Functions of Operating Systems

What the Specification Says

Describe the main features of operating systems, for example memory management, and scheduling algorithms;
Explain how interrupts are used to obtain processor time and how processing of interrupted jobs may later be resumed,
(typical sources of interrupts should be identified and any algorithms and data structures should be described);
Define and explain the purpose of scheduling, job queues, priorities and how they are used to manage job throughput;
Explain how memory is managed in a typical modern computer system (virtual memory, paging and segmentation should be described along with some of the problems which could occur, such as disk threshing);
Describe spooling, explaining why it is used;
Describe the main components of a typical desktop PC operating system, including the file allocation table (FAT) and how it is used, and the purpose of the boot file.

Alicia Sykes

Notes

Operating systems must:

- Provide and manage hardware resources
- Provide an interface between the user and the machine
- Provide an interface between application software and the machine
- Provide security for data on the system
- Provide utility software to allow maintenance to be done

Why OS's were developed:

- If a program needed an input or an output the program would have to contain the code to do this. There were often multiple occurrences.
- Short sub-routines were developed to carry out these basic tasks, like get the key pressed.
- The joining together of these sub-routines lead to the input-output-control-system (IOCS). As this started to get more complex it turned into the OS.
- Also as programs started to be written in higher-level languages translators were needed, these were incorporated into the OS

Types of Interrupts

- I/O interrupts – generated by the I/O devices to signal a job is complete or an error has occurred
- Timer interrupt – generated by internal clock to signal that the processor should attend a time critical activity
- Hardware error – for example power failure, where the computer must try to shut down as safely as possible.
- Program interrupt – generated due to an error in a program, such as violation of memory usage.

Dealing with interrupts

- Interrupts are stored in queues
- The position in the queue is determined by its importance to the user, system and job – priority.

Objectives of Scheduling

- Maximise the use of the whole computer system
- Be fair to all users
- Provide reasonable response time to all users – weather online or batch processing users
- Prevent the system failing or becoming overloaded
- Ensure the system is consistent by giving similar response times to similar

Deciding which order to schedule jobs:

- Priority – where some jobs have higher priority than others
- I/O bound jobs – if a job is bound to a device, and is not served first it may prevent the devise being served efficiently.
- Type of job – batch processing and real time jobs require different response times

- Resources used so far
- Waiting time – the amount of time a job has been waiting to use the processor

Ready, running and blocked queue

- A job may be in any one of three states
- Ready
- Running
- Blocked – e.g. waiting for a peripheral devise

What is a scheduler?

The scheduler is just a program, a set of instructions, used by the OS to decide where each of the jobs which are in its control should be and in what order to manipulate them.

Types of schedulers

- High-level scheduler (HLS) – places job in the ready queue, makes sure that the system is not overloaded.
- Medium-level scheduler (MLS) – swaps jobs between the main memory and backing store.
- Low-level scheduler (LLS) – moves jobs in and out of the ready state, and decides which order to place them in the running state

Pre-emptive and non-pre-emptive schedulers

- Non-pre-emptive - puts a job in the running state then leaves it there until the job does not need to run any more but needs to go to one of the other states, for example it may wait for an input or may have finished executing altogether
- Pre-emptive - has the power to decide that the job that is running should be made to stop to make way for another job

Methods of deciding what order jobs should go in

- First com first serve (FCFS) – the first job to enter the running queue is the first job to enter the running state. Favours long jobs.
- Shortest job first (SJF) – sort the jobs in the ready queue in ascending order of time needed
- Round robin (RR) – gives each job a maximum length of processor time, it will then go to the back of the queue again. Once its finished, it leaves the queue.
- Shortest remaining time (SRT) – the ready queue is sorted on the amount of expecting time still to do. Favours short jobs, and there's a danger long jobs may never be started.
- Multi-level feedback queues (MFQ) – involves several queues of different priorities with jobs migrating across

Memory management

In order for a job to be processed, it must be stored in the memory – obviously. If several job are stored in the memory, their data must be protected from the actions of other jobs.

