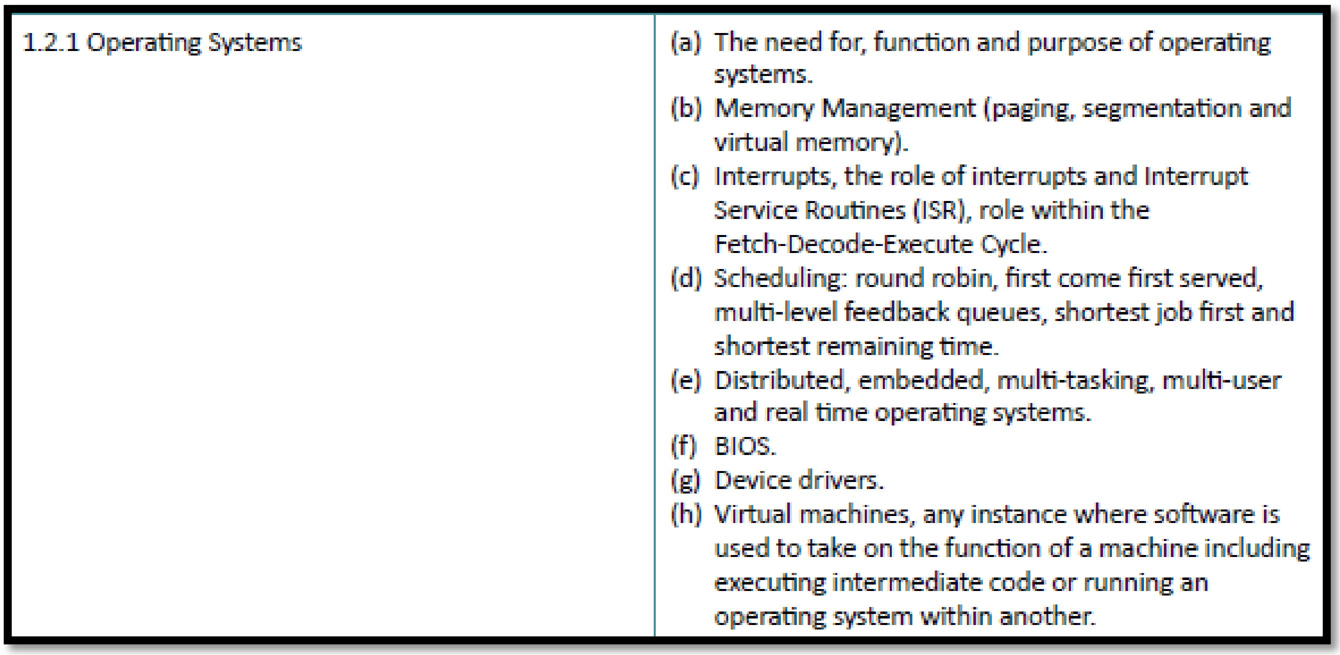
1.2.1 Operating Systems

Objectives



1.2.1 Operating Systems	(a) The need for, function and purpose of operating systems. (b) Memory Management (paging, segmentation and virtual memory). (c) Interrupts, the role of interrupts and Interrupt Service Routines (ISR), role within the Fetch-Decode-Execute Cycle. (d) Scheduling: round robin, first come first served, multi-level feedback queues, shortest job first and shortest remaining time. (e) Distributed, embedded, multi-tasking, multi-user and real time operating systems. (f) BIOS. (g) Device drivers. (h) Virtual machines, any instance where software is used to take on the function of a machine including executing intermediate code or running an operating system within another.

(a) The need for, function and purpose of operating systems

Purpose and features of an OS

The OS is responsible for managing the computer's resources. These include the storage devices, input and output devices, main memory and all software that is running on the computer. When the computer boots up, it will load the main part of the OS into RAM. This part of the OS is responsible for process management, hardware management and other key low-level tasks, and is known as the **kernel**. Once this has been loaded, the user interface part of the OS is loaded alongside key utility software.

