ie
SCHOOL OF
HUMAN SCIENCES
& TECHNOLOGY

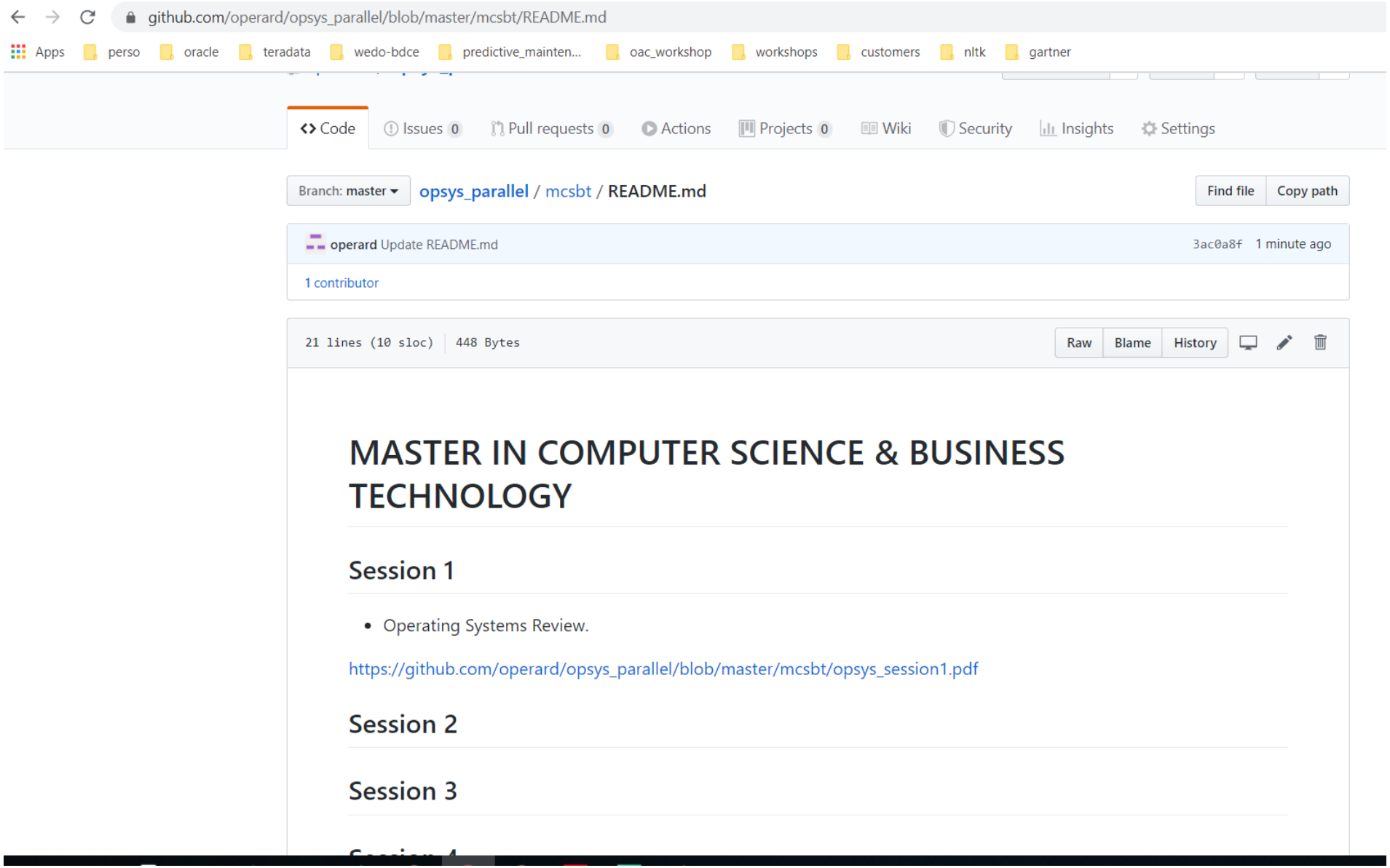
OPERATING SYSTEMS

MASTER IN COMPUTER SCIENCE & BUSINESS TECHNOLOGY

Professor: Olivier Perard
Email: operard@faculty.ie.edu
Github: https://github.com/operard/opsys_parallel/blob/master/mcsbt/README.md

