Answers

Answer 1

8(a)	**1 mark** per bullet point **1 mark** for identifying the state, **max 2** for description M**ax 3** marks for each state ∞ Ready ∞ The process is not being executed ∞ The process is in the queue… ∞ … waiting for the processor's attention / time slice ∞ Running ∞ The process is being executed by the processor ∞ The process is currently using its allocated processor time / time slice ∞ Blocked ∞ The process is waiting for an event … ∞ … so it cannot be executed at the moment ∞ …e.g. input/output	**6**
8(b)	For up to 2 maximisation techniques for each of memory and disk Max 2 for Memory, Max 2 for disk if no descriptions are given **1 mark** for identification of maximisation technique**, 1 mark** for description, **1 mark** for further description or information about improvement to **max 4 for memory** **Memory** ∞ Moving frequently accessed instructions to cache (1) for faster recall (1) as SRAM is used rather than DRAM for cache (1) ∞ Making use of virtual memory (1) with paging or segmentation (1) to swap memory to and from a disk (1) ∞ Partitioning memory (1) dividing main memory into static/dynamic partitions (1) to allow for more than one program/task to be available //multiprogramming (1) ∞ Removing unused items/tasks from RAM (1) by marking a partition as available (1) as soon as the process using it has terminated (1) **1 mark** for identification of maximisation technique, **1 mark** for description, **1 mark** for further description or information about improvement to **max 4 for disk** **Disk** ∞ Disk caching (1) a disk cache holds data that is frequently transferred to/from the disk (1) the cache can be held on disk or in RAM (1) ∞ Compression utility (1) decreasing the size of a file stored on disk (1) in order fit more / larger files on the disk (1) ∞ Defragmentation utility (1) files are rearranged to occupy contiguous disk space (1) this reduces the time taken to access files// decreases latency (1)	**6**

Answer 2

6(a)	**1 mark** per bullet point Page: • Virtual Memory is divided into blocks of a fixed size Page frame: • the main memory is divided into page frames of the same size as a page Page table: • the Page (Map) table shows the mapping of pages to page frames	**3**
6(b)	**1 mark** per bullet point to **max 3** • To allow multiprogramming / multitasking to take place • To ensure fair usage of the processor • To ensure fair usage of peripherals • To ensure fair usage of memory • To ensure higher priority tasks are executed sooner • To ensure all processes have the opportunity to finish	**3**
6(c)	A **signal** from a software **source** or hardware device seeking the **attention of the processer**	**1**
6(d)	**1 mark** per bullet point in the order given • JOB32 • JOB42 • JOB42	**3**

Answer 3

4(a)	1 mark per bullet point to **max 2** • Disk / secondary storage is used to extend the RAM / memory available • … so CPU can access more memory space than available RAM • Only part of program / data in use needs to be in RAM • Data is swapped between RAM and disk	**2**
4(b)(i)	1 mark per bullet point to **max 4** • Divide memory / RAM into frames • Divide virtual memory into blocks of same size called pages • Frames / pages are a fixed size • Set up a page table to translate logical to physical addresses • Keep track of all free frames • Swap pages in memory with new pages from disk when needed	**4**
4(b)(ii)	First-in-first-out // least-recently-used page // least-used-page	**1**
4(b)(iii)	1 mark per bullet point to **max 2** • Pages are required back in RAM as soon as they are moved to disk • There is **continuous** swapping (of the same pages) • No useful processing happens // deadlock • … (because) pages that are in RAM and on disk are inter-dependent • … (nearly) all processing time is used for swapping pages	**2**

Answer 4

5(a)(i)	1 mark per bullet point: ∞ Running process is halted ∞ Process moves to blocked state	**2**
5(a)(ii)	1 mark per bullet point max 2: ∞ Running process is halted // another process has use of the processor ∞ Process moves to ready state ∞ … Until next time slice allocated	**2**
5(b)(i)	1 mark per bullet point: ∞ Current process no longer running // processor is available ∞ Process was at the head of the ready queue / / process has highest priority	**2**
5(b)(ii)	1 mark per bullet point: ∞ The only ∞ Required resource becomes available // event is complete	**2**
5(c)	1 mark per bullet point to max 3: ∞ to allow multiprogramming ∞ to give each process a fair share of the CPU time ∞ to allow all processes to complete in a reasonable amount of time ∞ to allow highest priority jobs to be executed first ∞ to keep the CPU busy all the time ∞ to service the largest possible number of jobs in a given amount of time ∞ to minimize the amount of time users must wait for their results ∞ to maximise the use of peripherals	**3**

Answer 5

3 **(a)** The 245th page frame from the start of memory
// the 245th page frame from some base address [1]

(b) Flash memory // magnetic disk // hard drive [1]

(c) **(i)** Time of entry (NOT time in memory) [1]

(ii)

Page	Presence Flag	Page frame address	Additional data
4	1	542	12:07:34:49

[1 +1 + 1]

(iii) Number of times the page has been accessed [1]

(iv)

Page	Presence Flag	Page frame address	Additional data
3	1	132	0

[1 +1 + 1]

Accept only zero for 'additional data'

(d) For example:
Longest resident: page in for lengthy period of time may be being accessed often [1]
… so not a good candidate for being removed [1]

Least used: a page just entered has a low least used value … [1]
so likely to be a candidate for immediately being swapped out [1]

Answer 6

6 **(a)** A program is the written code ("static") [1]
A process is the executing code ("dynamic") [1]

(b) **running, ready:**
when process is executing it is allocated a time slice (running state) // process is allocated time on processor [1]
when time slice completed process / interrupt occurs can no longer use processor even though it is capable of further processing (ready state) [1]

ready, running:
process is capable of using processor (ready state) [1]
OS allocates processor to process so that process can execute (running state) [1]

running, blocked:
process is executing (running state) when it needs to perform I / O operation [1]
placed in blocked state – until I / O operation completed [1]

(c) when I / O operation completed for process in blocked state [1]
process put in ready state [1]
OS decides which process to allocate to processor from the ready queue [1]

(d) **high-level scheduler:**
decides which processes are to be loaded from backing store [1]
into memory / ready queue [1]

Answer 7

6 **(a) blocked → ready:**
process is waiting for resource/ I/O operation to complete (blocked state) [1]
when I/O operation completed process goes into ready queue (ready state) [1]
running → ready:
when process is executing it is allocated a time slice (running state) // process is allocated time on processor [1]
when time slice completed/interrupt occurs process can no longer use processor even though it is capable of further processing (ready state) [1]

(b) to be in blocked state process must initiate some I/O operation [1]
to initiate operation process must be executing [1]
if process in ready state cannot be executing/must be in running state [1]

(c) (i) exit/termination/completion [1]

(ii) when the process has finished execution [1]

(d) low-level scheduler:
decides which of the processes in ready state [1]
should get use of processor/be put in running state [1]
based on position/priority [1]
invoked after interrupt/OS call [1]
[max. 2]