TIKA

CONCORDIA UNIVERSITY

DEPARTMENT OF COMPUTER SCIENCE

Comp 343 — Winter 1990

SOLUTIONS TO MID-TERM EXAMINATION

Sec. V — R. Jayakumar

---

1. (a) Advantages: Modularity of the layered approach makes debugging and verification of the system much easier. Also, each layer hides the existence of certain data structures, operations, and hardware from higher-level layers.
   To whom: Mainly to the operating system designer. Also, the user benefits from information hiding.
   Disadvantages: High design complexity and possibly slower system.

   (b) In multiprogramming, several programs are in memory concurrently; the system switches among the programs for efficient processing, and minimal idle time.
   Advantages: Minimal idle time, improved CPU utilization, concurrent I/O and CPU operations, etc.

   (c) Even in the case of a single user, multiprogramming can be used to improve CPU utilization and achieve concurrent I/O and CPU operations.

2. (a) Because the interrupt facility makes possible concurrent processing and, as a result, improved resource utilization.

   (b) Busy-waiting: A process is waiting for an event to occur and it does so by executing instructions.
   Busy-waiting can be avoided by using semaphores in which case a process can block itself on a semaphore instead of busy waiting.

   (c) Semaphore waiting lists could be implemented as stacks; however, this could cause starvation.

3. (a) Readers have priority since a writer can proceed to write only when there are no more readers waiting to read (enforced by **if** readcount = 0 **then** signal(wrt)).

   (b) When writing is in progress, wrt semaphore is not available. Thus, the first reader will hold mutex semaphore and wait on wrt semaphore; and other readers will wait on mutex semaphore. When only reading is in progress, the first reader will hold wrt semaphore (thereby preventing a writer) and immediately release mutex semaphore; and other

readers can proceed to read because mutex semaphore is available.

(c) The first reader will hold wrt semaphore; all other readers will increase readcount and will be blocked on wrt semaphore. When the first reader completes, since readcount will not be zero (assuming there are more than one readers) wrt semaphore will not be released. Hence, all other readers and writers will be blocked indefinitely on the wrt semaphore.

4. (a) A swapping/relocation memory management scheme improves memory utilization by keeping only ready to run process in memory; which in turn improves CPU utilization due to multiprogramming and improves I/O utilization due to concurrent I/O and CPU operations.

(b) Address translation and error detection occurs at run time. A limit register, a base register, a comparator and an adder are required (refer to Fig. 6.6, page 229).

(c) Dynamic relocation is used because it requires less loading time. Implementation of swapping is simpler using dynamic relocation as a job can be loaded any where in the memory.

Concordia University
Department of Computer Science

COMP 343/543

November 5, 1990 Mid-Term Exam

1. You are doubtless familiar with the simple solution to the mutual exclusion problem using a semaphore (say, mutex) initialized to 1. The question is: (a) state the two main requirements of any solution to the mutual exclusion problem, and (b) would your simple solution continue to meet both requirements if the semaphore queue organization changed from FIFO to LIFO? Explain.

2. In a paged virtual memory, there is a sharp distinction between (a) memory references in which only hardware intervention is required, and (b) memory references in which both hardware and software intervention are required. Explain this distinction and its performance consequences.

3. Explain the concept of layered design (incremental machine design) of an operating system, in one or more sentences. Then, provide two examples of operating system services that use other operating system services. (Think of our 15-level model of an operating system).

4. What is the main performance difference between solutions to the mutual exclusion problem employing test-and-set, and those employing semaphores?

5. How could having a device driver contribute to device independence? Why is device independence a good thing? Does it make you less ~~tired~~ tied to the peripherals of particular manufacturer? Explain.

Q2

Concordia University
Computer Science 343/2-A - Operating Systems

Mid-Term
Date October 23, 1990

Max. Time 1 hr. 10 mins.
Dr. B.C. Desai

Answer all questions in the space provided under each question.
This test is worth 20 points.

1. (6 points)
(a) What are the objectives of an O.S.

1) To maximize use of expensive resources such as cpu, memory.

2) To make the bare machine easier to program by providing device drivers, file services, etc.

3) In maximizing resource use, we must implement multiprogramming/multiprocessing/timesharing. This brings up a new problem which the o/s must address: security. Processes must be prevented from accessing memory and files which properly belong to other processes and/or the o/s itself.

(b) What is multiprogramming?

Multiprogramming is running more than one program on the cpu at one time. This is distinguished from multiprocessing which is running more than one process on the cpu at once. But one program may consist of more than one process. Also, a process may be an interactive user, and this is not generally termed a "program".

Oct 5, 7U

2. (4 points) List the constraints that a solution to the critical section problem must satisfy.

1) mutual exclusion: no two processes may execute their critical sections at the same time.

