OCR A Level Computer Science

Your notes

2.1 Systems Software

Contents


Your notes

Operating Systems

Why do we need Operating Systems?

- Operating systems are essential for managing the hardware and software resources of a computer system
- They provide an interface between the user and the hardware, allowing the user to interact with the system and run applications

What are the main functions of an operating system?

- Operating systems such as Windows, MacOS, Android, Linux, iOS and Ubuntu have responsibilities that can be grouped into the following categories:
  - Resource management
  - File Management
  - Interrupt handling
  - Security
  - Providing a platform for software to run
  - Providing a user interface
  - Providing utilities

Resource Management

- Operating systems manage the computer's resources, including the CPU, memory, disk drives, and printers
- They allocate resources to specific tasks and ensure that they are used effectively
- e.g. when a user opens multiple applications simultaneously, the operating system decides:
  - How much memory to allocate to each application
  - When and for how long each gets to use the CPU
  - How to handle data being read from or written to the hard drive.

File Management

- Operating systems handle the storage, retrieval, and manipulation of data files

- When working with files, operating systems provide a GUI of the file system that allows a user to decide which directory a file should be saved in and what the file name will be


Your notes

Interrupt Handling

- Interrupt events require the immediate attention of the central processing unit
- In order to maintain the smooth running of the system, interrupts need to be handled and processed in a timely manner
- E.g. if a user clicks cancel on a file conversion process, a signal is sent from the mouse, interrupts the processor, and the operating system will trigger the cancellation routine

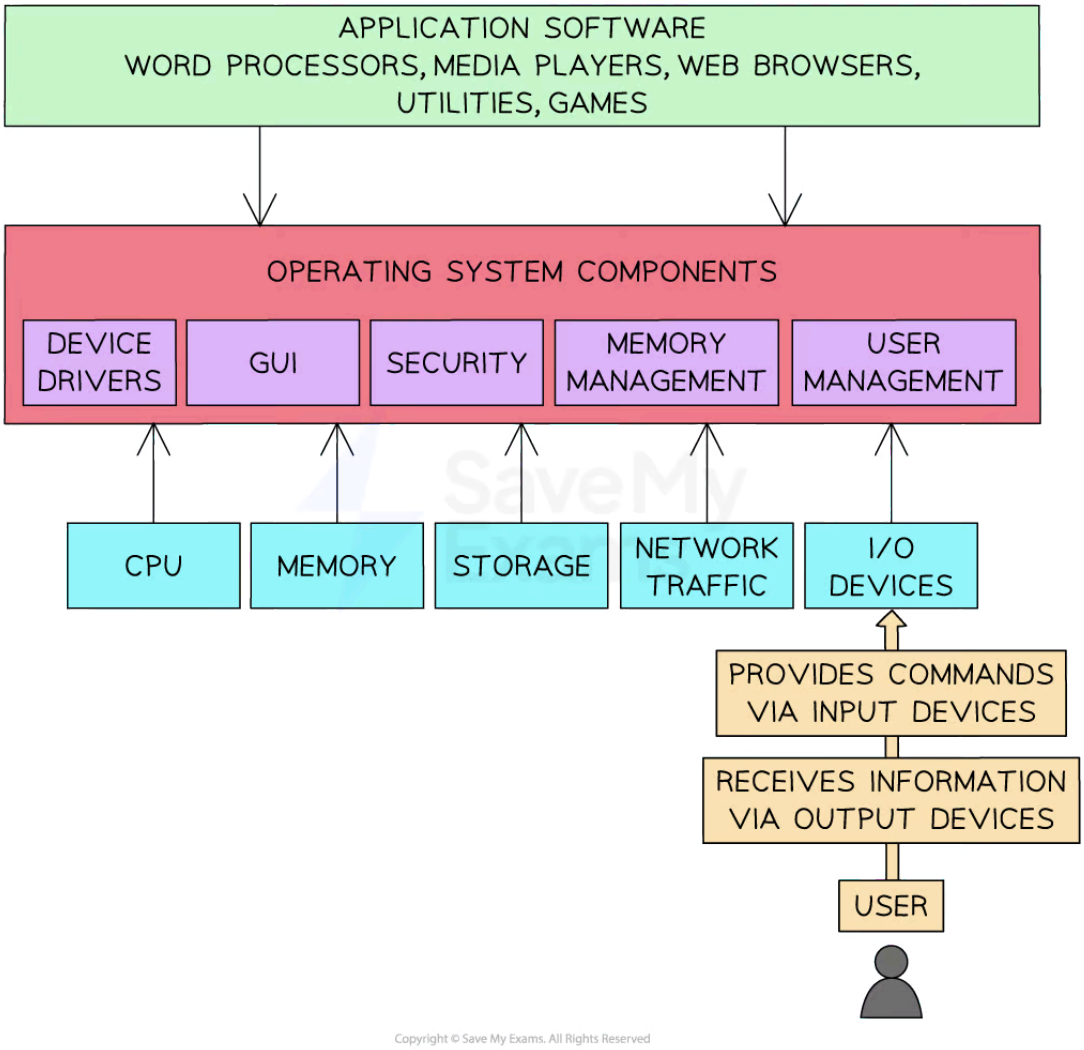
Security

- Operating systems provide various security features such as password-protected system accounts, a **firewall**, virus scanning and **file encryption**
- Password-protected system accounts are a very common feature in operating systems
- System accounts can also be restricted from performing certain actions, e.g. editing network settings, installing unapproved software, changing the account settings of other users

Providing a Platform for Software to Run

- Operating systems provide a platform on which application software can run, this is mainly by allowing software access to system resources
- e.g. if a computer game has intensive graphics and online play, the operating system will grant it access to the GPU and the network card

Your notes



Organisation of application layer, operating system components, and input/output

Providing a User Interface

- Operating systems provide interaction in 2 ways: visually through a graphical user interface (GUI) or text-based through a command-line interface (CLI)
- Most modern PC operating systems provide both options, and a user will prefer one over the other depending on the task
- Mobile operating systems such as Android and iOS provide GUIs that are suitable for interaction through touch

- Ubuntu is an OS popular with software engineers because it provides a no-frills GUI and an efficient CLI

Providing Utilities

Your notes

- Utility programs help with system maintenance and security
- Some utility programs include: file encryption, file compression, disk defragmentation, system backup, disk cleanup
- File encryption allows users to send files over networks securely
- File compression reduces the size of a file, which helps send large files over a network
- Disk defragmentation physically reorganises files on the hard disk so they can be found and accessed faster
- Disk cleanup scans the hard disk for duplicate and corrupt files and deletes them to create more space on the disk
- Backup software allows users to restore their system to a point in history

WORKED EXAMPLE

Operating systems usually come with utility software pre-installed.

Give two examples of utility software, explaining the purpose of both.

[4]

How to answer this question:

- You need to name two different examples of utility software and the purpose of each:
  - **Backup software:** creates system snapshots and will restore them on demand
  - **Disk defragmentation**: reorganises the hard disk so that files are better organised
  - **File compression**: reduces file size to consume less hard disk space or for faster transfer over a network
  - **Disk cleanup**: finds and removes duplicate or corrupted files, freeing up hard disk space
  - **File encryption**: Encodes files before transfer, will decode files that have been received
- For each utility, include why it's helpful in your answer

Answer:

Example answer that gets full marks:

1. Disk defragmentation is a utility that will better organise files on the hard disk so that the operating system can access them more efficiently. Better organised files will lead to a smoother operation of the system.
2. File encryption software will encode the contents of a file into a non-readable format. If files containing sensitive data are lost or stolen, the contents will be protected from misuse.

Acceptable answers you could have given instead:

1. Disk cleanup will scan the hard disk for duplicate files and remove them.
2. Backup software will allow users to restore a system to a previous point.