How free memory can be allocated
If a small gap in the memory becomes free, but it is not big enough for the next job, there are a number of options:

- All the jobs could be moved upwards, so that all the bits of free space are together, and the next job can fit. Although this would use a lot of processing power, as all the addresses would have to be recalculated.
- Another solution would be to split up the new job into available free memory. Although doing this could cause the jobs to all get split up into many tiny parts.

Linkers and Loaders

- The loader – small utility program that loads jobs and adjusts addresses.
- The linker – links parts of the program together

Linkers and loaders are integral to the management of memory space

Paging and Segmentation

- Pages are equal sized sections, where a job will be allocated a number of pages to store itself in. They may be in order, or out of order.
- Segments are of variable size to match the size of the job

Virtual Memory
When data is not needed to be accessed immediately, or programs have been 'minimised' for a long time, they may be transferred to virtual memory so as not to fill up the main memory. Virtual memory is on the hard disk, although it behaves exactly like the RAM.

Disk Thrashing
This is where the code contains many jump instructions so the processor spends most the time switching between the virtual memory and main memory. It involves the disk continuously searching for pages.

Spooling
This is where an I/O devise is a lot slower than the processor, so the job is moved into another storage location while it is being used. It is kept track of in a spool queue. A spool queue is only a reference to the jobs, and not a proper queue. A spool queue also allows for priorities, more important jobs can push into the queue and be done first.

Multi-tasking OS
There are two main types of OS, command driven e.g. MS DOS, and GUI. All OS's allow the user to do basic things like rename, delete and move files stored in a hierarchical structure on the disk. Most GUI's also allow for much more than this, they let the user multi-task, where several programs are running.

File Allocation Table (FAT)

- This is a table that uses a linked to point to the blocks on the disk that contain files.
- In order for this to be done, the disk must first be formatted, which involves dividing the disk, radially, into sectors and into concentric circles called tracks. Two or more sectors on a single track make up a cluster.
- To find a file, the OS looks for the file name, and gets the corresponding cluster number, which can be used to find the file.
- When a file is deleted, the clusters that were used to find the file can be set to zero.
- The FAT table is usually loaded into the RAM to speed things up.