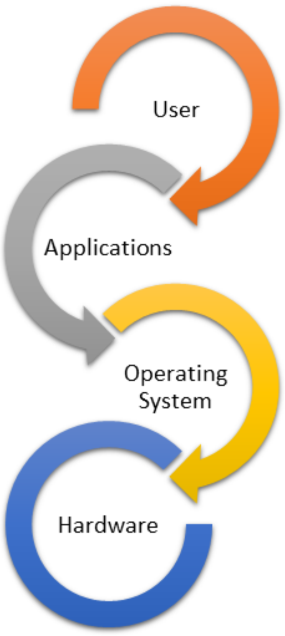
The OS is split into different layers and modules in order to perform its key responsibilities. Each key module is responsible for a different task under the OS remit. Here are the jobs that an OS carries out:

Managing the processor	Managing the memory	Handling external peripherals	Utility programs	Networking	Security	Providing a user interface
Deciding which process to execute next. Handling interruptions to the running process.	Loading programs. Reusing memory when programs close. Using virtual memory to compensate for a lack of RAM.	Dealing with I/O requests. Using device drivers to communicate with hardware.	File manager. Anti-virus. Defragmenter. Encryption. File compression. Installers. Clipboard manager. System monitor.	Interfacing to other computers via cable and WiFi.	Handling log-in with username and password. Preventing access to unauthorised files.	Windows, icons, menus, pointers. Touch gestures.

<u>TASK 1</u> Can you come up with a mnemonic to help you remember these 7 areas?

Where the Operating System fits in

1. Describe what this illustration shows:



The operating system is a layer of software between the applications and the hardware. It manages open applications, providing a user interface, multi-tasking, security and memory management. It provides a mechanism for the applications to use the hardware, sending outputs to the device drivers, and receiving input from the devices. The complexity of storing, retrieving, inputting and outputting to the hardware is transparent to the user.

2. Does the user ever interact with the operating system?

The operating system provides a user interface, but the user is interacting with applications and utilities.

(b) Memory Management (paging, segmenting and virtual memory)

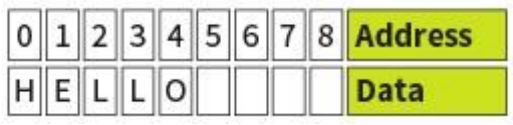
Memory management is the process of controlling and coordinating computer memory, assigning portions of memory to various running programs.

<u>TASK 2</u> Fill in the Gaps

address   operating system   save   byte   RAM   load

Memory addressing

In order to understand memory management, you need to understand the basics of memory addressing, which is how an individual byte of data can be found in ____________. Instructions and data are all stored in the same memory space. Each byte in memory is numbered, starting from 0, and this number is referred to as an _____________. In the table below, the text 'hello' is stored, with each letter having its own individual memory address. Memory addresses are essentially the position of a specific __________ of data in RAM.



0	1	2	3	4	5	6	7	8	Address
H	E	L	L	O					Data

In the above example, the text starts at address 0 and ends at address 4, using 5 bytes to store the whole word. If a computer has 128 MB of memory, it has a memory address range of 0–134 217 728. This address range is calculated by converting the megabytes to bytes, or 128 × 1024 × 1024 = 134 217 728 bytes. Most modern computing devices have 2 GB, 4 GB or even more.

When we talk about addresses, we are normally talking about where to find data in RAM. It is not possible on a modern computer to hard code memory addresses, as you have no way of knowing beforehand where data will be stored by the ___________________. The memory manager has the job of allocating memory addresses when a process wants to __________ or _________ data.

Segmentation

Segmentation is a method of allocating memory to processes, where segments are $\boxed{\textbf{variable sized}}$. Thus they can logically split RAM into segments of the exact size requested rather than forcing data into fixed sized chunks. This is ideal when storing blocks of code, such as a library. Each segment has a length and set of permissions (for example, read, write, execute) associated with it. A process is only allowed to make a reference into a segment if the type of reference is allowed by the permissions (e.g. if a section of code in a segment is read only, then the process would not be able to overwrite it)

One major advantage of segmentation is that memory can be shared between processes. If a block of code used by two processes is needed, it can be loaded into a segment. The other advantage is that because each segment has its own permissions, it allows the computer to have memory protection (ie stops a process writing to memory that it isn't allowed to). This prevents a bug or malware within a process from affecting other processes, or the operating system itself.

Paging

When the memory manager uses paging, it allocates $\boxed{\textbf{fixed}}$ sized blocks of memory to processes. These blocks of memory are known as pages. Pages are physical divisions because they are made to fit a certain size of memory. A transfer of pages between main memory and a secondary store, such as a hard disk drive, is referred to as paging or swapping. This is used in Virtual Memory (see later section).

<u>TASK 3</u> In what ways are paging and segmentation similar and different?