Your notes

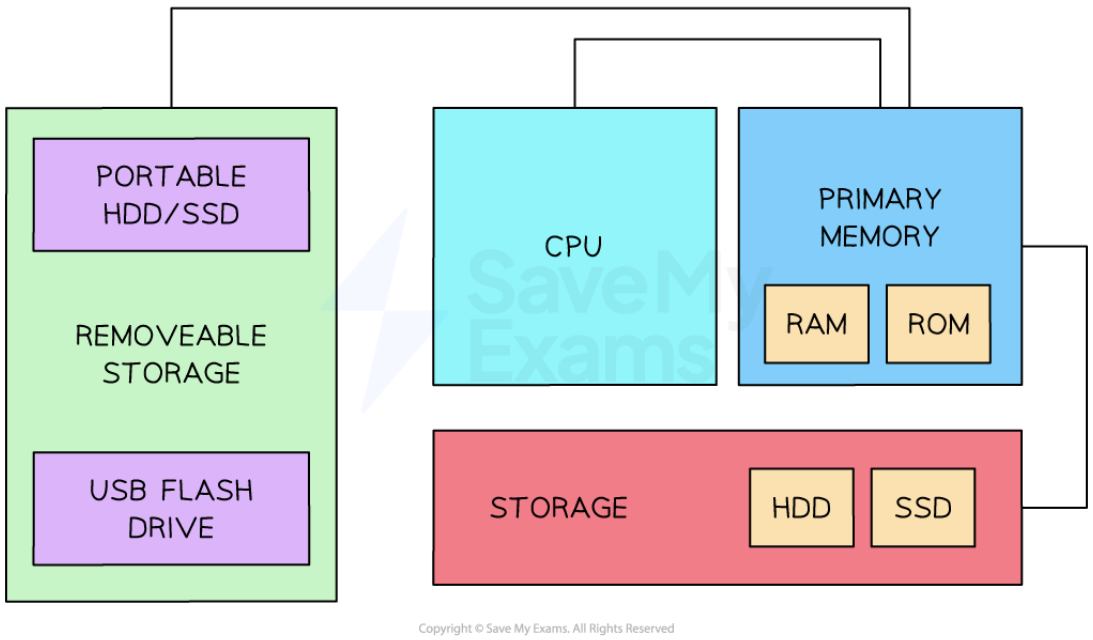

Your notes

Memory Management

What is Memory Management?

- Memory management is a **fundamental** role of the operating system, dealing with the **allocation** and **deallocation** of the computer's primary memory
- When a user opens an application, its **data is loaded from storage** into active memory so that it can run **smoothly**
- When a user opens a file from the file system, e.g. word document, the CPU loads this **file data,** as well as **application data,** into the primary memory
- Primary memory is a **limited resource** in the system, so it needs **careful management**
- Efficient allocation of memory enables a system to continue **multitasking**
- Memory management is made more efficient through 3 techniques:
  - Paging
  - Segmentation
  - Virtual Memory

Your notes



Links between different types of memory

EXAMINER TIP

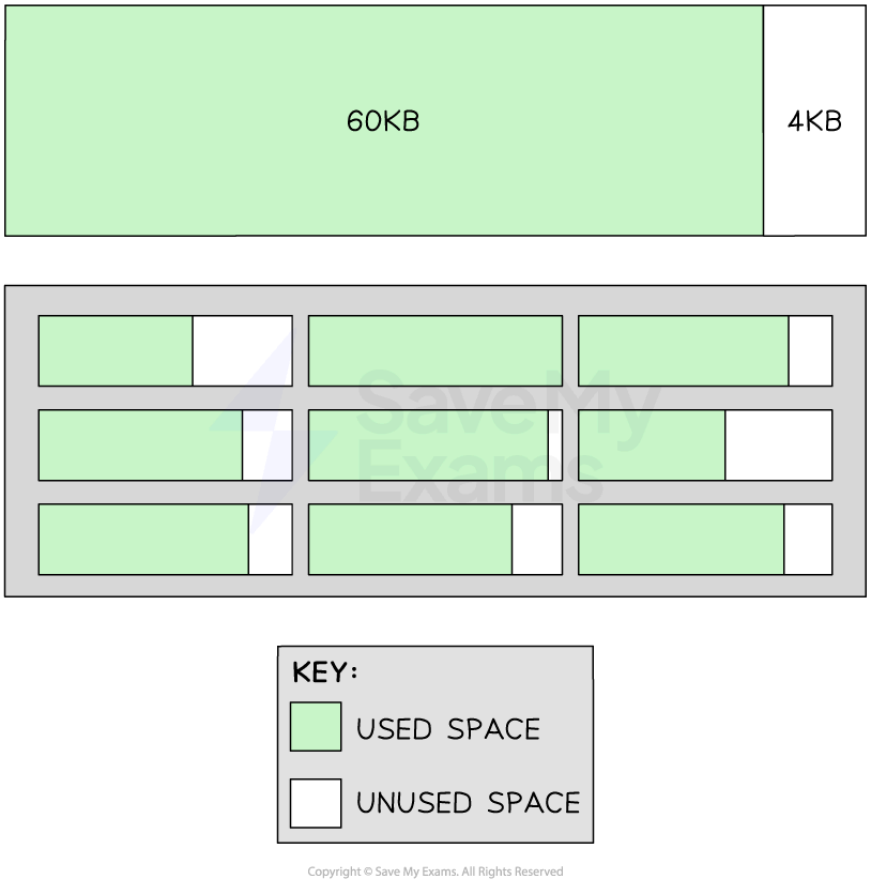
- The main benefit of memory management is to make computer systems run smoothly. Most users don't realise that as they effortlessly move between applications, it's made possible because the OS is rapidly reallocating memory depending on the task that the user is completing.
- Make sure you can name one benefit and one drawback for each memory management method in this revision note.

Paging

What is Paging?

- Data stored in memory will lead to the smooth running of applications
- Paging is a method of **chunking** the primary memory **into equal-sized blocks**
- When an application is launched, data will be moved from the hard disk into Pages for faster access
- As users move between applications, memory is dynamically allocated

- Pages will be taken away from applications not in active use and granted to applications that are in active use
- Paging can lead to **internal fragmentation**
- If a 200KB file is divided into four 64KB Pages, the last Page would have 8KB of unused space
- Unused space in a Page is wasteful because other unrelated data cannot be stored in this Page
- Over time, more pockets of wasted space will exist across the memory; this process is called **internal fragmentation**
- The image below shows a single 64KB Page with 4KB of unoccupied space
- The box below this shows many Pages, each with varying sizes of internal fragments