Loading and OS

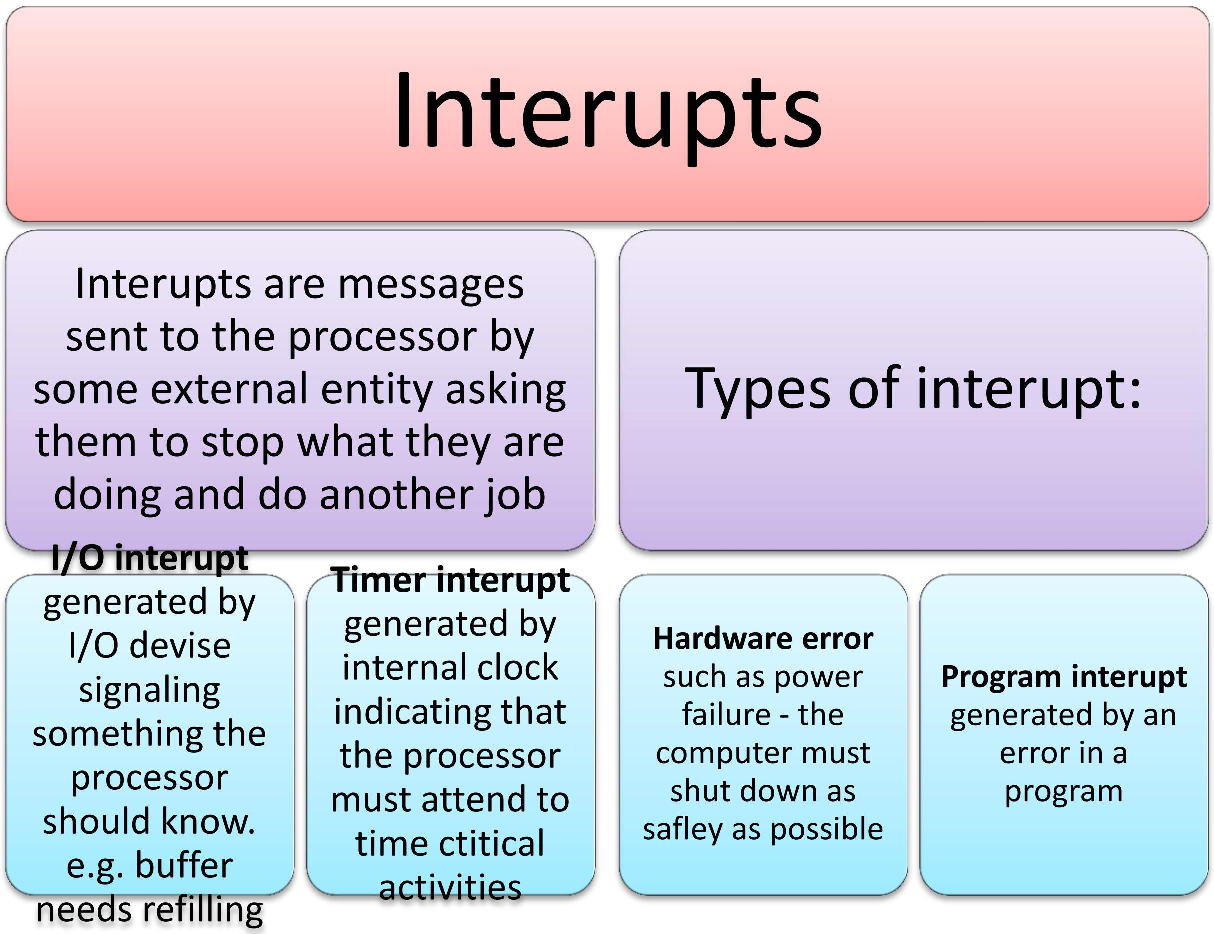
- PC is switched on
- Contains very few instructions
- Runs the power on self-test (POST) which resides in the permanent memory
- The POST clears the registers, loads the address of the first instruction in the boot program into the program counter
- The boot program is stored in the read-only memory and contains the skeleton of basic input output system. The BIOS structure is stored in ROM.
- The user-defined parts of the BIOS are stored in the CMOSRAM.
- The CPU then sends signals to check that all the hardware is working correctly
- This includes checking busses, system clock, RAM, disk drives and keyboard.
- If any of these devises contain their own BIOS then this is incorporated into the system's BIOS
- The PC is now ready to load the OS
- The boot program checks if a disk is present for drive A, if so it looks for an OS on that disk. If no OS is found, an error message is produced.
- If there was no disk in drive A, the boot program will check drive C.
- Once found the OS, if it is Windows, the boot program will look for MSDOS.SYS, and IO.SYS which holds the extensions to the ROM BIOS and contains a routine called SYSINIT which controls the rest of the boot procedure.
- SYSINIT then takes control and loads MSDOS.SYS which works with the BIOS to manage files and execute programs
- The OS searches the root directory for a boot file such as CONFIG.SYS which tells the OS how many files may be opened at the same time and instructions on how to load the necessary devise drivers.
- The OS tells MSDOS.SYS to load a file called COMMAND.COM, which is in three parts. The first part is a further I/O extension which joins with the BIOS to become part of the OS. The second part contains resident OS commands such as DIR and COPY.
- The files CONFIG.SYS and AUTOEXEC.BAT are created by the user so that the PC starts up in the same configuration each time it is switched on.