SCHOOL OF HUMAN SCIENCES & TECHNOLOGY

https://github.com/operard/opsys_parallel/blob/master/mcsbt/README.md



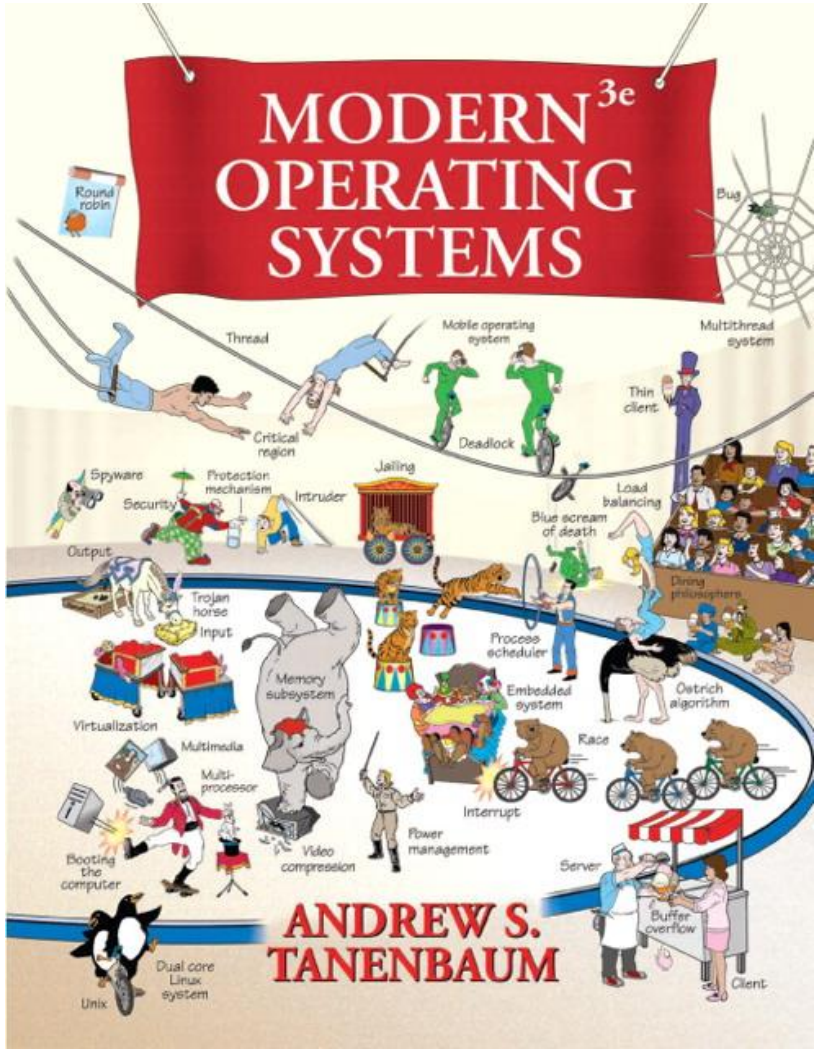
ie
SCHOOL OF
HUMAN SCIENCES
& TECHNOLOGY

Books References

Operating Systems book

- A free online version of “Modern Operating Systems” can be downloaded from:
- https://github.com/gramasaurous/opsys/blob/master/Modern.Operating.Systems.3rd.Edition.pdf

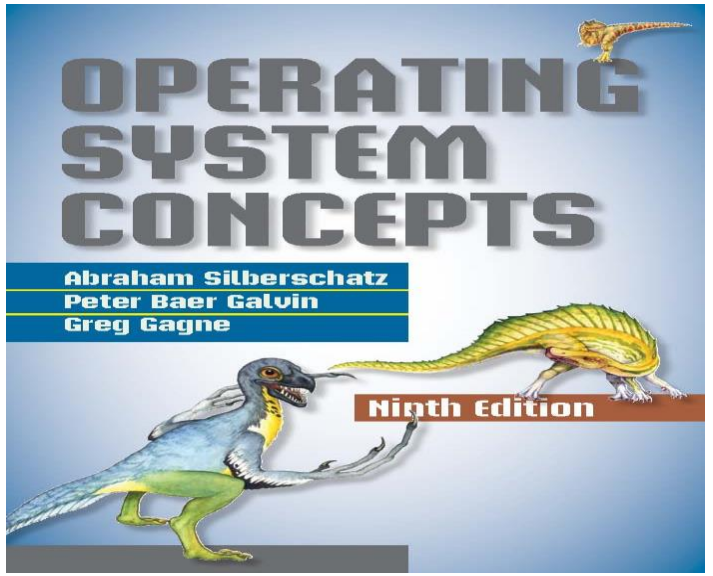
- https://github.com/gramasaurous/opsys/blob/master/MOS_3e_SM.pdf

