COMPSCI340SC & SOFTENG370SC 2002

Operating Systems Test

Question 1.

Early computers did not use multiprogramming. Give three hardware changes that needed to be made before multiprogramming could be used safely and efficiently. Describe why each change was necessary.

More memory – to hold more than one program at a time
Interrupts – to allow processing to continue while IO operations were carried out.
Memory protection – to keep the memory of each program safe from the other programs.
Privileged modes and instructions.
Also accept disk drives – to enable faster access to programs and data.

6 marks

Question 2.

Which of the following instructions should be privileged? Briefly describe what could go wrong if the instruction was not privileged.

- Disable interrupts
- Read the real-time clock
- Switch from user mode to kernel mode
- Switch from kernel mode to user mode

Disable interrupts – otherwise a process could grab and hold the processor
Switch from user mode to kernel mode – otherwise a process can enter kernel mode at any time.

4 marks

Question 3.

Describe the main differences between microkernel designs for operating systems and the so called monolithic designs. Give reasons why most modern general-purpose operating systems have moved to microkernels.

The kernel is smaller.
Most services are run as user level programs.
Requests for services are sent as messages via the microkernel.
More flexible – easier to add and modify services.
Can be regarded as safer as one part of the OS can communicate another only via messages rather than by directly accessing memory.
Because services are supplied via message passing it is easier to extend to distributed environments.

10 marks

Question 4.

Some operating systems (e.g. Solaris) employ a multi-leveled approach to thread implementation. Give four advantages such a system has over user-level or system-level thread implementations.

Supposedly has the advantages of each type.
Doesn't block when a thread makes a blocking system call – a new system-level thread can be created and associated with a process's thread.
Creation of threads (at the user-level) is simple and fast. The system-level threads are only created when required.
Switching between threads is faster (no entering the kernel for user-level switches).
Can use multiple CPUs

8 marks

Question 5.

Briefly describe why the Java thread method `stop()` is deprecated.

When a thread is stopped it is forced to give up any locks it has. It may leave data behind in an unsafe state.

5 marks

Question 6.

List 10 things that should either be found in a Process Control Block or be easily accessible from it in a multiprocessor operating system.

Any of the following (and some others):
Memory
Open files
Devices being accessed
User identification
Process identification
Process state
Threads
Resource limits
Links to other processes
Processor registers
Which processor
Access rights
Process group
Time running or creation time
Accounting information
Exception handlers

10 marks

Question 7.

What major problem can be caused by using fixed priorities to schedule processes? Describe a solution to this problem.

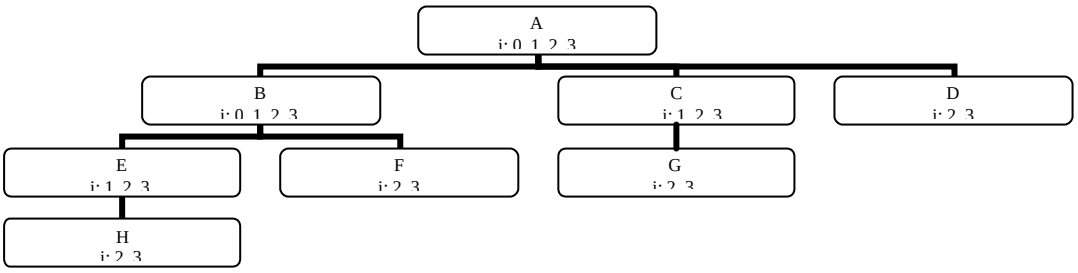
Starvation or indefinite postponement of lower priority processes.
Increase priorities of waiting processes over time. (Known as process aging.)

3 marks

Question 8.

Draw a tree showing the parent and child processes started by the following UNIX code. How many processes are started up (including the first process executing the code)?

```
main() {
  int i;
  i = 0;
  while (i < 3) {
    fork();
    i++;
  }
}
```



8 processes

7 marks

Question 9.

What is copy on write? Describe one operating system situation where copy on write solves a problem.