Keywords and Definitions

- **Operating System (OS)** – A piece of software that provides a platform on which the applications software can run. It controls the hardware, and provides communication with the outside world.
- **Interrupt** - messages sent to the processor by some external entity asking them to stop what they are doing and do another job
- **Scheduler** - a program, or set of instructions, used by the OS to decide where each of the jobs which are in its control should be and in what order to manipulate them
- **High-Level Scheduler (HLS)** – scheduler that places the jobs in the ready queue, and ensures that the system never gets overloaded.
- **Medium-Level Scheduler (MLS)** – Scheduler that swaps jobs between main memory and backing store.
- **Low-Level Scheduler (LLS)** - moves jobs in and out of the ready state, and decides which order to place them in the running state.
- **Pre-Emptive Scheduler** – scheduler that is allowed to move jobs out of the running state and into the ready queue. Has the power to move jobs.
- **Non-Pre-Emptive Scheduler** – scheduler that does not have the power to move jobs until they are finished.
- **First com first serve (FCFS)** – a priority method where the first job to enter the running queue is the first job to enter the running state. Favours long jobs.
- **Shortest job first (SJF)** – a priority method where jobs are sorted in the ready queue in ascending order of time needed
- **Round robin (RR)** – a priority method which gives each job a maximum length of processor time, it will then go to the back of the queue again. Once its finished, it leaves the queue.
- **Shortest remaining time (SRT)** – a priority method where the ready queue is sorted on the amount of expecting time still to do. Favours short jobs, and there's a danger long jobs may never be started.
- **Multi-level feedback queues (MFQ)** – a priority method which involves several queues of different priorities with jobs migrating across
- **Memory Management** - One of the key jobs of the OS, managing memory and allocating it accordingly to jobs.
- **The Linker** – a utility program included in the OS which links parts of the program together through the memory.
- **The Loader** – a utility program which loads jobs into the memory and adjusts addresses accordingly.
- **Pages** – equal sized sections in the memory, which are fixed and jobs are allocated a number of pages.
- **Segments** – similar to pages, although the size can be variable to match the job
- **Virtual Memory** - is part of the hard disk, but has faster speeds, it is slower than the RAM, but is usually considerably bigger. Acts in the same way as RAM

- **Disk Trashing –** caused by many jump instructions in the code, requiring main memory and virtual memory to be repeatedly switching. The hard disk will have to continuously search for pages.
- **Spooling -** It is a method used to place input and output on a fast access storage device, such as a disk, so that slow peripheral devices do not hold up the processor. It allows for queues when several jobs want to use peripheral devices at the same time. It stops different input and outputs becoming mixed up.
- **Command Driven OS –** an operating system without a GUI for example MSDOS
- **GUI OS** – An OS with a GUI, like most modern ones e.g. Windows 8
- **Multi-tasking OS** – most GUI OS's support multi-tasking, where multiple jobs can be carried out simultaneously.
- **File Allocation Table (FAT) -** a table that uses a linked lists to point to the blocks on the disk that contain files, the actual table is usually stored in the RAM.
- **Power on self-test (POST) –** a routine the resides in the permanent memory, it clears the registers, loads the address of the instruction into the program register. It then sends signals to the necessary hardware to check it's all there and working properly.
- **Basic Input Output System (BIOS) -** instructs the computer on how to perform a number of basic functions such as booting and keyboard control.
- **CMOS RAM** – stores the user-defined part of the BIOS. The BIOS is customisable.
- **Boot Program –** gets the system ready to accept an operating system. Unalterable and stored in the ROM.
- **Boot File** - contains some basic parameters to which the system will operate. It can be altered, and is stored in the CMOS RAM.

Key Points Posters



Interupts

Interupts are messages sent to the processor by some external entity asking them to stop what they are doing and do another job

