**Student Name: Ahmad Allahham**

**Student ID: 19170251**

*Please do NOT change the order of Question/Answer. If you are unable to answer any question, do not delete the Question/Answer number.*

**QUESTION ONE (total: 20 marks): Deadlock**

**Q1 a) (Total: 6 marks).**

Q1 a) (i) (2 marks).

**Answer:**

If every process is given one different resource from the n available resources, then all process will be left waiting for their second resource and all process will be in deadlock.

Q1 a) (ii) (4 marks).

**Answer:**

To guarantee that no process will be stuck in a hold-and-wait situation, each process should be granted all of the resources it needs prior to starting. Once the process is completed, it will release all of the resources it was holding so that they can be used by other processes. This ensures that a process only requests resources when it has none. If it is holding onto a resource and it needs more resources to execute that are not available, it must release all of the resources it is holding then request all of the resources necessary for its execution.

**Q1 b) (Total: 4 marks).**

Q1 b) (i) (2 marks).

**Answer:**

No a deadlock cannot occur, as long as the order of resource requests is maintained and no resource is granted to a process out of order.

Q1 b) (ii) (2 marks).

**Answer:**

Yes, if resource R1 is given to A and resource R2 is given to B, then both processes will be waiting for resources that they cannot obtain.

**Q1 c) (4 marks).**

**Answer:**

Banker’s algorithm can only work if the request for resources made by a process is less than the process’s needs. This means that a process cannot ask for more resources than it initially outlined, which is unrealistic as consumer’s wants for example may increase past its expected needs.

The second assumption that Banker’s algorithm depends on is that the request for resources made by a process must be less than the available resources. If a safe state is not achievable, the algorithm will not allocate any resources to the requesting process, as its request exceeds the available resources. In the real world where a safe state is not possible, a compromise is sometimes preferred, giving some/all resources to the requesting process, even if it results in an unsafe state.

**Q1 d) (Total: 6 marks).**

Q1 d) (i) (4 marks).

**Answer:**

Request (P0) = A(1)

Request (P0) < Available(A)

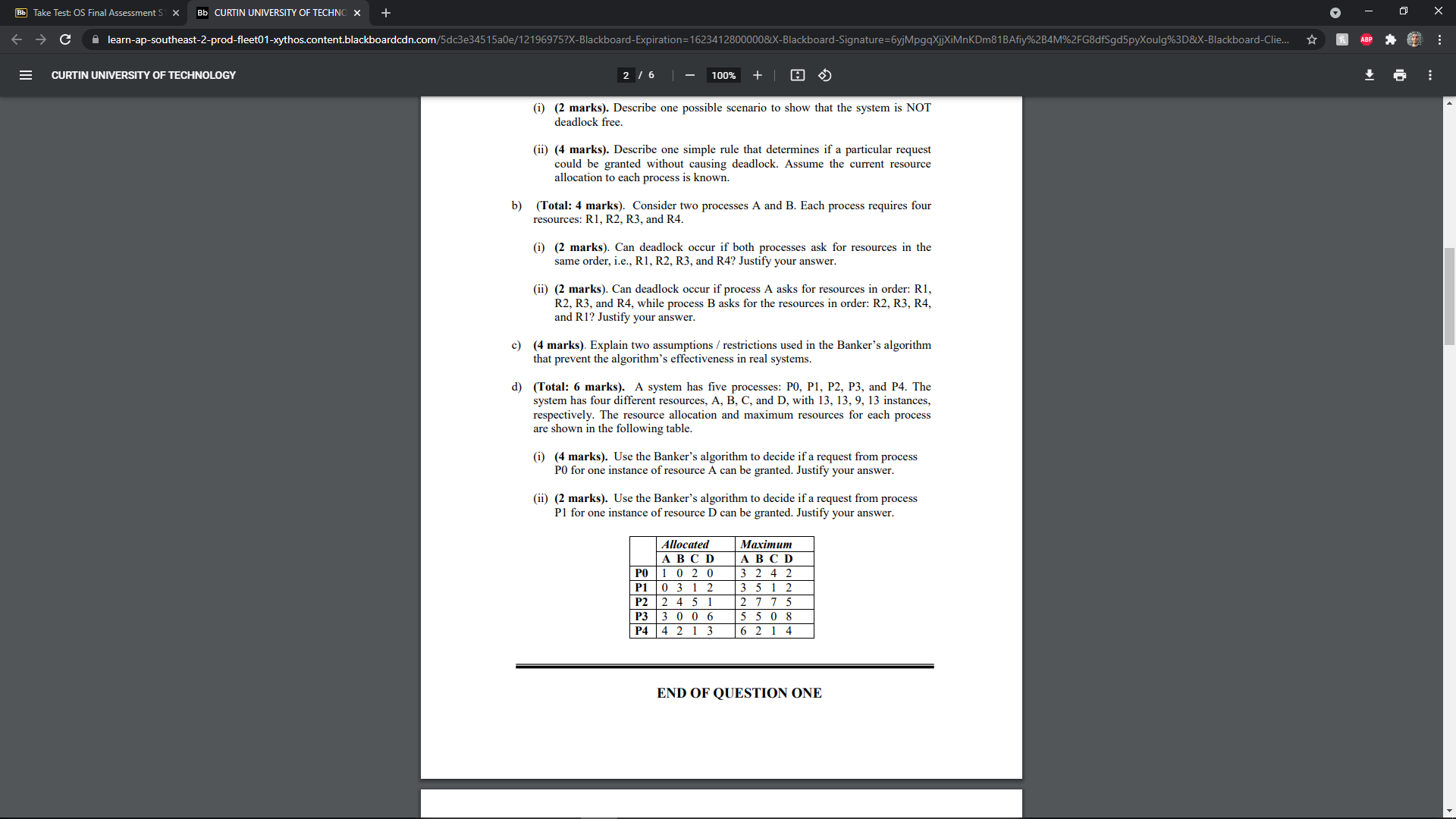
Request (P0) < Maximum(A)