Internal fragmentation

Your notes

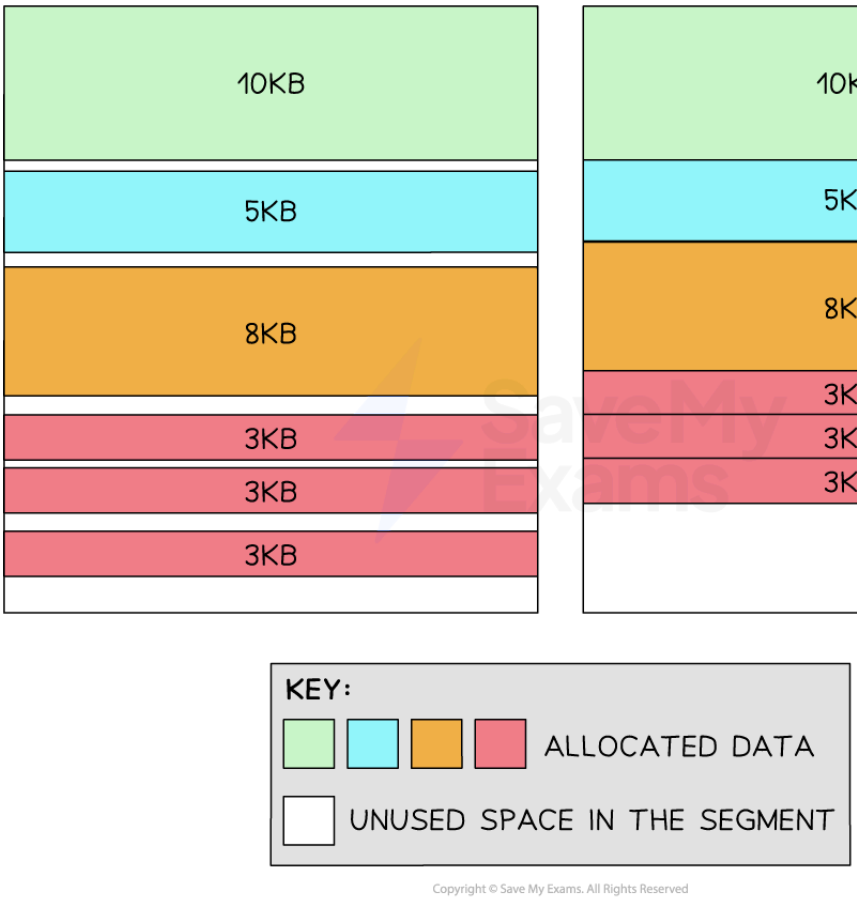
Segmentation

What is Segmentation?


Your notes

- Segmentation is a method of **chunking** memory into **blocks that correspond to different types of data** needed by an application
- A video editing application may have a Segment for video data, audio data and special effects
- Segments are **not all the same size**; they are sized depending on their allocated data
- Segmentation is space-efficient due to only allocating space depending on the amount an application needs
- Segmentation can lead to **external fragmentation**
- As Segments fill up the memory, physical gaps reduce the maximum size of new Segments that can be allocated
- Below (left) shows different application data assigned to a Segment
- The arrangement of data in the segment **becomes more fragmented over time** because as blocks are taken away it's not possible to guarantee a new block will occupy the same amount of space
- Below (right) shows a defragmented version of the Segment to highlight the total unused space

Your notes



External fragmentation

Virtual Memory

Virtual Memory

- If a computer is **running low** on primary memory, it can make secondary storage act as an 'extension' of the main memory
- The operating system can **offload data** from the primary memory into virtual memory
- Virtual memory creates **an illusion of a larger memory** and enables applications to continue to multitask
- However, accessing data in virtual memory is **considerably slower** compared to RAM
- **Solid-state drives** are faster than traditional hard-disk drives, but **neither are as fast** as RAM

- **Over-reliance** on virtual memory can lead to **performance issues**

Memory Management Technique	Description	Example	Benefits	Drawbacks
Paging	Divides memory into fixed-sized blocks (pages)	A process needing 200KB of memory is divided into four 64KB pages, leaving 8KB unused in the last page	Facilitates efficient memory management and enables the use of virtual memory	This can lead to internal fragmentation
Segmentation	Divides memory into variable-sized segments based on logical parts of a process	In a video editing application, different segments may be created for video data, audio data, effects, and UI elements	Allows for intuitive and efficient memory access	This can result in external fragmentation
Virtual Memory	Uses hard drive space as an 'extension' of RAM	When memory-intensive applications exceed the available RAM, the OS moves less frequently accessed pages to the hard disk	Allows more extensive programs to be run and facilitates effective multitasking	Slower to access than physical memory, which degrades performance if overused

Your notes

EXAMINER TIP

- Virtual memory is a method of memory management.
  - Not to be confused with Virtual Storage, which is storing files in the **cloud**.

WORKED EXAMPLE

Describe how the operating system would use virtual memory to load program C when there's not enough space in physical memory.

[3]

Answer:

Answer that gets full marks:

The operating system can use virtual memory to act as an extension of the computer's primary memory. This means less-critical data can be offloaded from the primary memory into virtual storage, useful when a higher-priority set of processes require immediate attention. If the OS offloaded data from the RAM into virtual memory, this would free up space for program C to be loaded into RAM.

Acceptable answers you could have given instead:

Program C can be loaded into the RAM if the operating system moves files and data into virtual memory. Virtual memory acts as an extension of the RAM.

Your notes

WORKED EXAMPLE

Imogen buys a desktop computer. It comes with an operating system installed.

Describe two ways that an operating system could manage physical memory.

[4]

How to answer this question

- Recall two methods of memory management (Paging, segmentation, virtual memory)
- Explain how each of these works and how they assist in memory management

Answer:

Example answer that gets full marks:

Imogen's operating system could manage physical memory in two ways. Paging, which is a method of dividing memory into fixed-size chunks known as pages. Application data can be allocated and deallocated to pages making it a flexible system depending on the activity of the user.

The system could also use virtual memory, a method of extending the available physical memory by using a portion of the hard drive. This allows more programs to run simultaneously and enhances the overall system performance.

Acceptable answers you could have given instead:

Imogen's computer uses paging to manage memory. This means it breaks down the memory into fixed-size pieces and swaps them in and out as needed.

The computer also uses something called virtual memory. It uses part of the hard drive to act like extra memory, so more programs can run at the same time, and everything works faster.

Your notes

System Interrupts

What are interrupts?

- An interrupt is a signal to the processor that **stops its current task** and performs a different task temporarily
- Interrupts can be hardware events or time-sensitive tasks
- When an interrupt occurs, the processor **suspends the current program** execution and transfers control to an **interrupt service routine**

Purpose and Role of Interrupts

- **Real-time Event Handling**: hardware errors and signals from input devices e.g. hard disk failure
- **Device Communication**: alerts from external devices e.g. printer jams and network errors
- **Multitasking**: suspending processing in one application so that the user can switch to another

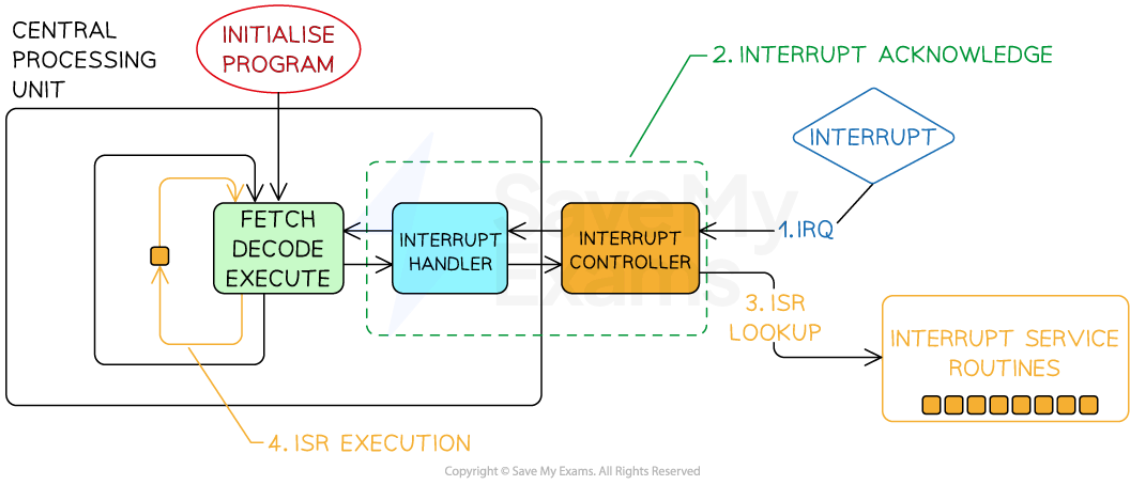
Types of Interrupts

Type	Definition	Example
Hardware Interrupts	Generated by external devices	Keyboard input, mouse movements, disk I/O requests
Software Interrupts	Triggered by software or the operating system	Application requests to open a file, **division by zero** errors
Trap Interrupts	Intentionally triggered by a program	Software debugging, handling unexpected error cases

The Interrupt Process

1. **Interrupt Request (IRQ)**
    - An external device or software generates an interrupt, signalling the processor to stop its current task
    - The interrupt controller passes this to the interrupt handler for assessment

2. **Interrupt Acknowledge**
    - The interrupt handler decides if the interrupt needs to be dealt with now or later
    - If yes, the **current contents of the processor registers are saved** in memory
3. **Interrupt Service Routine (ISR) Lookup**
    - The processor fetches the corresponding Interrupt Service Routine (ISR) associated with the interrupt type
4. **ISR Execution**
    - The processor transfers control to the ISR and executes the routine to handle the specific interrupt
5. **Interrupt Exit**
    - After the ISR completes, **the processor restores the content of the registers** from step 2
    - The fetch-decode-execute cycle is resumed



The interrupt process

What is an ISR?