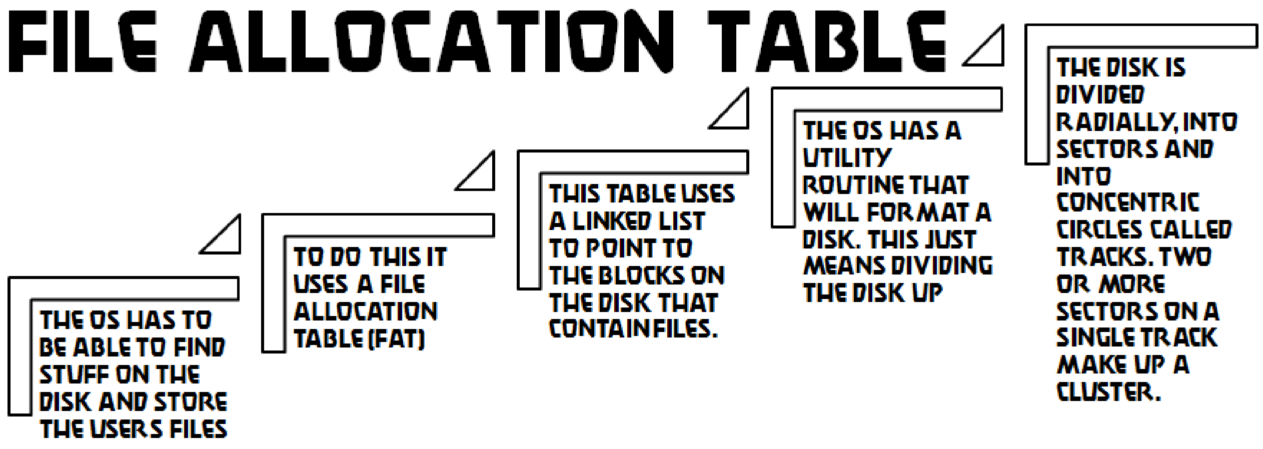
Types of interupt:

I/O interupt generated by I/O devise signaling something the processor should know. e.g. buffer needs refilling

Timer interupt generated by internal clock indicating that the processor must attend to time ctitical activities

Hardware error such as power failure - the computer must shut down as safley as possible

Program interupt generated by an error in a program



FILE ALLOCATION TABLE

THE OS HAS TO BE ABLE TO FIND STUFF ON THE DISK AND STORE THE USERS FILES

TO DO THIS IT USES A FILE ALLOCATION TABLE (FAT)

THIS TABLE USES A LINKED LIST TO POINT TO THE BLOCKS ON THE DISK THAT CONTAIN FILES.

THE OS HAS A UTILITY ROUTINE THAT WILL FORMAT A DISK. THIS JUST MEANS DIVIDING THE DISK UP

THE DISK IS DIVIDED RADIALLY, INTO SECTORS AND INTO CONCENTRIC CIRCLES CALLED TRACKS. TWO OR MORE SECTORS ON A SINGLE TRACK MAKE UP A CLUSTER.

SCHEDULING POLICIES

FCFS

- First Come First Served
- The first job to enter the ready queue is the first to enter the running state.
- This favours long jobs because once in the running state there is nothing to stop them carrying on running

SJF

- Shortest Job first
- Jobs are sorted in the ready queue in ascending order of times expected to be needed by each job. New jobs are added to the queue in such a way as to preserve this order

RR

- Round Robin
- gives each job a maximum length of processor time (called a time slice) after which the job is put at the back of the ready queue and the job at the front of the queue is given use of the processor. If a job is completed before the maximum time is up it leaves the system

SRT

- Shortest Remaning Time
- The ready que is sorted by the expected remaining time to complete a job. Long jobs may never get started

MFQ

- Multi level feedback ques
- Involves several queues of different priorities with jobs migrating downwards.

Spooling

What is Spooling?

- It is a method used to place input and output on a fast access storage device, such as a disk, so that slow peripheral devices do not hold up the processor.
- It allows for ques when severaljobs want to use peripherial devices at the same time
- It stops different input and outputs becoming mixed up

An example of spooling

- If two or more jobs are sent to a printer at the same time, the jobs are sent to a spool que where they wait for the printer to be free.
- The spooll que only stores reference to where the jobs are stored on a hard drive.

Benifits of spooling

- Keeps output of different ques seperate
- Saves the user having to wait for the processor
- Lets the processor get on with something else while the jobs are queued

Loading an OS

POST

- When the computer is powered on it only has access to the ROM, which is only big enough to hold a few instructions
- The computer then carries out the power on slef test (POST), which checks that everything needed for the computer to coe to life is availible.
- It also clears the registers of the CPU and loads the address of the first instruction in the boot program into the CPU

