# COMP3301: 2023 Exam solutions

## [UQAttic](http://uqattic.net/)

## Get more out of your study time. Join UQAttic's revision chat. Or [Slack](http://slack.uqcs.org.au), if you’re a better man.

#### [Other exam papers](https://docs.google.com/folder/d/0B6_D4T6LJ-uwZDQyMDYwY2QtMmFlNC00MWYyLWI4OWUtNzk3NWYwZmZlMjk5/edit)

### Please contribute to these documents.

### If you're looking for an effective way to familiarise yourself with the course material, you can't go past collaborating with fellow students. We have laboured to put these up, and so at the very least point out where you think we are wrong!

### You'll get more out of the course, you'll do better in the exam, and other students will benefit from your input as well.

### To get editing permissions, simply go to the [chatroom](http://uqattic.net) and provide us with your Google Account address.e

### Style.

### Type answers in blue beneath each question.

### If you're unsure of your answer, highlight your answer text then hit Ctrl+Alt+M to create a comment beside the text. Once you're satisfied with the answer, click the "Resolve" button on the comment.

### If you want some extra explanation from someone else on their answer, highlight the other person's answer and repeat the procedure above.

### Communicate.

### Head over to [uqattic.net](http://uqattic.net/) and click "Chat Now!". You'll find a chatroom full of students just like you. Talk about a revision document (like this one) or swap prep tips. If you have your own IRC client, point it to irc.uqattic.net, port 6667, channel #attic.

# Question 1. (20 marks)

Calculate performance metrics for various process schedulers.

All of the schedulers are pre-emptive – a process is pre-empted as soon as a higher priority process appears. The larger the priority number, the higher the priority level.

In the following tables, times are all in milliseconds.

Assume processes do not stall for I/O. Ignore scheduler delays. The CPU time of aborted processes is included in CPU utilisation.

1. Complete the following tables for a Round Robin (RR) Scheduler with a time quantum of 4ms.

| Process  Number | Arrival  Time | Execution  Time | Priority | Completion  Time | Turnaround  Time | Waiting  Time |
| --- | --- | --- | --- | --- | --- | --- |
| P1 | 0 | 4 | 1 | 4 | 4 | 0 |
| P2 | 5 | 5 | 2 | 21 | 16 | 11 |
| P3 | 6 | 1 | 2 | 10 | 4 | 3 |
| P4 | 10 | 4 | 3 | 20 | 10 | 6 |
| P5 | 13 | 6 | 4 | 19 | 6 | 0 |
| P6 | 14 | 4 | 1 | 25 | 11 | 7 |

| Average Waiting Time | 4.5 |
| --- | --- |
| Average Turnaround Time | 8.5 |
| Actual CPU Utilisation | 24/25 = 96% |
| Average Throughput | 6 jobs/25 ms = 240 jobs/s |

Process Timeline Table. Mark where a process Starts (S), Runs (R), Pauses (P), Ends (E). You can add more columns if required.

|  | 0 | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9 | 10 | 11 | 12 | 13 | 14 | 15 | 16 | 17 | 18 | 19 | 20 | 21 | 22 | 23 | 24 | 25 |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| P1 | SR | R | R | R | E |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| P2 |  |  |  |  |  | SR | R | R | R | P | P | P | P | P | P | P | P | P | P | P | R | E |  |  |  |  |
| P3 |  |  |  |  |  |  | SP | P | P | R | E |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| P4 |  |  |  |  |  |  |  |  |  |  | SR | R | R | P | P | P | P | P | P | R | E |  |  |  |  |  |
| P5 |  |  |  |  |  |  |  |  |  |  |  |  |  | SR | R | R | R | PR | R | E |  |  |  |  |  |  |
| P6 |  |  |  |  |  |  |  |  |  |  |  |  |  |  | SP | P | P | P | P | P | P | R | R | R | R | E |

^^^ why does P5 run for 6ms?