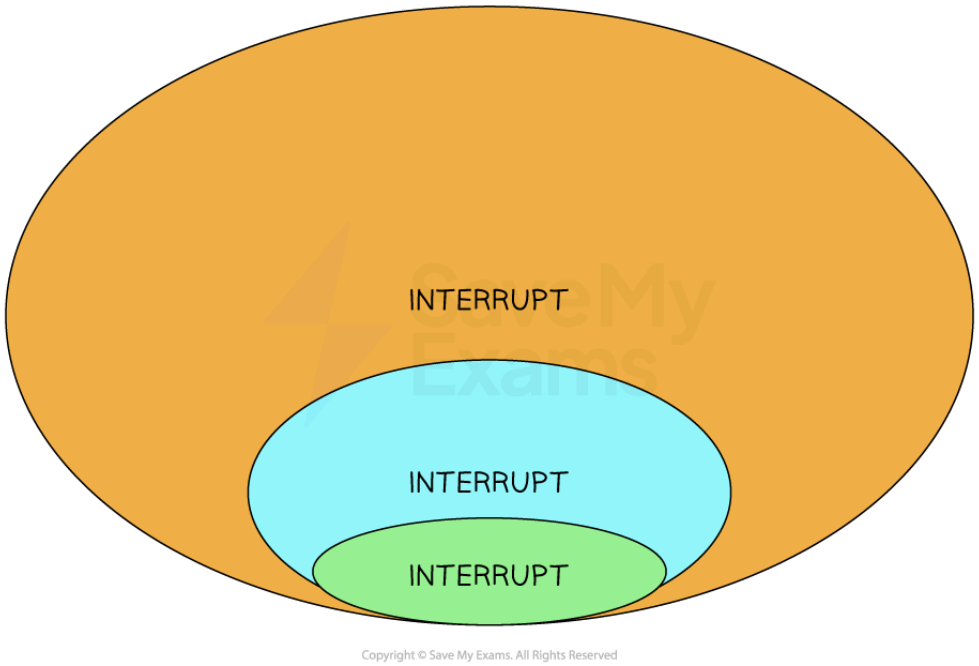
- An ISR is a **special function** that handles a particular interrupt type
- Each type of interrupt has a corresponding routine, e.g. printer jam, hard disk failure, file download error, network connection error all have routines to be followed when they happen
- ISRs should be **concise**, **efficient**, and carefully designed to minimise the time taken to execute, as they often need to handle time-sensitive events

Your notes

Interrupt priority and nesting

Your notes

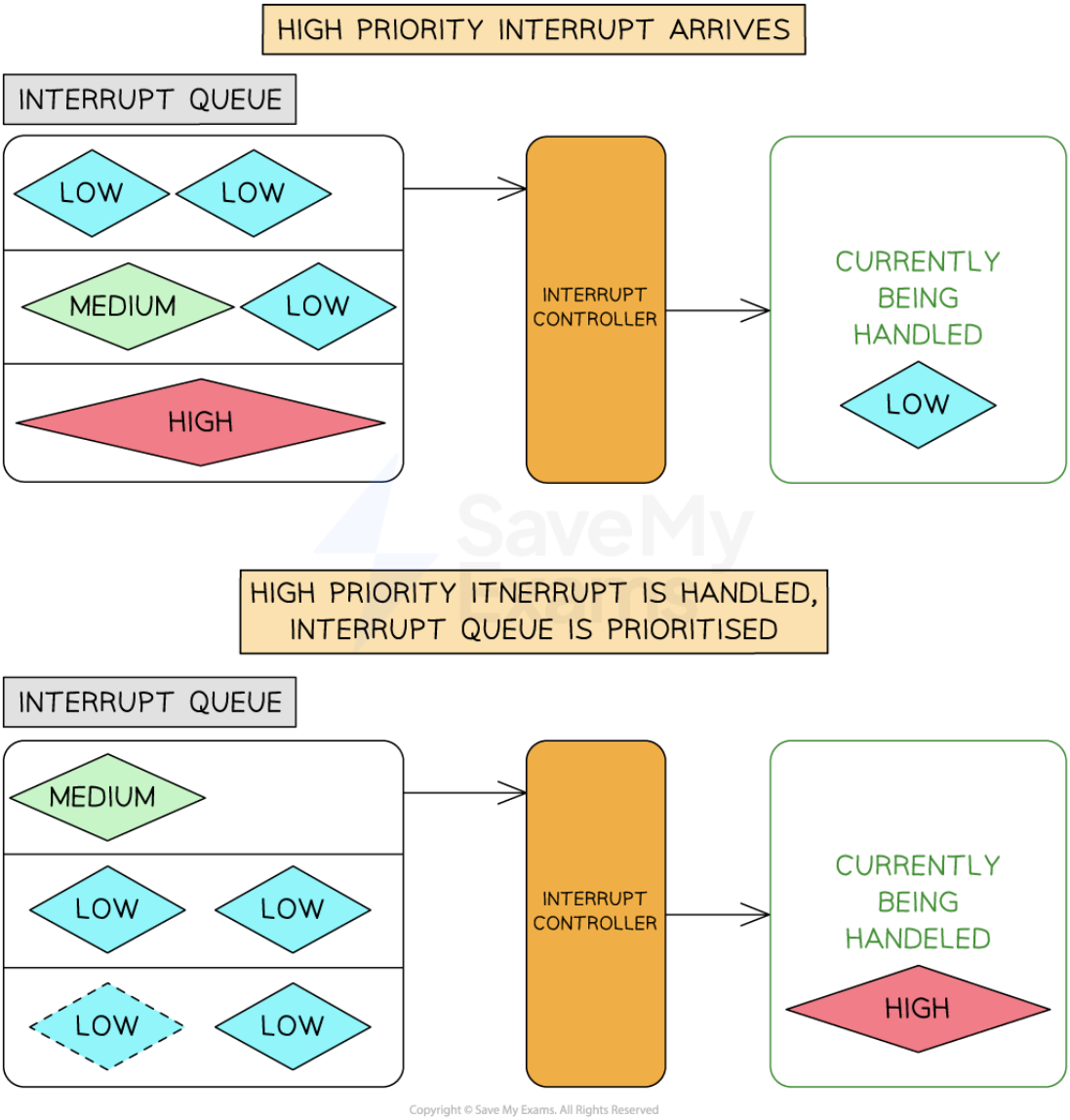
- Interrupt prioritisation means the processor can acknowledge and **switch to resolving a higher-priority interrupt**
- Prioritising interrupts is vital because many things can go wrong at the same time
- **Lower-priority ISRs may be temporarily suspended** until the higher-priority ISR completes the execution
- Nesting of interrupts refers to the ability of the processor to handle interrupts within interrupts
- Proper management of nested interrupts avoids potential conflicts and ensures system stability



Nesting of interrupts

Interrupt priority handling

Your notes



System handling of interrupt priority

Scheduling


Your notes

Scheduling

What is Scheduling?