Boot Program

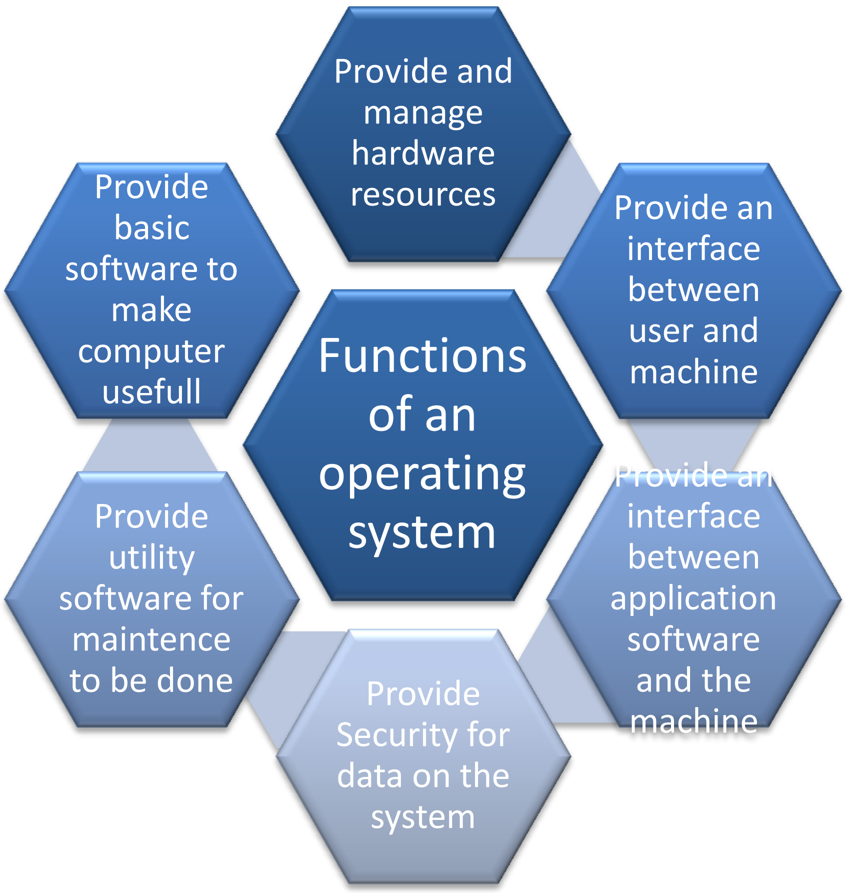
- The boot program is stored in the ROM, and contains the skeleton of the BIOS (only the main structure of the BIOS, because it contains user-defined options, which need to be modifiable, and ROM is read only) The user-defined parts of the BIOS would be stored in the CMOS RAM
- The CPU then sends signals to check all the hardware is working. Hardware containing it's own BIOS will be incorporated into the systems BIOS.

Ready to load the OS

- Once all the initial checks are done the PC is ready to load the OS. It will first check drive A, and if any bootable medium is present it will boot from it. If not it will check for drive C or return an error message to the user
- If the OS is Windows the boot program will look for IO.SYS and MSDOS.SYS, and load them.
- IO.SYS holds extensions to the ROM BIOS and contains a routine called SYSINIT which controls the rest of the boot procedure.
- SYSINIT now takes control and loads MSDOS.SYS which works with the BIOS to manage files and execute programs

Loading the OS

- The OS searches the root directory for a boot file such as CONFIG.SYS which tells the OS how many files may be opened at the same time., and may also contain instructions to load various device drivers.
- The OS tells MSDOS.SYS to load a file called COMMAND.COM which is in three parts. The first part is a further extension to the I/O functions and it joins the BIOS to become part of the OS. The second part contains resident OS commands, such as DIR and COPY
- The files CONFIG.SYS and AUTOEXEC.BAT are created by the user so that the PC starts up in the same configuration each time it is switched on.


Functions of an operating system
Provide and manage hardware resources
Provide an interface between user and machine
Provide an interface between application software and the machine
Provide Security for data on the system
Provide utility software for maintence to be done
Provide basic software to make computer usefull

Linkers and Loaders

The **Loader** is a small program that loads the jobs and adjusts the addresses into necessary places. It helps the OS with memory management

The **Linker** is the program that links all the different memory locations and parts of the program together.