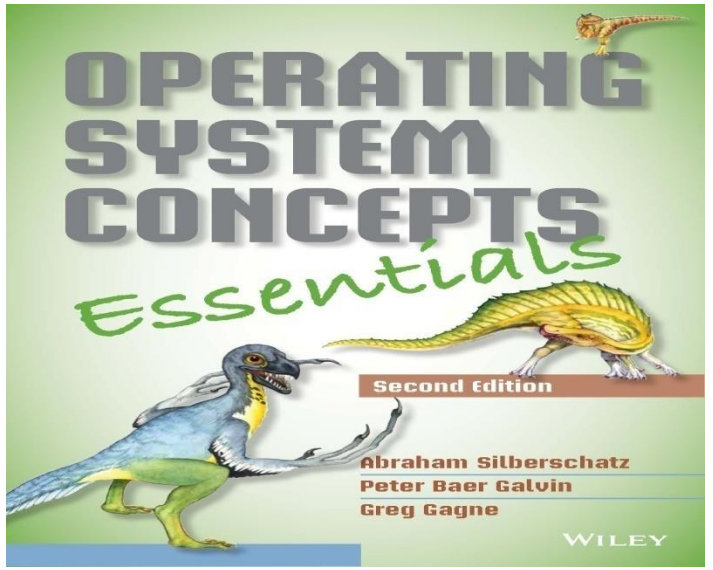


Other links (1)

- A. Silberschatz, P. B. Galvin, and G. Gagne, “Operating Systems Concepts (Essentials)”, 9th Edition, John Wiley & Sons, 2012.
- http://codex.cs.yale.edu/avi/os-book/

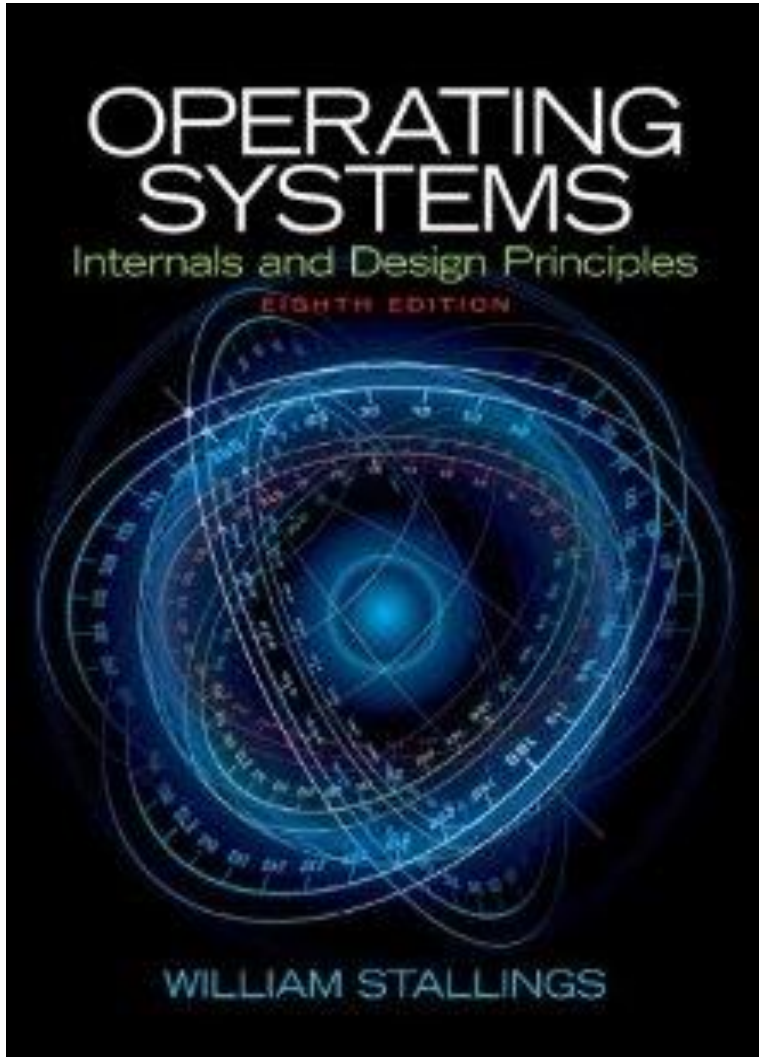
- https://www.os-book.com/OSE2/index.html
- https://www.os-book.com/OS9/index.html
- https://www.os-book.com/OS10/index.html