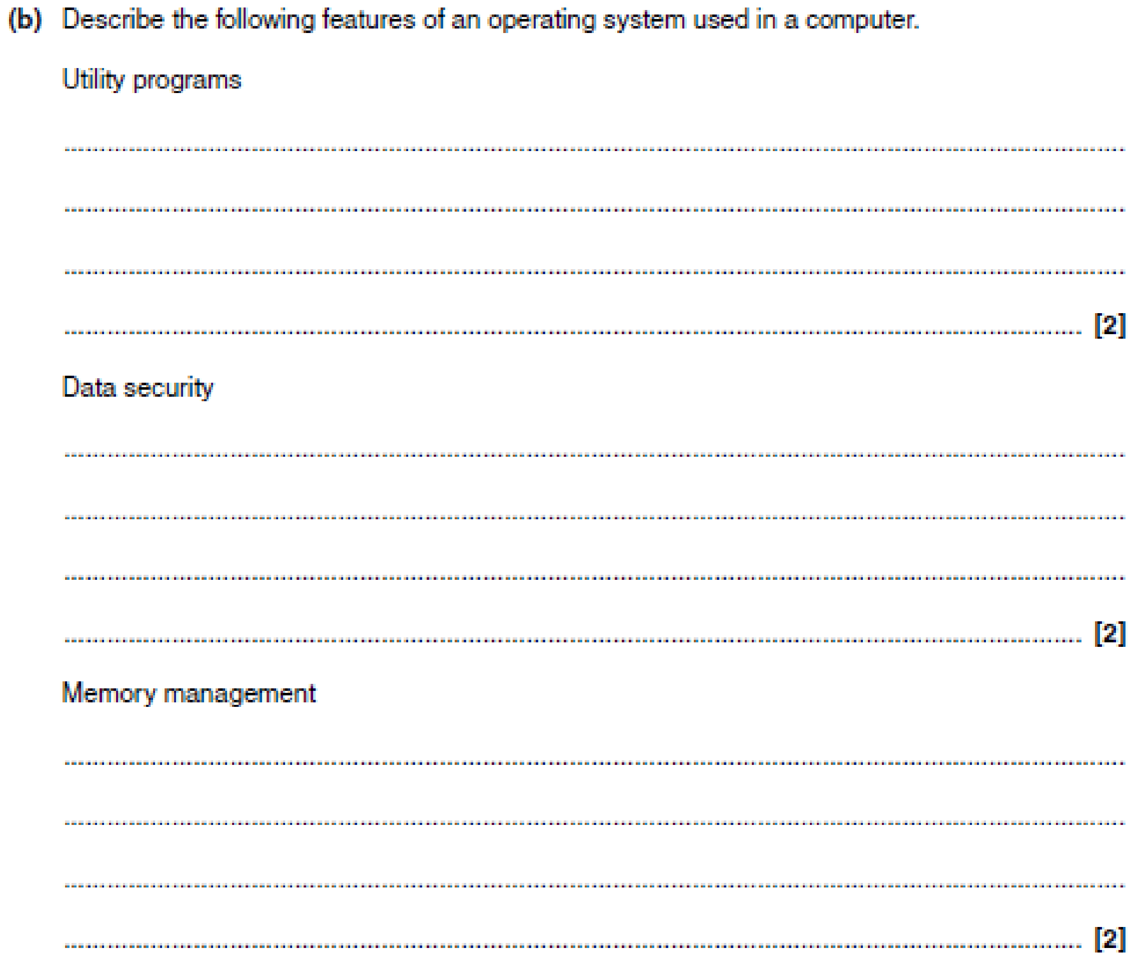
Virtual memory

Virtual memory is used when the size of the program exceeds the amount of physical memory available for it. The operating system keeps those parts of the program currently in use in main memory, and the rest on the disk. RAM is much faster than secondary storage and when the operating system is forced to use secondary storage instead of RAM, the system will appear to run slower. This process of transferring data from RAM to a storage device (hard disk) is done through paging/swapping.

TASK 4 – What is meant by disc thrashing? What causes it and what effect does it have?

<u>TASK 5</u> PAST PAPER Questions

January 2013, Question 1b



(b) Describe the following features of an operating system used in a computer.

Utility programs

....

....

....

.... **[2]**

Data security

....

....

....

.... **[2]**

Memory management

....

....

....

.... **[2]**

June 2010 Question 1a

(a) One feature of an operating system is memory management.

(i) State **two** reasons why memory management is necessary.

....

....

....

....

....

.... [2]

(ii) State why virtual memory may be needed.

....