The new system state:

|  |  |  |  |
| --- | --- | --- | --- |
| Available (after request) | | | |
| A | B | C | D |
| 2 | 4 | 0 | 1 |

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
|  | Need (after request) | | | |
|  | A | B | C | D |
| P0 | 1 | 2 | 2 | 2 |
| P1 | 3 | 2 | 0 | 0 |
| P2 | 0 | 3 | 2 | 4 |
| P3 | 2 | 5 | 0 | 2 |
| P4 | 2 | 0 | 0 | 1 |

|  |  |  |  |
| --- | --- | --- | --- |
| Available (safe sequence completed and request for A granted) | | | |
| A | B | C | D |
| 12 | 13 | 9 | 13 |



Safe Sequence: <P4, P3, P1, P2, P0>

Therefore, the request can be granted.

Q1 d) (ii) (2 marks).

**Answer:**

request(P1) = D(1)

Allocated(P1) + Request (P1) > Maximum (P1)

Therefore, the request cannot be granted as the request plus what has already been granted to P1 exceeds the maximum for P1.

**END OF QUESTION ONE**

### QUESTION TWO (total: 40 marks): Memory Management

**Q2 a) (4 marks).**

**Answer:**

This is because the run-time mapping from virtual to physical addresses is done using a hardware device which is the Memory Management Unit (MMU).

**Q2 b) (3 marks).**

**Answer:**

* Due to frequent access, it is better that the page is not paged out of main memory.
* The page is a pointer to other pages and is required for access.
* Pure demand paging execution: a page is not paged out of memory until it is required. Allows for execution prior to obtaining the pages it needs and produces a page fault for every page it needs.

**Q2 c) (3 marks).**

**Answer:**

False. Paging does indeed suffer from internal fragmentation, however, it increases when the page size INCREASES, as more memory would be allocated than the process needs.

**Q2 d) (Total: 6 marks).**

Q2 d)(i) (3 marks).

**Answer:**

|  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
|  | 7 | 8 | 2 | 3 | 8 | 1 | 8 | 7 | 8 | 7 | 8 | 3 | 0 | 9 | 0 | 1 | 2 | 8 | 1 | 2 |
| F1 | 7 |  |  |  |  | 1 |  |  |  |  |  |  | 0 |  | NF |  |  |  |  |  |
| F2 |  | 8 |  |  | NF |  | NF |  | NF |  | NF |  |  |  |  | 1 |  |  | NF |  |
| F3 |  |  | 2 |  |  |  |  | 7 |  | NF |  |  |  | 9 |  |  |  | 8 |  |  |
| F4 |  |  |  | 3 |  |  |  |  |  |  |  | NF |  |  |  |  | 2 |  |  | NF |

Final Frame:

F1 = 0

F2 = 1

F3 = 8

F4 = 2

Page Faults = 11

Q2 d)(ii) (3 marks).

