UNIVERSITY OF EDINBURGH

FACULTY OF SCIENCE AND ENGINEERING

COMPUTER SCIENCE THREE

OPERATING SYSTEMS

Wednesday 24 May 2000

14:00 to 15:30

Examiners:

Prof. K. Turner (External)

Prof. G. Brebner (Chairman)

INSTRUCTIONS TO CANDIDATES

Answer any TWO questions.

All questions carry equal weight.

1. (a) Illustrate what might go wrong if two separate processes use a shared variable without any synchronisation. How does the concept of a *critical section* help deal with such problems? [*3 marks*]

   (b) How might the ability for a process to disable processor interrupts help to achieve mutual exclusion of processes from a critical section? Why would this be a poor idea? [*4 marks*]

   (c) Peterson's solution to the mutual exclusion problem between two processes using only shared variables is of the form:

```
        shared int c1=1, c2=1, turn = 1;

        // process 1                     // process 2
        while (TRUE) {                   while (TRUE) {
<1>        c1 = 0;                          c2 = 0;
<2>        turn = 2;                        turn = 1;
<3>        while (c2==0 && turn==2);         while (c1==0 && turn==1);
<4>        // critical section 1             // critical section 2
<5>        c1 = 1;                           c2 = 1;
          // rest of loop                   // rest of loop
        }                                 }
```

   Explain informally how this works. [*4 marks*]

   (d) Using the expressions:
   $H_1 = ((turn == 2) \vee ((turn == 1) \wedge P_2@ < 3 >)) \wedge (c1 == 0)$
   $H_2 = ((turn == 1) \vee ((turn == 2) \wedge P_1@ < 3 >)) \wedge (c2 == 0)$
   where $P_n@ < m >$ means process n is executing line m, outline Dijkstra's formal verification of Peterson's solution. [*8 marks*]

   (e) Explain Lamport's 'Bakery' algorithm which solves the mutual exclusion problem for multiple processes. [*6 marks*]

2. (a) List objectives that the designer of a CPU scheduling strategy should aim to meet in each of the following contexts:

   i. a single-user workstation
   ii. a multiple-user interactive server
   iii. a background job-stream server
   iv. a real-time system

   [*6 marks*]

(b) Describe the operation of the following scheduling schemes and the benefits that each may have:

   i. first-come-first-served
   ii. round-robin
   iii. preemptive priority
   iv. non-preemptive priority

   [*8 marks*]

(c) What are *hard* and *soft* deadlines in real-time systems? Give examples where the need for each arises. Describe three scheduling schemes which are appropriate for real-time systems. [*8 marks*]

(d) What is meant by the term *priority inversion* and how can the problems it causes be alleviated? [*3 marks*]

3. (a) Describe what is meant by virtual memory *paging* and how the associated *page tables* are used. What problems does paging overcome and how does it do so? What components does each entry in a page table hold? Describe what each component is for. [*7 marks*]

   (b) What is meant by the terms *page fault* and *demand paging*? Describe the sequence of processing stages, both hardware and software, that occur when a virtual page address is accessed in a demand paged system, in particular when the page is not present in physical main memory. [*7 marks*]

   (c) How are the flags in a page table entry used in the *clock* page replacement scheme? Explain how the *strobing* of page table usage information can be used to approximate a Least Recently Used page replacement policy. [*7 marks*]

   (d) Distinguish between *local* and *global* page replacement policies and outline the benefits and drawbacks of each. [*4 marks*]