Paging and Segmentation

Paging is when the memory is divided into equal sections, each section being a page. Jobs are then allocated a number of pages. The pages for each job may be in a logical order, or they may be scattered about where ever there is a free page.

Segmentation is similar to paging although the segments that the memory is split up into can be of a variable size, this makes better use of the memory, as less is wasted although is more complex and difficult to control and more complex to predict.

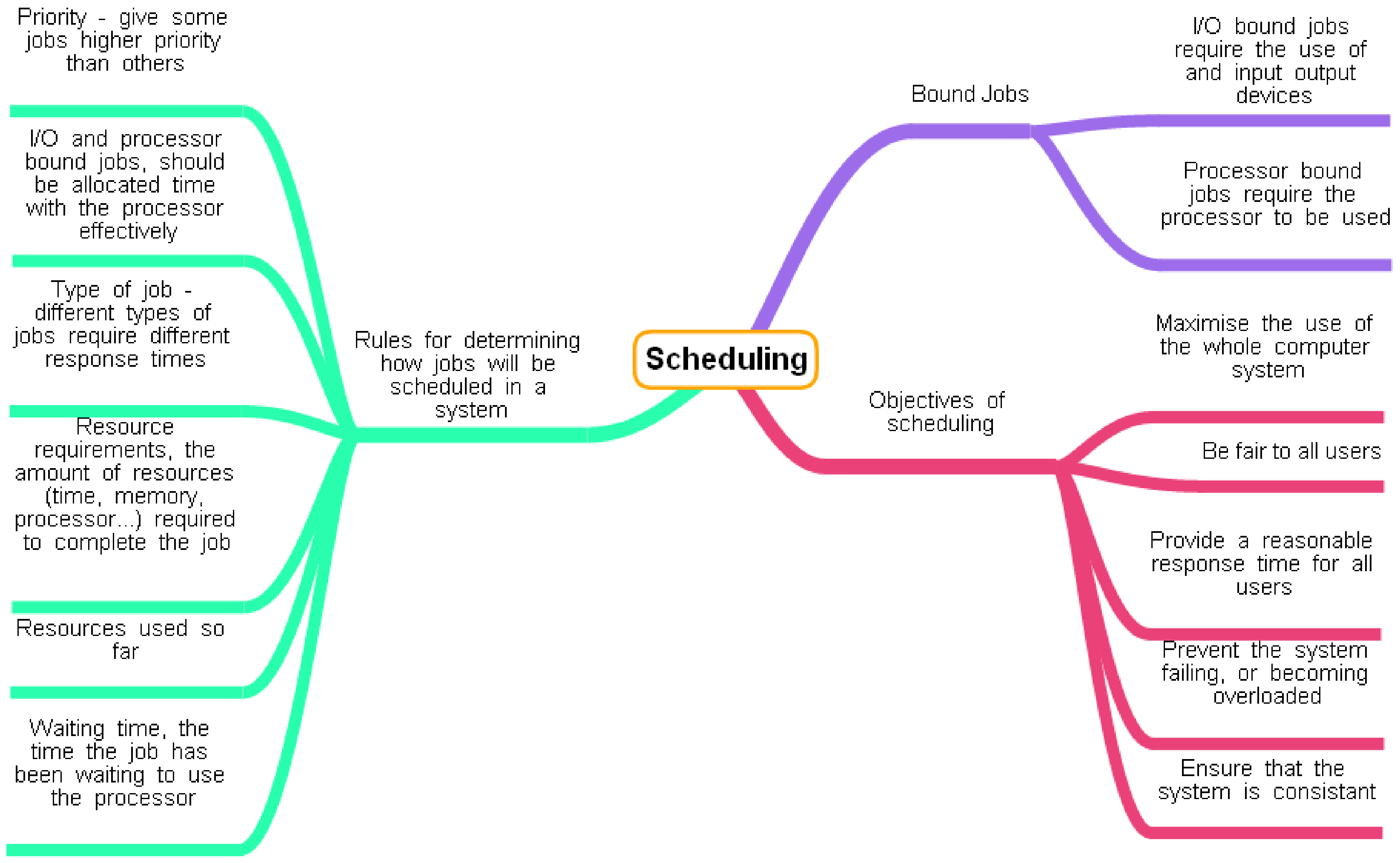
Virtual Memory

When a program is running, only the pages that contain the necessary data are needed to be stored in the RAM, as there is limited amount of space. The rest of the data that may be needed later on in the program or will be needed in a minute, (like a minimised window) will be stored in virtual memory.