^^^see #937 on (2024) ed. Thx :)

1. Repeat part a) but now assume that two CPU cores (core 1 and core 2) can be used to run processes, concurrently. Complete the following tables for a Round Robin (RR) Scheduler with a time quantum of 4ms. You must consider the core affinity (core 1, core 2 or X – any core). You can assume that a process with no affinity (X) can run on any core. What is the meaning of X, how can we choose for that? Let them run alternately，like 1,2,1,2,1,2 ?**HELP** The X means that it can run on either core, so schedule it on any core that’s free when it arrives (or the next core that becomes free after it arrives)THX

| Process Number | Arrival Time | Execution Time | Priority | Core Affinity | Completion Time | Turnaround Time  (Completion time - arrival time) | Waiting Time |
| --- | --- | --- | --- | --- | --- | --- | --- |
| P1 | 0 | 4 | 1 | 1 | 4 | 4 - 0 = 4 | 0 |
| P2 | 5 | 5 | 2 | 2 | 10 | 10 - 5 = 5 | 0 |
| P3 | 6 | 1 | 2 | X | 7 | 7 - 6 = 1 | 0 |
| P4 | 10 | 4 | 3 | 2 | 14 | 14 - 10 = 4 | 0 |
| P5 | 13 | 6 | 4 | 1 | 19 | 19 - 13 = 6 | 0 |
| P6 | 14 | 4 | 1 | X | 18 | 18 - 14 = 4 | 0 |

| Average Waiting Time | 0 |
| --- | --- |
| Average Turnaround Time | 4 ms |
| Actual CPU Utilisation | (4 + 5 +1 + 4 + 6 + 4) / (19 \* 2) = 0.6316 = 63.16% |
| Average Throughput | 6 / 0.019 = 315.79 = 315 processes per second |

Process Timeline Table (Mark where a process Starts (S), Runs (R), Pauses (P), Ends (E). You can add more columns if required.

Core 1

|  | 0 | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9 | 10 | 11 | 12 | 13 | 14 | 15 | 16 | 17 | 18 | 19 | 20 | 21 | 22 | 23 | 24 | 25 | 26 | 27 | 28 | 29 | 30 | 31 | 32 | 33 | 34 | 35 | 36 | 37 | 38 |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| P1 | SR | R | R | R | E |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| P5 |  |  |  |  |  |  |  |  |  |  |  |  |  | SR | R | R | R | R | R | E |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| P3 |  |  |  |  |  |  | SR | E |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| P6 |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |

Core 2

|  | 0 | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9 | 10 | 11 | 12 | 13 | 14 | 15 | 16 | 17 | 18 | 19 | 20 | 21 | 22 | 23 | 24 | 25 | 26 | 27 | 28 | 29 | 30 | 31 | 32 | 33 | 34 | 35 | 36 | 37 | 38 |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| P2 |  |  |  |  |  | SR | R | R | R | PR | E |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| P4 |  |  |  |  |  |  |  |  |  |  | SR | R | R | R | E |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| P3 |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| P6 |  |  |  |  |  |  |  |  |  |  |  |  |  |  | SR | R | R | R | E |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |

# Question 2. (20 marks)

Suppose that a disk drive has 2000 cylinders, numbered 0 to 1999. The disk rotates at 250 rpm. The drive is currently serving a request at cylinder 103, and the previous request was at cylinder 101. The queue of pending requests, in FIFO order, is:

201; 210; 1800; 530; 1500; 300; 1200; 655;

For each of the indicated disk scheduling algorithms (a) to (b) below, calculate the following information:

* 1. Starting from the current head position, where a block has just been read, and assuming the previously read block was in cylinder 101, what is the order in which pending blocks are read?
  2. Assume that a disk seek takes 10ms + 5 µs per cylinder. Assume that the rotational latency is 50% of the rotation time, and also assume that there are 500 blocks per track. Based on the block read order, calculate the total time to read each block (latencies + read time).
  3. Then calculate the total time to read the 8 blocks, starting from the time the read at 103 was completed.
     1. **FCFS**

**rps = (rpm / 60)**

**spr = 1 / rps**

**Half spr because the question tells you to “rotational latency is 50% of the rotation time”, => rotational latency = spr / 2**