**Answer:**

No, as page faults were frequent. Page faults = 11 with:

Final Frame:

F1 = 0

F2 = 1

F3 = 8

F4 = 2

**Q2 e) (4 marks).**

**Answer:**

EAT = 0.95 \* (20 + 100) + 0.05 \* (20 + 100 + 100) = 125 ns

**Q2 f) (6 marks).**

**Answer:**

2-level page table, virtual address that needs to be supported per level = 238/2 = 219 bytes.

Page size = 214 bytes, with each entry in the table being 4 bytes = 22 bytes.

Therefore, total page requirement = 214+22 < virtual address per level (219 bytes). Therefore, my friend’s design satisfies the requirements.

**Q2 g). (3 marks).**

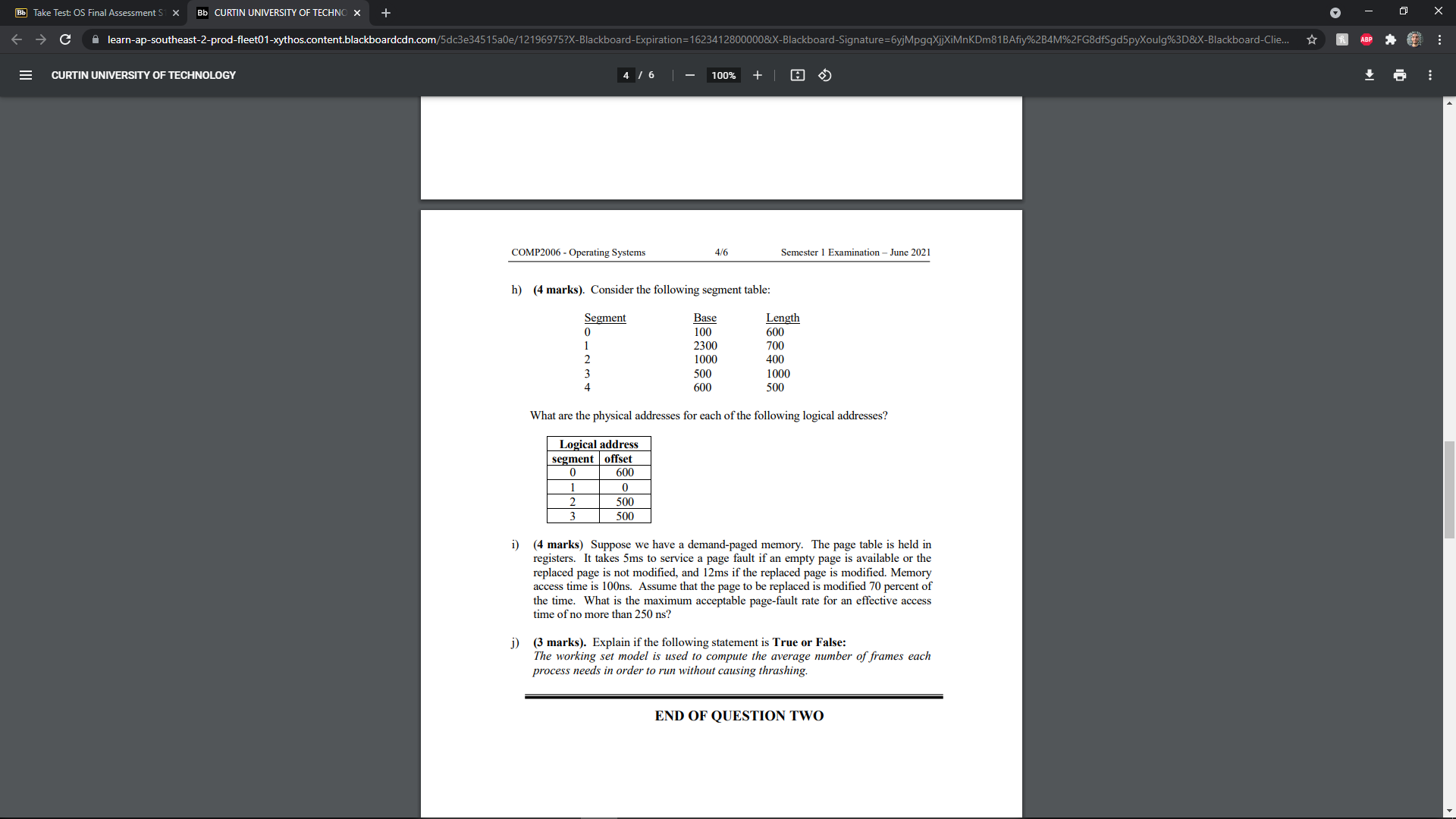
**Answer:**

False. In a FIFO page replacement algorithm, the more frames used the more page faults will be generated. Therefore, N frames will result in less page faults than N+1 and therefore perform better.