Virtual Memory is part of the hard disk, but has faster speeds, it is slower than the RAM, but is usually considerably bigger.

If the memory is nearly full, more of the data will be stored in virtual memory, the OS will have to spend a long time both loading and saving pages.

Disk thrashing occurs when the OS has to spend a considerable proportion of its time swapping data between virtual and real memory.


Scheduling
Bound Jobs
I/O bound jobs require the use of and input output devices
Processor bound jobs require the processor to be used
Objectives of scheduling
Maximise the use of the whole computer system
Be fair to all users
Provide a reasonable response time for all users
Prevent the system failing, or becoming overloaded
Ensure that the system is consistant
Rules for determining how jobs will be scheduled in a system
Priority - give some jobs higher priority than others
I/O and processor bound jobs, should be allocated time with the processor effectively
Type of job - different types of jobs require different response times
Resource requirements, the amount of resources (time, memory, processor...) required to complete the job
Resources used so far
Waiting time, the time the job has been waiting to use the processor

Past Exam Questions with Mark Scheme Answers

Why interrupts are used in a computer system

to obtain processor time...
for a higher priority task
to avoid delays
to avoid loss of data
as an indicator to the processor…
that a device needs to be serviced

State some sources of interrupts

(imminent) power failure/system failure
peripheral eg printer (buffer empty)/hardware
clock interrupt
user interrupt eg new user log on request
software

Why is scheduling used?

maximise number of users…
…with no apparent delay
maximise number of jobs processed…
…as quickly as possible
obtain efficient use of processor time / resources…
…dependent upon priorities
…to ensure all jobs obtain processor time/long jobs do not monopolise the processor

Why are jobs given different priorities in a job queue?

some jobs are more urgent than others
priorities are used to maximise the use of the computer resources

Why is memory management necessary?

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

Why is virtual memory needed?

to allow programs to run that need more memory than is
available

How is virtual memory used?

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

What's the problem of disk threshing?

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

Methods of scheduling

round robin
each user allocated a short period of time/in a sequence
or
system of priorities
highest priority first
or
length of job

shortest job first
or
first come, first served
jobs processed in order of arrival

What is spooling and when is it used?

output data to disk drive/storage device
for printing at another time
to allow sharing/on a network
job references stored in a queue/buffer
avoids delays / avoids speed mismatch
as printers are relatively slow
jobs can be prioritised

How are paging and segmentation similar?

ways of partitioning memory
allow programs to run despite insufficient memory/used for virtual memory
segments and pages are stored on backing store
segments and pages are assigned to memory when needed

How are paging and segmentation 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

What problems may occur from paging and segmentation?

disk threshing
more time spent swapping pages than processing
computer may 'hang'

When is the boot file used?

while the operating system is loading
when the computer is switched on
after POST

What is the purpose of the boot file

provides personal settings

What's virtual memory

use of backing store…
…as additional memory
uses paging / swapping pages (between memory & backing store)
holds part of the program not currently in use
allows large programs to run (when memory size is insufficient)

What's the purpose and use of the file allocation table?

a map of where files are stored…
…in backing store/hard disk
provides addresses/pointers to (start of) files
stores file names, and stores file sizes
stores access rights,
identifies free space
is updated by the operating system when files are saved/deleted
Is used by the operating system when files are accessed

Further Recourses

VRS(http://vrs.as93.net)

- Prezzi Presentation
- More revision posters
- More past exam questions
- Links for further reading
- Key words list
- Revision Quizzes
- This document
- User contributed content

Revision Quizzes (http://RevisionQuizzes.com)

- Jobs the OS must do quiz
- Loading an OS quiz
- Functions of the OS summary quiz