**Rotational latency = (1/(250/60))/2 = 0.12 seconds = 120ms**

**Block read time = time to read track / # blocks per track = 240/500 = 0.48 ms per block**

**Seek time = rotational latency + 10ms + cylinders \* 5µs**

**rpm = 250 , time to read 500 blocks (1 track) = 1/250 minutes = 240 ms**

**rotational latency = 240 / 2 = 120 ms**

**+ block read time = 240/500 = 0.48 (for each read)**

**total read time = 120 + 0.48 + 10 + (cylinders \* 0.005)**

**(My method)**

**Seek time: Disk Seek Time \* (Block distance)**

**Block Read time: Seek Time + Rotational Latency + Block Read Time**

**E.g 201 :**

**Seek time: 10ms + 0.005ms \* (201 - 101) = 10.5**

**Block Read Time: 10.5 + 120 + 0.48 = 130.98**

* + - 1. 

| 103 | 201 | 210 | 1800 | 530 | 1500 | 300 | 1200 | 655 |
| --- | --- | --- | --- | --- | --- | --- | --- | --- |
| (ii)Block Read Time These numbers (except total time) are off by 0.48, add 0.48 ms to each number and it’ll be right +1 true. | | | | | | | | |
|  | 120 + 10 + (`) \* 0.005 = 130.49ms | 120 + 10 + (210 - 201) \* 0.005 = 130.045 ms | 120+10+(1800-210)\*0.005 = 137.95ms | 120 + 10 + (1800 - 530) \* 0.005 = 136.35 ms | 120 + 10 + (1500 - 530) \* 0.005 = 134.85 ms | 120+10+(1500-300)\*0.005 = 136ms | 120 + 10 + (1200 - 300) \* 0.005 = 134.5 ms | 120 + 10 + (1200 - 655) \* 0.005 = 132.725 ms |
| (iii) Total Time 1072.91ms + (0.48 \* 8) = 1076.75 | | | | | | | | |

* + - 1. Block Read Order:
    1. **SCAN**
       1. Block Read Order:

| 103 | 201 | 210 | 300 | 530 | 655 | 1200 | 1500 | 1800 |
| --- | --- | --- | --- | --- | --- | --- | --- | --- |
| (ii)Block Read Time These numbers (except total time) are off by 0.48, add 0.48 ms to each number and it’ll be right | | | | | | | | |
|  | 120 + 10 + (201-103) \* 0.005 = 130.49ms | 120 + 10 + (210 - 201) \* 0.005 = 130.045 m | 120 + 10 + 90 \* 0.005 = 130.45 ms | 120 + 10 + (530 - 300) \* 0.005 = 131.15 ms | 120 + 10 + (655 - 530) \* 0.005 = 130.625 ms | 120+10+(1200-655)\*0.005 = 132.725 | 120 + 10 + 300 \* 0.005 = 131.5 ms | 120+10+(1800-1500)\*0.005 = 131.5 ms |
| (iii) Total Time 1048.485 ms + (0.48 \* 8) = 1052.325 | | | | | | | | |



Question 2 (continued)

| 1. For a magnetic disk, describe what the following terms are.    1. Read/Write Head   The physical assembly which moves radially and changes the magnetic properties of the disk to store or read information.   * 1. Sector   The smallest subdivision of the disk, it sits within a track.   * 1. Track   A circle of sectors on one platter   * 1. Cylinder   Tracks which are vertically aligned with one another.   * 1. Spindle   The spindle is the rotating element at the center of the platters which allows the platters to spin. | | |
| --- | --- | --- |
| spin spin kuru kuru | /5 |  |

| 1. For magnetic disk performance, write the equations to calculate the average access time and the average IO time. Use the terms: *average seek time, average latency, transfer amount, transfer rate and controller overhead*    1. Average access time = +   Average access time = seek time + average latency   * 1. Average IO time = Average access time + ( / ) +   = Average access time + (transfer amount/transfer rate) + controller overhead | | |
| --- | --- | --- |
|  | /5 |  |

# Question 3. (20 marks)

A process accesses the following data memory pages, as seen in the table below. During each iteration, the corresponding pages are accessed, e.g. pages 0 and 1 are accessed in iteration 0.