Copy on write is when a section of memory is copied when a process writes to it. It is used whenever two or more processes share the same memory but changes by one are not to be seen by the other. In this case as soon as one of the processes writes to the section the section is duplicated for the writing process. Examples are the data area of processes after a fork system call or some fast message passing systems.

5 marks

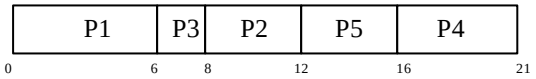
Question 10.

Here are the arrival and burst times for a number of processes:

Process	Arrival Time	Burst Time
P1	0	6
P2	2	4
P3	3	2
P4	5	5
P5	8	4

From this table draw a Gantt chart showing a Shortest Job First schedule *without* preemption and calculate the average waiting time.

Not to scale



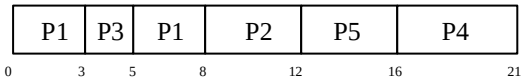
Average waiting time
(0 + 6 + 3 + 11 + 4) / 5 = 4.8

P2 and P5 can be swapped, it gives the same average waiting time.

5 marks

From the table draw a Gantt chart showing a Shortest Job First schedule *with* preemption and calculate the average waiting time. If there are two or more processes with the same shortest remaining burst time and one is currently running, then do not preempt it.

Not to scale



Average waiting time
(2 + 6 + 0 + 11 + 4) / 5 = 4.6

P2 and P5 can be swapped, it gives the same average waiting time.

5 marks

Question 11.

When we have resources that must be used by one process at a time we can get a phenomenon known as priority inversion. What is priority inversion?

When a process with a low priority holds some resource wanted by a process with a higher priority. In this case the high priority process has to wait for the low priority one.

4 marks

Question 12.

What is the X protocol? What is it used for and how does it work in a distributed environment?

It is a protocol which can be used to produce a graphical user interface. It is used to control user interface devices such as keyboards, mice and displays. It has a client-server model. The client is an ordinary program linked with special libraries. The server controls the devices. Requests get sent from the client to draw things on the display and receive data from the user. It works easily in a distributed environment because it is based around message passing requests and responses.

5 marks

Question 13.

Here is a semaphore class written in Java. Complete the corresponding `semSignal` method. Don't worry about the possibility of indefinite postponement.

```
public class Semaphore {
  private int s;

  public Semaphore(int count) {
    s = count;
  }

  public synchronized void semWait() throws InterruptedException {
    while (s < 1)
      try {
        wait();
      }
    s--;
  }

 public synchronized void semSignal() {
  s++;
  if (s == 1)  // the test is unnecessary
   notifyAll(); // or notify
 }
```

5 marks

Why did I state "Don't worry about the possibility of indefinite postponement"?

There is no way of controlling which thread will run first after the notifyAll and hence get the resource. Thus it is possible for a thread to be postponed indefinitely.

3 marks

Question 14.

The UNIX `pipe` system call is used to implement the shell pipelines. E.g. `ls –l | wc` Briefly describe how this could be done. You should mention creating the processes (`ls` and `wc` in this case) and how they must be connected with a pipe. Remember that pipes have to be constructed before the child processes that use them.

The shell makes a new pipe. It then forks twice. One child sets up its output to go to the pipe then it execs ls, the other sets up its input to come from the pipe then it execs wc.
Then the output from the ls command is used as the input of the wc command.

5 marks

Question 15.

What does "Request – Reply – Release" mean and what is it used for?

Request a resource by sending a message.
Wait for the reply to say that the resource is yours.
Send a release message when the resource is free.
It is used to coordinate access to a resource. Typically in a distributed environment with a coordinator process controlling access.

5 marks

Question 16.

Having unique timestamps for events is very important in distributed environments. Describe a simple scheme that can be used to keep timestamps from different machines ordered according to the happened-before relation.

Logical clocks. Each machine maintains its own logical clock. Every time a message from another machine is received the timestamp is compared with the local logical clock. If the timestamp is greater than the local time then the local time is advanced.

5 marks