Other links (2)

- W. Stallings, “Operating Systems: Internals and Design Principles”, 8th ed, Pearson, 2015.
- http://williamstallings.com/OperatingSystems/



SCHOOL OF
HUMAN SCIENCES
& TECHNOLOGY

Current Syllabus

Syllabus

- **Session 1**
  - What is an Operating System?
  - History of Operating Systems
  - Computer Hardware
  - File System
  - Operating Systems
- **Session 2**
  - System Calls
  - Monolithic Systems
  - Virtual Machines
  - Process Model
  - Cloud Google Shell *Practice
  - Process Life Cycle

ie SCHOOL OF HUMAN SCIENCES & TECHNOLOGY

Syllabus

- **Session 3**
  - **Interprocess Communication**
  - Producer-Consumer problem
  - Deadlock
  - Banker's Algorithm
- **Session 4**
  - **Concurrency, synchronization and Scheduling**
  - Read and Write problem
  - Monitors
  - Semaphores
  - **Dining Philosophers Problem *Practice**

Syllabus

- **Session 5**
  - **Concurrency, synchronization and Scheduling**
  - Process Scheduler
  - Scheduler Algorithms: FCFS, SJF,
  - Python FCFS *practice
- **Session 6**
  - **Memory Management**
  - Address space
  - Dynamic address
  - Segmentation, Paging
  - Replacement policies: FIFO, LRU
  - **LRU python code *practice**

SCHOOL OF HUMAN SCIENCES & TECHNOLOGY

Syllabus

- **Session 7**
  - **File systems**
  - ownership and permissions
  - changing and removing permissions
  - Access modes
  - file creation mask
  - moving around the file system
  - wild cards
  - **File testing *practice**
- **Session 8**
  - **Input Output I/O**
  - overhead, latency, bandwidth
  - Direct Memory Access DMA
  - Sync and Async I/O
  - **Redirection *practice**

Syllabus

- **Session 9**
  - **Multimedia**
  - Browser support
  - Multimedia formats
  - Multimedia operating systems
- **Session 10**
  - **Network**
  - TCP/IP Network model
  - **The routing table *practice**

Syllabus

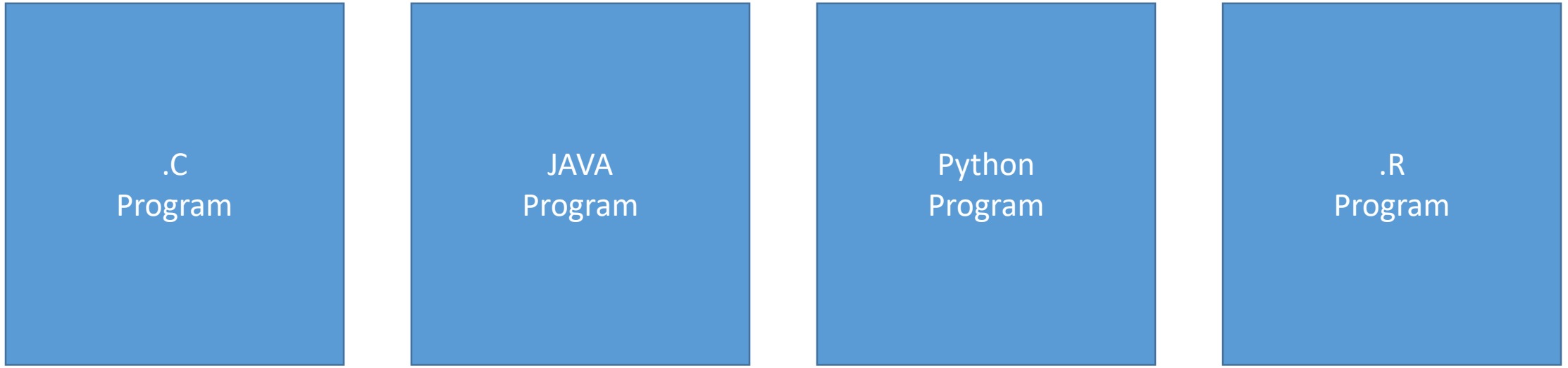
- **Session 11**
  - **Security and Encryption**
  - Phishing
  - Trojan Horse
  - XSS Attacks
  - **my unsecure bank *practice**
- **Session 12**
  - **Exam**

SCHOOL OF
HUMAN SCIENCES
& TECHNOLOGY

New Syllabus

Why must you study Operating Systems?