Table showing Memory Pages Numbers accessed

| Iteration 0 | 0 | 1 | 1 | 2 | 3 |
| --- | --- | --- | --- | --- | --- |
| Iteration 1 | 1 | 1 | 2 | 3 | 5 |
| Iteration 2 | 2 | 1 | 3 | 4 | 7 |
| Iteration 3 | 3 | 1 | 4 | 5 | 9 |
| Iteration 4 | 4 | 1 | 5 | 6 | 11 |

| **(a)** At the end of iteration 2, calculate the total number of page faults generated if the First In First Out (FIFO) page replacement algorithm is used, with **i)** 3 page buffer and **ii)** 5 page buffer. Also state what the minimum buffer size is to ensure that no page faults occur.  i) 8 [+4]   | Buff 1 | 0 |  |  | 3 |  |  | 4 |  |  | | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | | Buff 2 |  | 1 |  |  | 5 |  |  | 7 |  | | Buff 3 |  |  | 2 |  |  | 1 |  |  |  |     ii) 7 [+4]   | Buff 1 | 0 |  |  |  |  | 4 |  | | --- | --- | --- | --- | --- | --- | --- | --- | | Buff 2 |  | 1 |  |  |  |  | 7 | | Buff 3 |  |  | 2 |  |  |  |  | | Buff 4 |  |  |  | 3 |  |  |  | | Buff 5 |  |  |  |  | 5 |  |  |   iii) minimum buffer size = 7   | Buff 1 | 0 | | --- | --- | | Buff 2 | 1 | | Buff 3 | 2 | | Buff 4 | 3 | | Buff 5 | 5 | | Buff 6 | 4 | | Buff 7 | 7 | | | |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
|  | /5 |  |

| **(b)** At the end of iteration 3, calculate the total number of page faults generated if the Least Recently Used (LRU) page replacement algorithm is used, with **i)** 3 page buffer and **ii)** 5 page buffer. Also state what the minimum buffer size is to ensure that no page faults occur.  i) 13 +3   | B1 | 0 |  |  | 3 |  | 1 |  |  | 7 |  | 4 |  |  | | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | | B2 |  | 1 |  |  | 5 |  | 3 |  |  |  |  | 5 |  | | B3 |  |  | 2 |  |  |  |  | 4 |  | 1 |  |  | 9 |     Missing a 011 at t3 but its a no fault so doesnt change the output  ii) 9 +3 I had 10 due to 3 being pushed out by 7 and needing to be reallocated after7  Why would 3 get pushed out by 7? Wouldn't we need to look at the last 4 used pages as we have a 5 page buffer?   | B1 | 0 |  |  |  |  | 4 |  |  |  | | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | | B2 |  | 1 |  |  |  |  |  |  |  | | B3 |  |  | 2 |  |  |  |  | 5 |  | | B4 |  |  |  | 3 |  |  |  |  |  | | B5 |  |  |  |  | 5 |  | 7 |  | 9 |     iii) minimum buffer size = 8 | | |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
|  | /5 |  |

Question 3 (continued)

| **(c)** Describe three possible solutions to minimise the number of page faults. At least two solutions should assume no more physical memory is available.  i) Larger buffer -1 a larger buffer would not meet the assumption. What assumption?  “No more physical memory”, not a concern anymore with the other two responses  ii) Increase page size.  iii) Least frequently used page replacement algorithm.or LRU | | |
| --- | --- | --- |
|  | /3 |  |

| **(d)** For memory paging, explain what happens to a ‘dirty’ victim frame?  Dirty means it has been changed while it was being stored, so it needs to be flushed to the disk prior to replacing the page.  A dirty page is a page where the modify bit has been set (done by the hardware whenever any byte in the page is written to), if a dirty page has been selected for replacement it must be written to storage before being replaced. | | |
| --- | --- | --- |
|  | /3 |  |

| **(e)** Explain how the LRU page replacement algorithm is implemented with a counter. You must also explain how a page is replaced using the LRU page replacement counter.  Every page has a counter, the counter is the clock at the latest reference, when replacement is needed, find the lowest value. | | |
| --- | --- | --- |
|  | /4 |  |

# Question 4. (20 marks)

| **(a)** Give two examples of what a process can be doing in a critical section. i) Changing shared variables +1 “Writing to shared memory” < i like this answer more  ii) Writing to a file +1 “Reading from shared memory that someone else can write to”  From the slides | | |
| --- | --- | --- |
|  | /2 |  |

| **(b)** Write example pseudo code for a critical section. It must use a critical, entry, exit and remainder sections.  do {  entry section  critical section  exit section  remainder section  } while (true) | | |
| --- | --- | --- |
|  | /4 |  |