.... [1]

(iii) Describe how virtual memory is used.

....

....

....

....

....

.... [3]

(iv) Describe the problem of disk threshing.

....

....

....

....

....

.... [3]

Specimen Paper Questions 1c

(c) An operating system may use segmentation or paging.

(i) Describe **two** ways in which segmentation and paging are similar.

....

....

....**[2]**

(ii) Describe **two** ways in which segmentation and paging are different.

....

....

....

....**[2]**

Answers

January 2013, Question 1b

Two from each: Utility • Maintenance program/carries out housekeeping • Eg, virus checker. Data security • Use of passwords • Access rights • Encryption. Memory management • Partitioning • Protect processes from each other/allocates memory • Virtual memory/paging/segmentation.	6	Accept hardware drivers Accept 'hiding data', 'restricting data' as a description of access rights Accept reference to 'automatic back-ups'

June 2010 Question 1a

1	(a)	(i)	• to allocate memory to allow separate processes to run at the same time • to deal with allocation when paging/segmentation • to reallocate memory when necessary • to protect processes/data from each other • to protect the operating system/provide security • to enable memory to be shared	[Max 2]
1	(a)	(ii)	to allow programs to run that need more memory than is available	[1]
1	(a)	(iii)	• use of backing store as if it were main memory/temporary storage • paging/fixed size units • swap pages between memory & backing store… • …to make space for pages needed	[Max 3]
1	(a)	(iv)	• occurs when using virtual memory/moving pages between memory & disk • disk is relatively slow • high rate of disk access • more time spent transferring pages than on processing	[Max 3]

1(c)
1(c)(i)
	An operating system may use segmentation or paging.
	Describe <u>two</u> ways in which segmentation and paging are similar.
	• Allow programs to run despite insufficient memory
	• segments and pages are stored on disk
	• segments and pages are assigned to memory when needed.
	[1 per bullet, max 2]
		[2]

1(c)(ii)	**Describe <u>two</u> ways in which segmentation and paging are different.** • Segments are different sizes but pages are fixed size • segments are complete sections of programs, but pages are made to fit sections of memory • segments are logical divisions, pages are physical divisions. [1 per bullet, max 2]	**[2]**

(c) Interrupts, the role of interrupts and Interrupt Service Routines (ISR), role within the Fetch-Decode-Execute Cycle.

TASK 6 In an OS, what is meant by *polling*? What are the problems with it?

Definition - An interrupt is

Why are interrupts used in a computer system?

- ✓ A more efficient alternative to polling.
- ✓ For a peripheral or routine to raise attention of CPU.
- ✓ To suspend the currently executing process.
- ✓ To carry out a task with a higher priority.

Interrupts

The OS decides when a process will run and for how long. However, it also needs to respond to input devices and other system-wide events, for example when the hard drive is ready to serve another request or the user has pressed a key on the keyboard. The OS needs to be made aware of these events in order to decide which course of action to take. These signals, sent from hardware to the CPU, are known as interrupts. Software can also generate interrupts during normal execution.

TASK 7

List as many examples of interrupts as you can:

TASK 8

An **arithmetic overflow error** can cause an interrupt. What is meant by this?

Example of an interrupt in action Imagine a user wishes to save a 30 MB file onto the hard drive. In order to do this, the file will need to be stored in small chunks of 1 MB. Using the buffer and **interrupt** system, the first chunk is sent to the hard drive, which, in turn, writes it to the disk. While this is occurring, the CPU will continue doing other tasks, such as dealing with keyboard interrupts or playing music.

Once the hard drive has finished saving the file, it sends an interrupt to the CPU requesting more data, which is duly sent by the CPU. This cycle will continue until the file has been saved; all 30MBs. It is important to remember that the CPU is much faster than any other part of the computer, including the user. If the CPU waited for the hard drive to finish saving before it did anything else (i.e. was synchronous), a lot of CPU cycles would be wasted. Interrupts allow the CPU to make much better use of time.

TASK 9 What is a buffer? How is it used in a computer?

Include examples of where they are used. Do your own research, you could start here http://www.webopedia.com/TERM/B/buffer.html

TASK 10 What happens if more than one interrupt happens at the same time?

Recording interrupts

When an interrupt occurs, a signal is sent to the CPU via a control bus or, in the case of a software interrupt, a flag is set. A special register, known as the interrupt register, is then updated by flipping a bit to 1. Each bit in the interrupt register represents a different interrupt and is referred to as a flag. When a flag is set to 1, it represents an interrupt that needs to be processed. Initially the interrupt register will have all flags set to 0.

