This page has been left blank intentionally.
Q1.

(a) What are the different states that a process may have in a multiprogramming system?
   Explain clearly the reasons for transitions between the states that you have mentioned.
   (5)

(b) Why is the timer interrupt important in a multiprogramming system? Explain clearly the sequence of events when the CPU receives a timer interrupt while executing a process.
   (5)

(c) In a number of early computers, an interrupt caused the register values to be stored in fixed locations associated with the given interrupt signal. Under what circumstances is this a practical technique? Explain why it is inconvenient in general.
   (5)

(d) While UNIX has two processor access modes, user and kernel, the VAX/VMS operating system had four access modes, as follows (from highest to least privilege):

   **kernel:** Executes the kernel of the operating system, which includes memory management, interrupt handling and I/O operations

   **executive:** Executes many of the operating systems service calls, including file management routines

   **supervisor:** Executes other operating systems services, such as responses to user commands

   **user:** Executes user programs, plus utilities such as compilers, editors, linkers and debuggers

   What are the advantages and disadvantages of providing four modes instead of two?
   (5)
Q2.

(a) Explain clearly the connection between multiprogramming and virtual memory in a computer system. Why is virtual memory necessary for implementing multiprogramming effectively?

(b) Why should the page size be a power of 2 in a computer that uses paging for memory management? Explain clearly how a logical address is translated to a physical address in a computer system that uses a two-level page table with the following details:

- Each address has 32 bits.
- The lower order 12 bits are used as the offset.
- The higher order 20 bits are divided into two parts of 10 bits each for accessing the two-level page tables.

What is the total number of pages possible in the virtual memory of this computer? What is the size of a page?

(c) What is the principle of referential locality? Explain how a cache memory improves the performance of a computer using this principle. Give an example of an application that may not benefit from caching.

(d) Explain what is meant by a page fault in an operating system that uses paging for implementing virtual memory.

What are the actions taken by the operating system when a page fault occurs?
Q3.

(a) What is the meaning of the term *critical section* in relation to concurrency? Explain how a *semaphore* is used to enforce mutual exclusion in a concurrent program.

(b) In the UNIX system, a process is blocked when it executes a system call for input or output. Moreover, all the pages of the process may be swapped out to make room for other processes in the physical memory.

Explain how this may create a deadlock situation. Explain how this deadlock is prevented in the UNIX system.

(c) Explain the following disk scheduling policies:
- Shortest Seek Time First (SSTF)
- SCAN
- C-SCAN

Explain how the SSTF policy may create starvation.

(d) Explain briefly how an operating system should enforce security for the following resources:
- main memory of the system
- file system