UNIVERSITY OF EDINBURGH

COLLEGE OF SCIENCE AND ENGINEERING

SCHOOL OF INFORMATICS

INFR09047 OPERATING SYSTEMS

Tuesday 30th April 2019

09:30 to 11:30

INSTRUCTIONS TO CANDIDATES

Answer any TWO of the three questions. If more than two questions are answered, only QUESTION 1 and QUESTION 2 will be marked.

All questions carry equal weight.

CALCULATORS MAY NOT BE USED IN THIS EXAMINATION

Year 3 Courses

Convener: C. Stirling
External Examiners: S.Rogers, S. Kalvala, H.Vandierendonck

THIS EXAMINATION WILL BE MARKED ANONYMOUSLY

1. (a) Consider six processes, P0, P1, P2, P3, P4, P5 available to start execution at times 0, 1, 2, 3, 4, 5. They have the following execution times: P0: 11 cycles, P1: 11 cycles, P2: 41 cycles, P3: 3 cycles, P4: 7 cycles, P5: 12 cycles.

   Calculate the average waiting time and average turnaround time when using the following scheduling policies: First-come-first-served, round-robin with quantum=10, shortest remaining time first. [*6 marks*]

   Note that: the average waiting time of a schedule, is the average time before a process is scheduled to start execution and the average turn-around time is the average time for each process to complete.

   (b) In virtual memory, six page frames are currently available, and the operating system employs the least recently used (LRU) page replacement algorithm. A process has a code section that accesses seven pages sequentially. Initially the seven pages are stored only on disk.

   If the process executes this code snippet five times, how many page faults will occur? [*3 marks*]

   Using Belady's algorithm, calculate the minimal number of page faults possible. [*3 marks*]

   How does this compare to a most recently used (MRU) page replacement policy? [*3 marks*]

   (c) A single-platter disk has 220 tracks (number 1 at the inside, 220 at the outside). At the current time, the head is stationary at track 100, after moving there from an inner track. The disk request queue contains requests for tracks (front) [55 , 58, 39, 90, 160, 38, 184, 77, 1, 190] (back).

   For each of the five mentioned algorithms (FCFS, SSTF, SCAN, C-SCAN, C-LOOK), show the sequence of moves made to service the request queue [*5 marks*]

   (d) There are many process scheduling policies. In your opinion, what policy is most suitable for an Android based smart-phone? Explain why. [*5 marks*]

2. (a) Draw or describe how the following are implemented in hardware by computer systems. Highlight any additional resources required such as tables or registers.

   i. logical to physical address translation [*1 mark*]

   ii. segmented memory [*2 marks*]

   iii. paged memory [*2 marks*]

   (b) What is memory fragmentation? Why is it important? [*3 marks*]

   (c) How is fragmentation handled in

   i. Buddy allocator [*1 mark*]

   ii. Segmented memory [*2 marks*]

   iii. Paged memory [*2 marks*]

   (d) Assume a single-level page table where each page table entry occupies 4 bytes. A virtual address is 32 bits long and a page comprises 2KB (2^{11} bytes). How big is the page table? [*2 marks*]

   (e) i. Suggest two solutions to reduce the page tables size. Calculate the new page table size in both cases. [*4 marks*]

   ii. Which solution is preferable? Explain why? [*2 marks*]

   (f) Indirection (in other words pointers to other structures) is used in page based virtual memory and in file systems. What problem are they trying to solve? Highlight any similarities and differences in the problem addressed and their solutions. [*4 marks*]

3. (a) You are supplied with an atomic `test_and_set` intrinsic function from your cpu vendor:

```
bool test_and_set(bool *flag) {
 bool old = *flag;
 *flag = True;
 return old;
}
```

   i. Using this intrinsic write two simple spin-lock functions, LOCK() and UNLOCK(), in C or pseudo-code [*4 marks*]

   ii. Using LOCK() and UNLOCK() write two binary semaphore functions ACQUIRE() and RELEASE() in C or pseudo-code which avoids excessive busy waiting. You have available 2 functions: pause() which puts the current process on a wait queue and restart() which removes the first available process from the wait queue and executes it. [*4 marks*]

   iii. Explain why your ACQUIRE() and RELEASE() are more efficient than LOCK() and UNLOCK(). [*2 marks*]

   (b) How do monitors help remove problems associated with low level synchronisation? [*2 marks*]

   (c) The following piece of pseudo-code is intended to implement a producer-consumer program with a bounded buffer.

```
Monitor bounded_buffer {
 buffer resources[N];
 procedure add_entry(resource x) {
  insert x in array resources
 }
 procedure get_entry(resource *x) {
   x = get resource from array resources
 }
}
```

   i. What problem is there with this code? [*2 marks*]

   ii. Modify the code to solve this problem. [*5 marks*]

   (d) Assume we have two processes P1, P2 and 2 resources R1, R2. P1 hold R1 and requests R2, P2 holds R2 and requests R1

   i. Draw or describe how this creates deadlock. [*2 marks*]

   ii. How does Banker's algorithm prevent this from happening? [*4 marks*]