**Q2 h). (4 marks).**

**Answer:**

|  |
| --- |
| Physical Address |
| 700 |
| 2300 |
| 1000 |
| 1100 |



**Q2 i). (4 marks).**

**Answer:**

Let the page fault rate = ‘x’

Time taken by access in memory:

T1 = (1-x) \* (100ns) = (100 – 100x) ns

Time taken by page fault modified:

T2 = x \* (0.7) \*(100 + 12 \* 106) = 8400070x ns

Time taken by page fault not modified:

T3 = x \* 0.3 \* (100 + 5 \* 106) = 1500030x ns

EAT = 250 >= 100 – 100x + 8400070x + 1500030x

Therefore, the maximum page fault rate = 0.00001515 = 0.0015%

**Q2 j). (3 marks).**

**Answer:**

True. The working set model is used to ensure the demanded frames by processes is less than the total number of available frames. This ensures that thrashing does not occur. Each process is allocated enough frames. Should there be more frames present, the degree of multiprogramming is then increased, as long as the demanded pages is less than the available pages.

**END OF QUESTION TWO**

### QUESTION THREE (total: 28 marks): File, I/O, and disk

**Q3 a) (4 marks).**

**Answer:**

False. Reading from a memory block always consumes more time than writing. In a write operation, the communication is done one-way, meaning that there is data transfer from place to another. However, reading operations involve the instruction travelling to the disk, the data being read then data transfer from memory block A to the recipient. Therefore, since more back and forth communication is involved with reading from memory block A than writing to memory block A, the time taken by the read instruction is longer than the write instruction.

**Q3 b) (3 marks).**

**Answer:**

True. The producer-consumer problem has to do with multiple consumers attempting to access the same file and potentially modify it. Operating systems may have multiple process attempting to access and modify a file at the same time therefore, the producer-consumer problem is relevant to I/O operations in an operating system.

**Q3 c**) **(3 marks).**

**Answer:**

False. Reliability of RAID systems is due to the duplication of data. Raid systems that duplicate the available data strips improves reliability as failure of a certain disk stirp does not result in the loss of data.

**Q3 d**) **(3 marks).**

**Answer:**

True, since the internal file structure packs a number of logical records into physical blocks which causes internal fragmentation.

**Q3 e**) **(3 marks).**

**Answer:**

False.

The number of rotations per second (RPS) = 5400 / 60 = 90 RPS

Time per rotation = 1/90 = 0.011s

Average time per read = timer per rotation / 2 = 0.00556 s = **5.56 ms**

**Q3 f**) **(4 marks).**

**Answer:**

Number of cylinder moves = 12 + 100 + (100 – 22) = 190 cylinder moves

Therefore, seek time = 190 \* 5 = 950 ms = 0.95s

**Q3 g) (4 marks).**

**Answer:**

Size per block = block size in bytes – block number storage in bytes = 512 – 4 = 508 bytes

For 16 direct blocks, max file size = 16 \* 508 = 8128 bytes

For single indirect, max file size = 508 bytes

**Q3 h) (Total: 4 marks).**

Q3 h) (i) (2 marks).

**Answer:**

200 reads + 2 writes = 202 I/O operations

Q3 h) (ii) (2 marks).

**Answer:**

1 read (using FCB) + 1 write (using FCB) = 2 I/O Operations

**END OF QUESTION THREE**

**QUESTION FOUR (total: 12 marks): Protection and Security**

**Q4 a) (3 marks).**

**Answer:**

False. Since the association between the process and the domain is static, the set of resources available to the process will remain constant throughout its life time, therefore its allocated privileges will not change (not violating the principle of least privilege).

**Q4 b) (4 marks).**

**Answer:**

All users can print, since Anna can print, John can switch to Anna, student can switch to John who can switch to Anna and Staff can switch to Student who can switch to John who can switch to Anna.

**Q4 c) (5 marks).**

**Answer:**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
|  | fileA | fileB | fileC | fileD |
| anna | Read  Write | Read  Write  Execute |  |  |
| john | Read | Read  Execute | Read  Write | Read  Write |
| staff | Read | Read  Execute | Read  Write | Read |
| OS | Read | Read  Execute | Read  Write | Read |
| student | Read | Read  Execute |  |  |

**END OF EXAM PAPER**