- Deciding **which tasks** to process, for **how long**, and in **what order** is achieved through scheduling algorithms
- A CPU is responsible for processing tasks as fast as possible
- Different algorithms are used to prioritise and process tasks that need CPU time
- The algorithms have different uses, benefits and drawbacks.

Scheduling categories

- **Preemptive**: allocates the CPU for **time-limited slots**
- **Non-preemptive**: allocates the CPU to tasks for **unlimited time slots**

Preemptive Scheduling

- Allocates the CPU for a specific **time quantum** to a process
- Allows interruption of processes currently being handled
- It can result in low-priority processes being neglected if high-priority processes arrive frequently
- Example algorithms include **Round Robin** and **Shortest Remaining Time First**

Non-Preemptive Scheduling

- Once the CPU is allocated to a process, the process holds it until it completes its **burst time** or switches to a 'waiting' state
- A process cannot be interrupted unless it completes or its burst time is reached
- If a process with a long burst time is running, shorter processes will be neglected
- Example algorithms include **First Come First Serve** and **Shortest Job First**

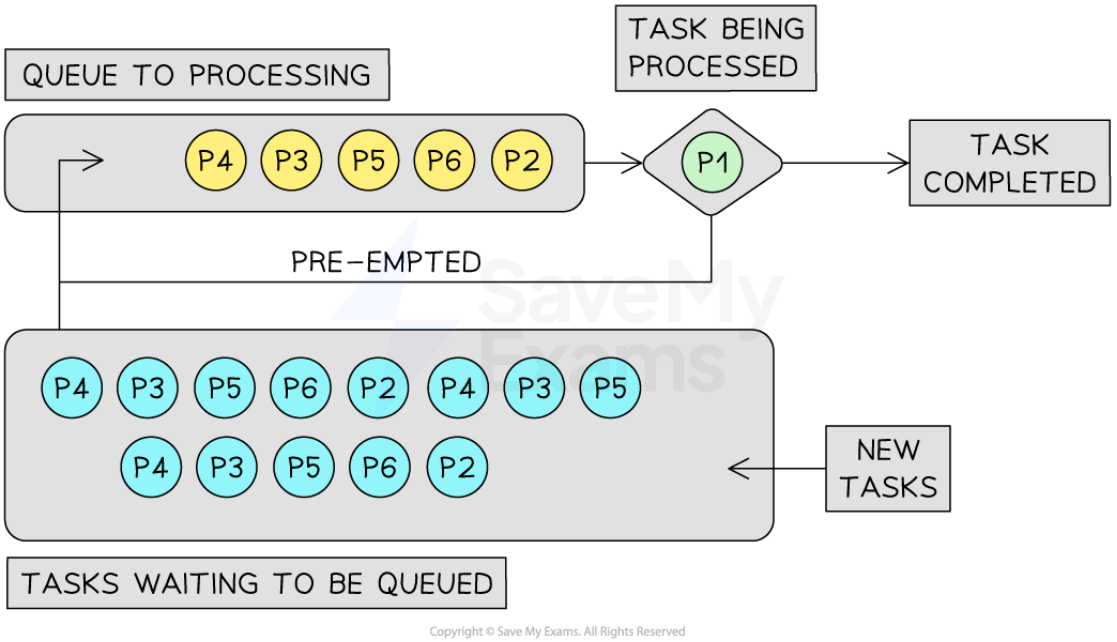
Scheduling Algorithms

Round robin (RR)

- RR is a pre-emptive algorithm, equally distributing processor time amongst all processes

- Each process is given a **time quantum** to execute
- Processes that are ready to be worked on get queued
- If a process hasn't been completed by the end of its time quantum, it will be moved to the back of the queue

Your notes

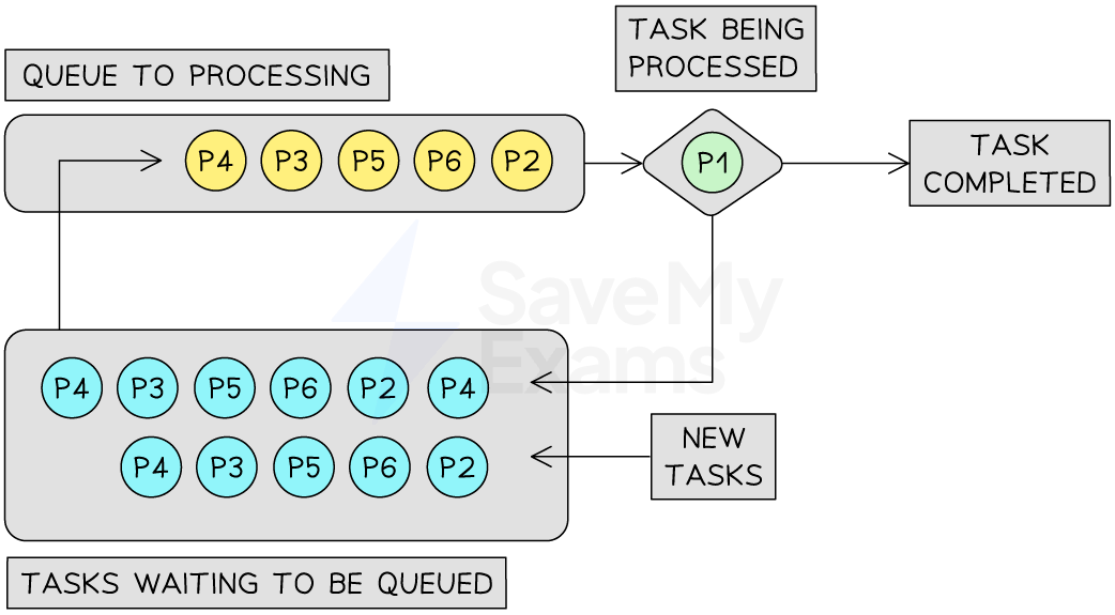


Round robin scheduling algorithm

First-Come-First-Served (FCFS)

- FCFS is non-preemptive, prioritising processes that arrive at the queue first
- The process currently being worked on will block all other processes until it is complete
- All new tasks join the back of the queue