2) bounded waiting: no process must be permanently denied use of the cpu because it is waiting to perform its critical section. (For example, 3 processes want to execute their C.S. The cpu alternates between executing processes 1 & 2, & this makes process 3 wait "forever".)

(Who decides who's next process to enter the C.S.?)

3) progress: Each process must give up their claim to the critical section after they have been through it, in order to allow the next process in, ie a process may not get stuck in its critical section, thus locking others out of theirs.

3. (4 points) Give the implementation of the semaphore wait operation using the Test&Set instruction.

wait (...)

begin

repeat

Test&Set (lcm)

until lcm = 1;

end;

- block ele?

2

correct answer:

wait (sem)

begin

Test&Set (hypsem)

if sem1 > 0

then sem1 := sem1 - 1;

else block the process;

hypsem := 1;

end;

Oct 23 9?

4. (6 points) In order to make (and eat) an eggroll, one needs three ingredients: eggroll shell, filling, and sauce. Consider a system with three eggroll eater processes and one agent process. Each eggroll eater continuously makes an eggroll and eats it. One of the processes has eggroll shells, another the filling, and the third has the sauce. The agent has an infinite supply of all three and a wok to cook it in. The agent places two of the ingredients, chosen randomly, on the table. The eggroll eater who has the required third ingredient comes to the table, makes, cooks and eats the eggroll at the table and then leaves to await for more. The agent seeing the table free puts another two of the three ingredients and the cycle repeats. Write a program to synchronize the agent and the eggroll eaters.

var shell, filling, sauce: Boolean semaphore; int 0;
counter : Boolean semaphore; init 1;

eaterShell
begin
repeat
wait (shell);
make eggroll & eat it;
signal (counter);
until false;
end;

agent
begin
repeat
wait (counter);
pick two ingreds at random;
signal (third ingred);
until false;
end;

eaterFilling
begin
repeat
wait (filling);
make eggroll & eat it;
signal (counter);
until false;
end;

6

eaterSauce
begin
repeat
wait (sauce);
make eggroll & eat it;
signal (counter);
until false;
end;

CONCORDIA UNIVERSITY

DEPARTMENT OF COMPUTER SCIENCE

Comp 343 — Winter 1990

MID-TERM EXAMINATION

Sec. V — Time: 75 Minutes

Instructor: R. Jayakumar

ANSWER ALL THE QUESTIONS

CONCISE ANSWERS WILL BE APPRECIATED

1. (a) [10%] If an operating system is constructed as a series of layers, what advantages does this provide? To whom? What disadvantages might there be over other designs?

   (b) [8%] What is multiprogramming? What are the advantages of multiprogramming?

   (c) [7%] An operating system for a personal computer will have only one user (at a time). Explain why it is still useful to have multiprogramming on such a machine.

2. (a) [8%] The critical section problem arises within a set of concurrent programs because these programs are allowed to interrupt each other. Thus, a very simple way of avoiding the critical section problem is not to provide interrupt facility in the computer system. Discuss why this solution is not followed in general.

   (b) [9%] What is busy-waiting? Can busy-waiting be avoided altogether? If so, explain a way of avoiding it. If not, explain why it cannot be avoided.

   (c) [8%] Semaphore waiting lists are often implemented as queues served in FIFO order. Could they be implemented as (LIFO) stacks? What problems might this cause?

3. The general structure of a reader and a writer process in a solution to the readers/writers problem is shown in the next page. (The semaphores mutex and wrt are initialized to 1, and the variable readcount is initialized to 0.)

   (a) [5%] Which class of processes (readers or writers) has priority? How?

   (b) [10%] Explain how this solution delays a new reader when writing is in progress, but does not delay a new reader

when only reading is in progress.

(c) [10%] What is the effect of changing the statement

```
if readcount = 1 then wait(wrt)
```

in the reader process to

```
if readcount >= 1 then wait(wrt)?
```

Explain your answer.

Reader process:

```
wait(mutex);
   readcount := readcount + 1;
   if readcount = 1 then wait(wrt);
signal(mutex);
      ...
    reading is performed
      ...
wait(mutex);
   readcount := readcount - 1;
   if readcount = 0 then signal(wrt);
signal(mutex);
```

Writer process:

```
wait(wrt);
      ...
   writing is performed
      ...
signal(wrt);
```

4. (a) [7%] Explain how a swapping/relocation memory management scheme optimizes resource usage in a computer system.

(b) [8%] In swapping/relocation memory management, when do address translation and error detection occur? What are the hardware requirements?

(c) [10%] Which type of relocation (namely, static or dynamic) would you use in swapping/relocation memory management? Why? How does the relocation scheme you selected make the implementation of swapping more convenient?

*** END OF EXAMINATIONS ***