Data Provided: None

The University Of Sheffield.

DEPARTMENT OF ELECTRONIC AND ELECTRICAL ENGINEERING

Autumn Semester 2009-2010 (2 hours)

Operating Systems 6

Answer **THREE** questions. **No marks will be awarded for solutions to a fourth question.** Solutions will be considered in the order that they are presented in the answer book. Trial answers will be ignored if they are clearly crossed out. **The numbers given after each section of a question indicate the relative weighting of that section.**

1. **a.** Describe the spawning of a child process using the fork system call under the UNIX operating system. **(7)**

b. How many processes in total will be produced by running the code in Figure 1? Explain your reasoning.

```
int main()
{
fork();
fork();
fork();
return 0;
}
```

Figure 1: Example code **(5)**

c. A multi-programmed operating system can run several instances of the same program. How can the operating system differentiate between them? **(2)**

d. Why do many operating systems use spinlocking for process synchronisation on multiprocessor systems but not on single processor systems? **(6)**

2. **a.** What is the purpose of the command interpreter? Why is the command interpreter normally separate from the operating system kernel? **(7)**

b. Nowadays, what are the principal advantages of a virtual machine? **(7)**

c. Virtual machines (VMs) are often categorised into: hardware VMs, system-level VMs and application-level VMs. Explain what each of these terms mean and give an example of each kind of virtual machine. **(6)**

3. **a.** Describe the actions taken by the kernel during a context switch between processes. **(2)**

A number of processor architectures have a multiple register sets. Why is this useful to an operating system? **(4)**

b. Is there any speed advantage to splitting a process in to multiple threads of execution? What factors determine whether there is any advantage to multi-threading? **(6)**

c. In respect of counting semaphores, how are the wait and signal operations defined? Show that unless these two operations are performed atomically mutual exclusion cannot be guaranteed between two processes both waiting on the same semaphore. **(8)**

4. **a.** What is the cause of disk thrashing in a paged virtual memory system? How can the operating system detect page thrashing and, once detected, what steps can the operating system take to eliminate it? **(8)**

b. Explain the operation of the open() and close() file operations in a multi-programmed operating system. How can this mechanism be extended to prevent two (or more) processes from both gaining write-access to the same file. **(6)**

c. If a process terminates abnormally (*i.e.* with an error), how does the operating system ensure that any files which the process has opened are closed and thus available to future processes? **(6)**

PIR