranked on a scale from 0 (lowest) to 127 (highest).

Job	Arrival Time	Burst Time	Priority
1	0	6	60
2	3	8	70
3	10	7	80
4	9	3	122

Table 3: Job arrival times, burst times and priorities.

(i) [4 marks] Draw a Gantt chart and then find the average **waiting time** for the non-preemptive SJF scheduling algorithm.

(ii) [4 marks] Draw a Gantt chart and then find the average **turnaround time** for the preemptive priority scheduling algorithm.

(iii) [9 marks] Find the average waiting times for the **RR scheduling algorithm**, with quanta 4,6 and 8.

(iv) [3 marks] What happens to the RR scheduling algorithm as the length of the quantum approaches infinity? For the jobs described in table **??** what is unusual about the results of the data set from (iii)?

Question 5. Memory Management [25 marks]

(a) [6 marks] Consider the various policies for allocating contiguous memory to processes. One way in which we can optimise the performance of each policy is to design the memory manager's free list to give maximum support to the policy.

(i) How should the free list be organised to best support the best-fit policy?

(ii) How should the free list be organised to best support the worst-fit policy?

(iii) How should the free list be organised to best the first-fit policy?

(b) [4 marks] Consider a variation on the first-fit policy called *next-fit*. This policy maintains a list like first-fit but searches the list in a circular fashion, begining a search to satisfy a new request from the point in the list at which it satisfied the last request.

How do you think the behaviour of *first-fit* and *next-fit* will differ?

(c) [15 marks] Let $\omega = 2\ 3\ 4\ 3\ 2\ 4\ 3\ 2\ 4\ 5\ 6\ 7\ 5\ 6\ 7\ 4\ 5\ 6\ 7\ 2\ 1$ be a sequence of page references in a logical address space.

(i) A process has a page frame allocation of 3 and the physical memory is initially unloaded. The system uses **Belady's optimal** page replacement algorithm.

For each reference in ω indicate whether or not a page fault occurs and which logical pages are present in physical memory following the reference. How many page faults are incurred by the entire reference string?

(ii) A process has a page frame allocation of 3 and the physical memory is initially unloaded. The system uses the **LRU** page replacement algorithm.

For each reference in ω indicate whether or not a page fault occurs and which logical pages are present in physical memory following the reference. How many page faults are incurred by the entire reference string?

(iii) A process has a page frame allocation of 3 and the physical memory is initially unloaded. The system uses the **FIFO** page replacement algorithm.

For each reference in ω indicate whether or not a page fault occurs and which logical pages are present in physical memory following the reference. How many page faults are incurred by the entire reference string?

(iv) The system uses the **Working Set** page replacement algorithm with a window size of 6. Assume a process starts with no pages in physical memory.

For each reference in ω indicate whether or not a page fault occurs and which logical pages are present in physical memory following the reference. How many page faults are incurred by the entire reference string?

What is the **working-set size** after the last reference in ω has been processed?

Question 6. File Systems [25 marks]

(a) [3 marks] Every system maintains meta data about a file in a file descriptor similar to the Unix inode. List three types of information that are commonly found in such a file descriptor.

(b) Consider a system which keeps both a per process open file table and a system open file table.

(i) [3 marks] What information is typically stored in a process's open file table?

(ii) [3 marks] What information is typically stored in the system open file table?

(iii) [6 marks] Outline what happens when a process opens a file.

(c) [5 marks] Two Comp 305 students are discussing file systems. Yang points out that many systems attempt to store the meta-data for all files listed in a directory in the same cylinder as the directory to provide efficient access. She suggests this could be improved further by storing the meta-data within the directory. Mary disagrees and argues the meta-data should be stored separately. Who do you agree with? Justify your answer.

(d) [5 marks] Modern file systems such as Unix are implemented in a way that avoids any requirement that a file's data blocks be stored contiguously. However, in Unix at least the direct data blocks are stored on the same cylinder as the meta-data. Why is this?

Question 7. Security [25 marks]

(a) We have stated that it is important to seperate policy and mechanism when designing a security system.

(i) [2 marks] What is meant by **policy**? Give an example of a policy.

(ii) [2 marks] What is meant by **mechanism**? Give an example of a mechanism.

(iii) [2 marks] Discuss the benefits of the seperation of policy from mechanism.

(iv) [4 marks] Define **authentication mechanisms** and **authorisation mechanisms** in a way that clearly indicates the roles played by both.

(b) There are situations in which the simplifying assumptions made by operating systems can create problems for the designer of a complex application.

(i) [5 marks] Outline a realistic scenario in which Unix access lists are an inadequate mechanism for protection.

(ii) [5 marks] Explain the additional mechanism(s) available in Unix to address this problem and how they could be used to solve the scenario you presented above.

(c) It is usually considered that dynamic address translation hardware provides the basic memory protection mechanism.

(i) [3 marks] What is the *protection state*, i.e. that part of the state that enforces the memory protection, in relocation hardware?

(ii) [2 marks] How does the operating system ensure that the protection state is not changed indiscrimanately?

****