Question 4 (continued)

| 1. Explain what the following critical section solutions are.    1. Mutual Exclusion (mutex :3) :3 :3 :3 :3   Only one process is allowed in the critical section at any time   * 1. Progress   If no process is currently in its critical section, and there are processes waiting to enter their critical sections, then one will be selected, pre infinitum.   * 1. Bounded Waiting   A process can only be refused entry into its critical section at most N times.  9o | | |
| --- | --- | --- |
|  | /6 |  |

| **(d)** What is an atomic instruction and list two possible operations of atomic instructions.  An atomic instruction is one that cannot be interrupted in the middle of its execution  load and store, not mentioned in lecture slides, however reading and writing to a register is atomic.  test\_and\_set - Sets a bit in memory to 1 and returns its previous value  compare\_and\_swap - Checks the contents of a word memory against an expected value, and if they match, swaps it with the register  there’s also read-modify-write, load-linked/store-conditional, but those are too modern for this course obviously | | |
| --- | --- | --- |
|  | /6 |  |

| **(e)** When a process enters its critical section, what hardware feature must be disabled?  **Interrupts** must be disabled, because otherwise an interrupt handler could alter data that would otherwise be protected by whatever mechanism is protecting the critical section.  Me when I CSSE2010.  Me when I lock a mutex in userland and now my process cannot be preempted (I am writing malware and can now do whatever I want to your system :3)  She lock on my mutex till I critical section  me when cheese :0 | | |
| --- | --- | --- |
|  | /2 |  |

Dead set

# Question 5 (20 marks)

* + - * 1. For the following access permissions, fill in the below access matrix. Use R for Read, W for Write, S for Switch, X for Execute, O for Owner and \*\* for Copy Rights.

When in Domain1 (D1), a user can execute object 2 (O2) and read object 1 (O1).

When in Domain 2 (D2), a user can write object 1 (O1) and execute object 2 (O2).

When in Domain 3 (D3), a user can read object 3 (O3) with ownership and switch to Domain 2 (D2).

When in Domain 4 (D4), a user can switch to Domain 3 (D3) and copy the write permission of Object 3 (O3).

When in Domain 2 (D2), a user can access the printer.

# Fill in spaces with answers

| Object | O1 | O2 | O3 | Printer | D1 | D2 | D3 | D4 |
| --- | --- | --- | --- | --- | --- | --- | --- | --- |
| Domain |
| D1 | R | X |  |  |  |  |  |  |
| D2 | W | X |  | RWX?  PRINT +3  R |  |  |  |  |
| D3 |  |  | RO |  |  | O S[-1]  S +1 |  |  |
| D4 |  |  | W\*\* |  |  |  | S |  |

**D4 can copy write permission only, not for example execute so it should be W\*\* not \*\***

**D3 is the owner of O3 and so O should be in (D3,O3)**

**Never said D3 needs ownership of D2 so i don't know why D3,D2 is O S**

| **(b)** For a lock-key protection access matrix, what elements are the locks and keys used for?  Locks are for objects, keys are for domains. | | |
| --- | --- | --- |
|  | /2 |  |

Question 5 (continued)

| For an access matrix, describe the four special access rights, below.   1. ?Owner   Has absolute control over an object.   1. Copy Operation   Copy an operation between objects in the access matrix  Isn’t it the ability to grant a given permission for a given object to another domain?  In a), domain 4 can grant W\*\* privileges to any domain for Object O3 +1  In an access matrix, a "copy" operation/right represents the ability to copy an access right from one domain-object pair to another domain-object pair. It's typically denoted as "copy op" where op is the right being copied.   1. Control   One domain can modify access rights of other domains for a specific object   1. Transfer   Switch from one domain into another  Allows a domain to transfer its own rights to another domain  The key difference from copy is that the original domain LOSES the right when it's transferred | | |
| --- | --- | --- |
|  | /8 |  |

# Question 6. (20 marks)

You have been asked to design an operating system for a web based AI Chatbot called ChatHPT. ChatHPT can be used for many applications, including generating code. ChatHPT uses a large language model and requires RAID storage. ChatHPT is accessed by users via a web page interface that takes user input and returns the output. As a free service, ChatHPT must be able to service millions of users, every hour. ChatHPT is also a target for malicious users and therefore needs to ensure that only authorised users and administrators can access ChatHPT. To save power costs and to achieve carbon neutral usage, the power to run ChatHPT is supplied by solar panels and battery storage. This means that power outages can occur for short periods of time and without warning. This may cause errors in files that are used. The operating system should use multiple CPU cores for parallelism.