Your notes

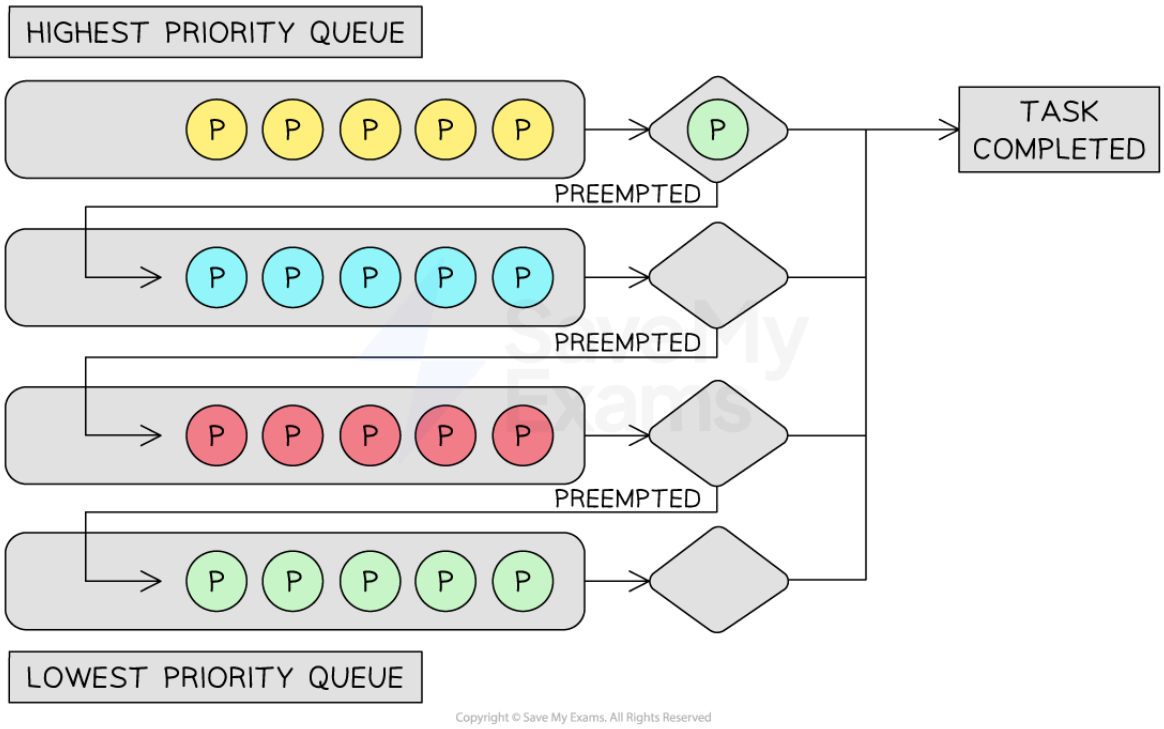


First-Come-First-Served scheduling algorithm

Multi-Level Feedback Queue (MLFQ)

- MLFQ is a pre-emptive priority algorithm where shorter and more critical tasks are processed first
- Multiple queues are used so that tasks of equal size are grouped together
- All processes will join the highest priority queue but will trickle down to lower priority queues if they exceed the time quantum

Your notes

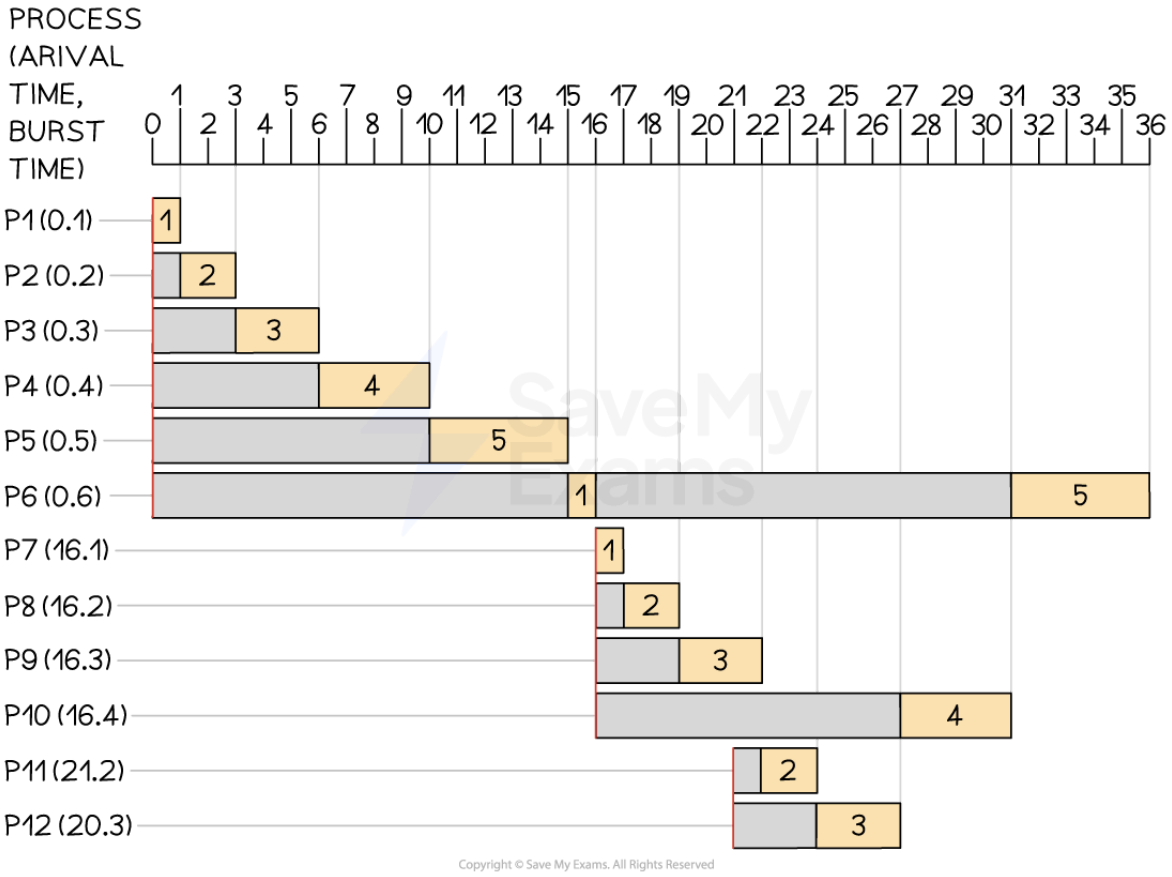


Multi-Level Feedback Queue scheduling algorithm

Shortest job first (SJF)

- SJF is non-preemptive, where all processes are continuously sorted by burst time from shortest to longest
- When new processes arrive on the queue, they are prioritised based on their burst time in the next cycle
- Shorter jobs are placed at the front of the priority queue
- Longer jobs have lower priority, so they are placed at the back

Your notes



Shortest job first scheduling algorithm

Shortest remaining time first (SRTF)

- SRTF is a preemptive version of SJF, where processes with the shortest remaining time are higher priority
- Time quantum is set, and if a task doesn't complete in time, it will be re-queued for further processing
- Before the next cycle starts, all processes are inspected and ordered by the shortest remaining time to complete