Sometimes more than one interrupt can be sent to the CPU at the same time, or before the CPU has had a chance to deal with the previous interrupt. This is handled by simply setting multiple flags, but only if they are for two different interrupts. If more than one of the same interrupt is generated before the CPU has had a chance to deal with it, the second and subsequent interrupts are ignored. This is a fairly common occurrence for low-priority interrupts such as I/O) tasks or timers.

Each interrupt will be assigned a priority depending on how critical it is to the system. For example, keyboard interrupts can be safely postponed (have you ever typed on a computer only to wait a few seconds before text appeared?), while interrupts such as hardware switches must be dealt with immediately.

Interrupt Service Routines

Each interrupt has a corresponding Interrupt service routine (ISR). The interrupt service routine is a set of instructions that have to be carried out when the event occurs.

This website gives an overview of what happens when a mouse moves:

http://www.teach-ict.com/as_as_computing/ocr/H447/F453/3_3_1/interrupts/miniweb/pg5.htm

Interrupts and their role in the Fetch-Decode-Execute cycle

TASK 11 – Fill in the gaps

An interrupt can occur any time during program execution. Whenever it is caused, a series of events take place so that the instruction fetch execute cycle can again resume after the OS calls the routine to handle the interrupt.

Checking for the interrupt:

When each cycle of the Fetch/Decode/Execute has been completed, the Interrupt register is checked to see if an interrupt has been generated.

If it has, the following steps are performed by the OS:

1. _______________ the execution of current instruction
2. Push the address of _________________________ on to the ______________
3. Save all the other registers on to the stack too.
4. Load the PC with the _________________________________
5. This starts the Fetch Execute cycle again for the interrupt instruction.

Once the OS completes the execution of the interrupt, the following happens:

- Check for _________________________ and action if necessary
- The ___________________________________ to be executed is obtained from popping the value of the address in the stack back on to the PC.
- The contents of all the ____________ are restored from the stack.
- The suspended instruction can now continue/resume processing.

TASK 12 - Interrupts - PAST PAPER Questions.

June 2013 Question 1

(a) In a computer system explain what is meant by an interrupt.

....

....

....

.... **[2]**

(b) One example of an interrupt is a user interrupt.

State **two** other types of interrupt.

1 ....

2 ....
[2]

(c) Explain the need for interrupts to have priorities.

....

....

....

.... **[2]**

January 2012 Question 1

1 (a) An interrupt may occur during processing.

(i) State the purpose of an interrupt and give **one** example of a high priority interrupt.

....

....

....

.... [2]

(ii) Describe the use of a data structure while interrupts are being serviced.

....

....

....

....

....

....

....

.... [4]

(iii) Explain how an interrupt is detected during the fetch-execute cycle.

....

....

....

.... [2]

(b) After most interrupts, normal processing is resumed.

(i) Give **one** example of an interrupt after which processing is resumed.

....

.... [1]

(ii) On completion of processing an interrupt, state the steps that need to be taken before resuming the processing of the original job.

....

....

....

.... [2]

Past Paper Answers

June 2013 Question 1

1	(a)		Two from: • Interrupt/Signal sent to processor • Request for processing time • Allows important tasks to be processed/to take precedence	2	Not message
	(b)		Two from: • Power (failure) • Peripheral/I/O interrupt • Clock interrupt • Software interrupt.	2	Not hardware unless explanation or example Not timer
	(c)		Two from: • To decide between interrupt & current task • To choose which interrupt to process if 2 (or more) occur together • To ensure most urgent task is performed first • To ensure most efficient use of processor.	2	

Jan 2012 Question 1

Question			Answer	Marks	Guidance
1	(a)	(i)	• to alert the processor that a task needs attention/request processing time • power failure	2	accept "causes a break in execution" Marks are independent Allow hardware failure/clock as examples
		(ii)	• stack • LIFO • to store the contents of registers • to return values to registers… • …in order to resume processing	4	
		(iii)	• Interrupt register is checked… • when each cycle completed • by comparing priority of the current task with interrupt register	2	
	(b)	(i)	• eg printer out of paper	1	accept any valid example
		(ii)	• reset flag(s) to inactive state • check for further interrupts… • … & service them if necessary • restore contents of registers (from stack)	2	

(d) Scheduling: round robin, first come first served, multi-level feedback queues, shortest job first and shortest remaining time.