You can also assume the following:

The operating systems uses multiple CPU cores for redundancy.

Magnetic disk mass storage is used, to save co  
“?st. For each of the indicated aspects of the operating system:

1. Describe your suggested design alternative,
2. Give one other less favourable choice,
3. Explain two advantages of the suggested alternative compared to the less favourable choice

| 1. What type of Parallelism should be used (Data vs Task)?    1. Suggested solution    2. Less favoured alternative    3. Two advantages of suggested solution |
| --- |
| (i) Task parallelism +1  (ii) Data parallelism  (iii) As a part of the service is a webserver and another part is a chat bot, we need to split tasks. There isn’t really 2 advantages, just one works and the other doesn’t. Good luck running a web server on a GPU  (iii): task parallelism allows us to split client handler threads off and deal with requests in parallel. Forward passing through a neural network isnt very intensive (compared to backward passing) and doesnt require much data so splitting off data portions and computing separately is no use. The only other data we deal with is user input which is a short string, so data parallelism isnt particularly useful.  (i) Data parallelism  (ii) Task parallelism  (iii) Data parallelism treats each request individually and decides what operation to do on it. Task parallelism would have to separate all the different request types into different threads (say, “redirect to https”, “get homepage”, “calculate AI response”) and send incoming requests to those threads, meaning only one of each can be served at a time which is a massive bottleneck  Data parallelism is also easier to implement in this case, since we don’t need to set up that mess of queues as would be needed for task parallelism |

Question 6 (continued)

| 1. What type of CPU scheduler should be used?    1. Suggested solution    2. Less favoured alternative    3. Two advantages of suggested solution |
| --- |
| (i) Round Robin or FCFS, essentially the same here as a GPU would handle the intensive AI operations, all the CPU needs to do is handle authentication and HTTP requests.  (ii) Rate monotonic, nothing here is periodic.  (iii) Lower overhead as it’s FCFS, low CPU impact hence maybe less power????  Both are not suitable i think X)  Round Robin yeah, but FCFS nah. Round Robin is good because most of the clients are doing the same thing so having a quantum would be cool, but with FCFS some of the processes could just not get time. Lachlan confirmed in Ed post as well.  ^^ I mean RR and Rate monotonic are not suitable  (i):SJF  Could this not cause starvation? A bunch of small requests causing the larger ones to never get processed. Starvation +1  (ii): FIFO  (iii): SJF allows questions with short responses to be handled earlier since although the gpu does the work the text output still has to be sent. Further, since we are sending a text output its easy to compute the length of the job.  Tutor has told me that as long as you give justification and demonstrate that you understand the pros and cons of whatever solution you pick (as long as it’s reasonable) that’s probably fine |

Question 6 (continued)

| (c) What security mechanism should be used to ensure that only authorised users and administrators can have access?  (i) Suggested solution  (ii) Less favoured alternative  (iii) Two advantages of suggested solution |
| --- |
| (i) authentication - user login + secure password  (ii) uhm uhm digital signature? Pure trust  (iii) identifies user  .  i) Role-Based Access Control  ii) Access Control Lists  iii) -Adding millions of new users simply requires assigning them to appropriate roles.  -Changes to permissions can be made at role level affecting all users in that role. -Role checking is efficient as roles are typically cached and there are relatively few roles to check |

Question 6 (continued)

| (d) What RAID Disk level (0 to 6) should be used?  (i) Suggested solution (ii) Less favoured alternative  (iii) Two advantages of suggested solution |
| --- |
| (i) RAID level 5  (ii) RAID level 4  (iii) RAID 5 provides better performance on writes due to spread out parity blocks rather than on only one drive, so less computing time spent on recalculating parity.  Takes up less storage than RAID 6 due to reduced redundancy information. |