Your notes

CYCLE 1

	BURST TIME
P1	1
P2	2
P3	3
P4	4
P5	5
P6	6

CYCLE 2

NEW TASKS ARRIVING	BURST TIME
P6	3
P7	4

	BURST TIME
P3	1
P4	1
P5	1
P6	3
P7	4
P6	4

CYCLE 3

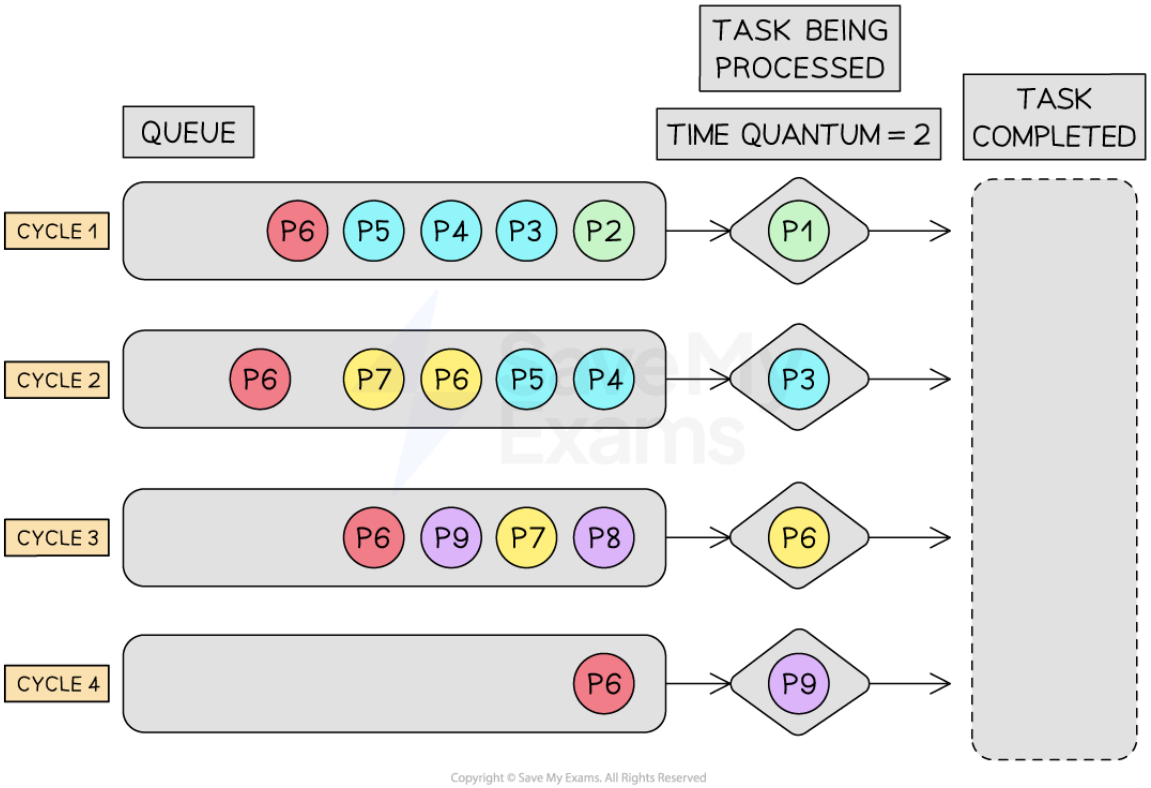
NEW TASKS ARRIVING	BURST TIME
P8	1
P9	3

	BURST TIME
P6	1
P8	1
P7	3
P9	4
P6	5

CYCLE 4

	BURST TIME
P9	1
P6	2

Your notes



Shortest remaining time first scheduling algorithm

Comparison and Summary of Scheduling Algorithms

Algorithm	Benefits	Drawbacks
Round Robin	All processes get a fair share of the CPU Good for time-sharing systems Predictable, as every process gets equal time	Choosing the right time quantum can be difficult This can lead to a high turnaround time and waiting time for long processes

First Come, First Served	Simple and easy to understand Fair in the sense that processes are served in the order they arrive	This can lead to poor performance if a long process arrives before shorter processes High-priority tasks wait for their turn in the queue
Multi-Level Feedback Queues	Smaller tasks are prioritised Creates a prioritisation system where similar-sized tasks are queued together	More complex than other algorithms Setting the correct parameters (e.g., number of queues, ageing rules) can be complex
Shortest Job First	Minimises waiting time Efficient and fast for short processes	Requires knowing the burst time of processes in advance Long processes can starve if short processes keep arriving
Shortest Time Remaining	Ideal for jobs that have shorter burst times It is preemptive, so it can be aligned with CPU for best performance (time quantum)	Like SJF, it requires knowing the burst time of processes in advance High context switching overhead due to preemption

Your notes

- The suitability of a scheduling algorithm largely depends on the specific scenario and the system requirements
- A drawback in one scenario may not be a drawback in another

WORKED EXAMPLE

A company makes anti-virus software. When running anti-virus software, an operating system uses a scheduling algorithm to allocate CPU time to the anti-virus software.

Explain why a First Come First Served scheduling algorithm would not be suitable in this situation.

[2]

How to answer this question:

- Think of the conditions that anti-virus software runs optimally
- Recall the way the FCFS algorithm works and its benefits and drawbacks
- Link how the optimal running of anti-virus is incompatible with FCFS scheduling

Answer:

Example answer that gets full marks:

Anti-virus software is high-priority because it scans the operating system constantly, looking for threats. When a threat is detected, anti-virus will quarantine or eliminate them. To work effectively, anti-virus software needs high-priority access to CPU time.

Using FCFS could delay these critical tasks if many other processes are in the queue ahead of the anti-virus software. Other less crucial tasks could get CPU time before the anti-virus process, leading to potential security risks.

Acceptable answers you could have given instead:

The FCFS algorithm is unsuitable because essential antivirus processing would be placed at the back of the queue and wait for its turn. Lower-priority tasks would use valuable CPU time, meaning the system could be at risk.


Your notes

Your notes

Types of Operating System

What is an OS?

- An operating system (OS) is a fundamental software that manages the computer hardware, provides common services for computer programs, and acts as an interface between users and the machine
- There are several different types of operating systems, each with unique characteristics and purposes
- Understanding these types is essential for both developing and using computer systems

List of operating system types

Type	Description
Distributed Operating Systems	Run on multiple machines appearing as a single unit, used for efficient task distribution and load balancing
Embedded Operating Systems	Designed for specific tasks, is the system running inside a device that is not primarily a computer system, e.g. microwave, dishwasher, washing machine
Multi-tasking Operating Systems	Allows multiple tasks to run concurrently on a single processor, manages system resources and allocates CPU time to different processes
Multi-user Operating Systems	Supports multiple users accessing computer resources concurrently, efficiently manages resource allocation, and provides features for data security and user privacy
Real-Time Operating Systems (RTOS)	Designed for immediate data processing, and can ensure tasks are processed in specific timeframes, highly efficient

Examples of each type

Type	Example
Distributed Operating Systems	Hadoop is an **open-source** OS designed to process big data using multiple **nodes** in a distributed network.

Embedded Operating Systems	**IoT** devices and many household devices contain embedded OS. These devices don't typically run a well-known OS. They run a proprietary OS that has a simple set of functions.
Multi-Tasking Operating Systems	Windows, MacOS, and Linux are multi-tasking OS that can run multiple applications simultaneously.
Multi-User Operating Systems	Windows, MacOS, and Linux are multi-user OSs where multiple users can log in and run independent processes.
Real-Time Operating Systems	Real-time OS are used in industries like aerospace and automotive where low **latency** is critical to safety.

Your notes

WORKED EXAMPLE

A taxi firm is investigating replacing its drivers with self-driving cars.

Explain why the self-driving system will use a real-time operating system.

[3]

How to answer this question:

- Be able to state the purpose of an operating system and recall the advantages of a real-time operating system
- Link the advantage to the scenario in the question

Answer:

Example answer that gets full marks:

A self-driving car system must process data in real-time from input sensors such as radar cameras. Data needs to be processed instantly so that the vehicle can operate safely. A vehicle that cannot process data quickly will be an unreliable danger to people. Faster processing of input data will lead to safer operation of the vehicle. A real-time system will replicate hazard perception, navigation, and vehicle control that humans can instinctively do.

Acceptable answers you could have given instead:

An operating system needs to process tasks quickly. A real-time operating system needs to process them instantly. A real-time operating system should be used in a self-driving car so that it can react to hazards quickly.

WORKED EXAMPLE

A company releases an in-home virtual assistant called 'Bertie Butler'. When placed in a room, the device listens out for the phrase "Hey Bertie". When someone says that phrase, it listens to the following question and tries to give a relevant answer.

The Bertie Butler device runs off an embedded operating system.

Define the term 'embedded operating system'.

[2]

How to answer this question:

- Recall the features of an embedded system
- Use the scenario to include examples in your answer

Answer:

Example answer that gets full marks:

Embedded systems are those running inside everyday items that are not primarily computers. Embedded systems usually have a simple set of features that they perform efficiently. Some other embedded systems, such as microwaves, have much simpler features and functions.

Acceptable answers you could have given instead:

An embedded system runs in most household devices that require some processing e.g. microwaves, dishwashers, electric toothbrushes


Your notes

Your notes

BIOS

What does BIOS mean?

- BIOS stands for Basic Input/Output System
- BIOS is a piece of **firmware** stored on a small memory chip on the motherboard
- On system start, the BIOS is the first software to run
- It performs a **POST (Power-On Self-Test)**, a diagnostic testing sequence that ensures all the hardware components are working properly
- If the BIOS encounters any errors during this test, it will either halt the boot process or issue an error message
- If the POST succeeds, the BIOS will run the **Bootstrap loading sequence**, which is the program responsible for starting the operating system

WORKED EXAMPLE

In the late 1990s, the CIH virus hit the headlines because it could overwrite and destroy the contents of a computer's BIOS.

Describe the effect of a computer having its BIOS overwritten.

[2]

How to answer this question:

- Explain the role of BIOS in a computer system
- Explain the consequence of having a destroyed BIOS

Answer:

Example answer that gets full marks:

The BIOS is crucial in initialising the hardware components and operating the system. If a virus, such as the CIH virus, overwrites or destroys the contents of a computer's BIOS, it will be unable to perform the Power-On Self-Test (POST) and fail to load the bootstrap program. This would mean the computer will not be able to initialise the operating system.

Acceptable answers you could have given instead:

If the BIOS has been overwritten, it will not be able to perform the Power-On Self-Test (POST) and, therefore, unable to load the bootstrap program. This would mean the operating system would fail

to start.

Your notes

Your notes

Device Drivers

What is a Device Driver?

- A device driver is a piece of software that enables communication between an operating system and specific hardware devices such as:
  - Printers
  - Graphics cards
  - Network cards
- Device drivers allow the OS to **control and interact** with those devices
- Because many external devices have embedded system software, a driver **bridges the gap** between a major operating system and a tiny hardware OS
- Device drivers make it possible to **perform specific operations** on the hardware e.g. a printer driver enables the OS to send print commands and manage print jobs
- Most hardware manufacturers write their own device driver software, meaning a single operating system may have **several printer drivers** installed

WORKED EXAMPLE

Adding a DVD drive to a computer often requires the installation of a piece of software called a device driver.

State the purpose of a device driver.

[1]

How to answer this question:

- Pick one example of a hardware device and describe what happens between it and the operating system
- Explain that this is made possible through device driver software

Answer:

Example answer that gets full marks:

The purpose of a device driver is to enable the operating system to interact with and control a hardware device. For example, a printer would have an associated device driver so the OS could send or receive data from a printer system.

Acceptable answers you could have given instead:

The purpose of a device driver is to enable communication between an operating system and external hardware.

Your notes

Virtual Machines

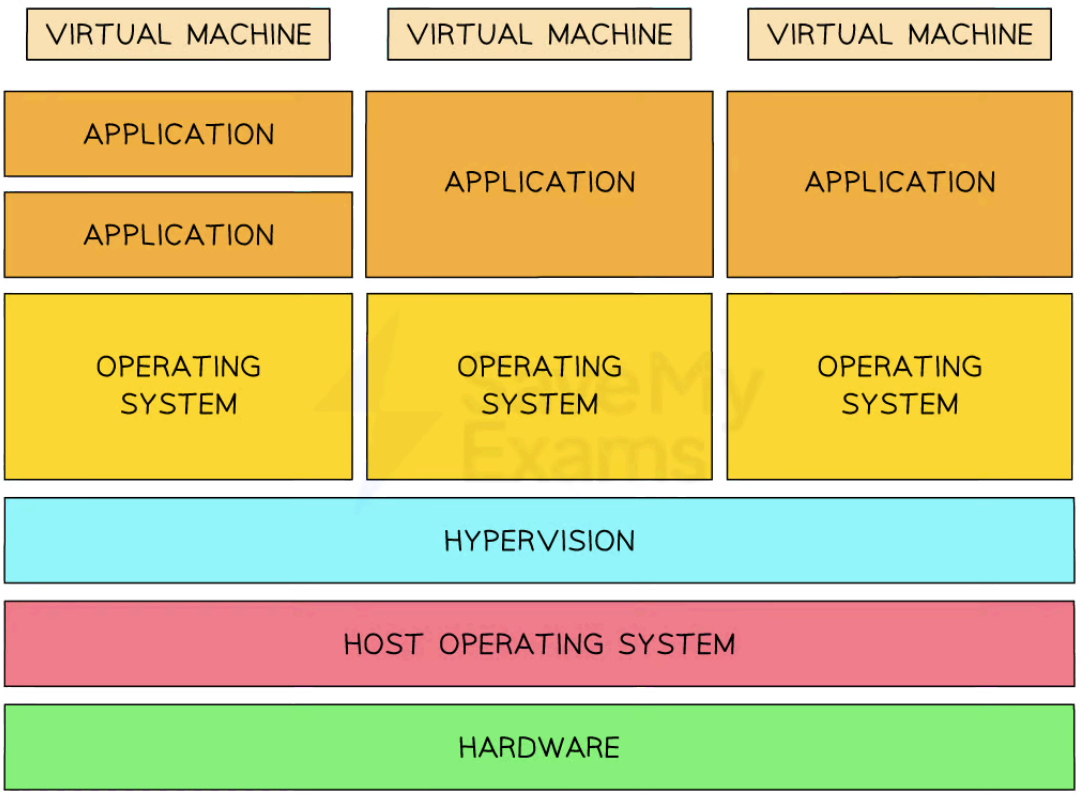
Your notes

Virtual Machines

What are virtual machines?

- Virtual machines (VMs) are entire operating systems **running inside another operating system**
- A user running Windows 11 could run a virtual machine of MacOS
- This would allow them to navigate the GUI of MacOS and install software on it
- Running a virtual machine helps access software that is only designed to run on specific operating systems
- VM management software includes a **Hypervisor** that monitors all activity happening inside the VM

Structure of several virtual machines running on a single piece of hardware



Cross-platform and forwards compatibility

- Not all software is designed to run on all operating systems
- Apple commonly makes software that **only runs on MacOS** for performance reasons
- A Windows user could run a virtual machine of MacOS and install the software they need
- Most software **needs to be updated** to work on the latest versions of operating systems
- A user running the latest release of Windows may need to run a virtual machine of a *previous* release of Windows to use an application that hasn't received a **forwards-compatibility update**


Your notes

In software testing

- VMs are a way to create **isolated test environments**, that leave the host operating system unaffected
- Isolated environments allow a developer to:
  - Monitor the way their software **affects system performance**
  - Test on a **clean-slate system**, while no other applications are running
- VM management software can create virtual machines that **act** like they have **older hardware**
- This allows developers to build software that can be run on older hardware so that **more users can use the software**
- A developer can test against **various operating systems**, such as MacOS, Linux and Windows, for **greater compatibility**
- This is achieved through compiling into intermediate code and running against different operating systems

Consequences

- VMs **share** the same system **hardware** as the **host** OS
- Over-use of VMs can **exhaust** the host OS of the system of CPU, hard disk and memory
  - VM software such as **VirtualBox** can set maximum limits on system resources
  - A **low-specification** machine could be configured to allocate only 1GB of memory and 20% of CPU
  - A **high-specification** machine could afford up to 8GB of memory and 50% of the CPU
- Operating systems are commonly **free to download**, but require an **activation payment** to access all features

Your notes

WORKED EXAMPLE

Explain why the programmers of anti-virus software may make use of virtual machines when developing the updates.

[3]

How to answer this question:

- Recall two benefits of using virtual machines in developing software
- Link these benefits to some considerations needed when developing anti-virus software

Answer:

Example answer that gets full marks:

Virtual machines (VMs) are essential in developing anti-virus software updates for several reasons. VMs create an isolated environment that is separate from the host operating system. This enables developers to safely work with virus test code that could risk the integrity of their own operating system. VMs can also be configured to emulate various types of hardware. This allows developers to understand how their anti-virus software will perform on different devices and under different conditions, ensuring a wider range of compatibility.

Acceptable answers you could have given instead:

Virtual machines create an isolated test environment from the host operating system. This means working with harmful test code carries less risk. Virtual machine management software can monitor the VM through the Hypervisor. This will show how the software affects VM system performance, allowing the programmer to make changes where needed.