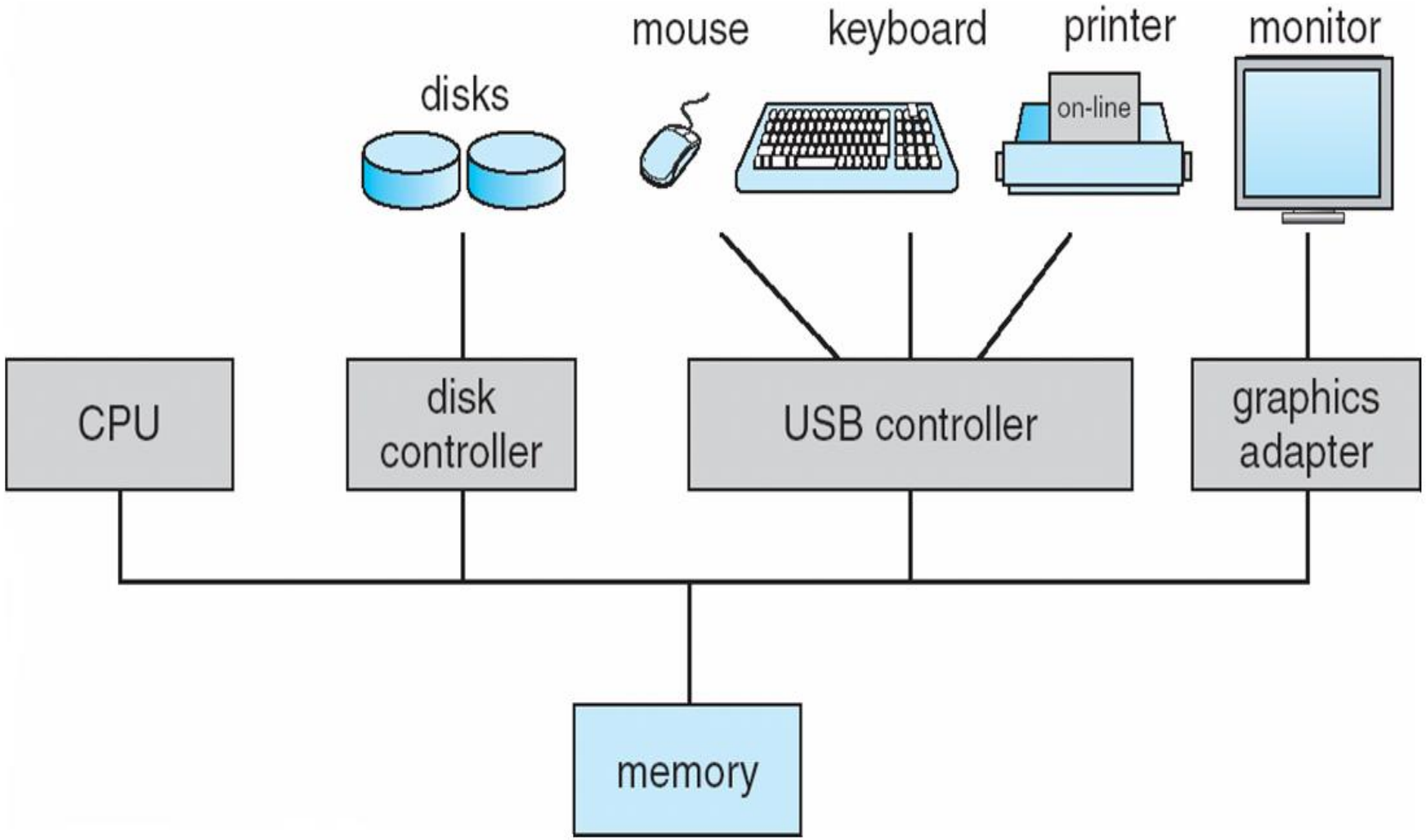
Which programming languages do you use?



What is an Operating System (1)?

- A modern computer consists of:
  - One or more processors
  - Main memory
  - Disks
  - Printers
  - Various input/output devices.
- Managing all these varied components requires a layer of software – the **Operating System (OS).**

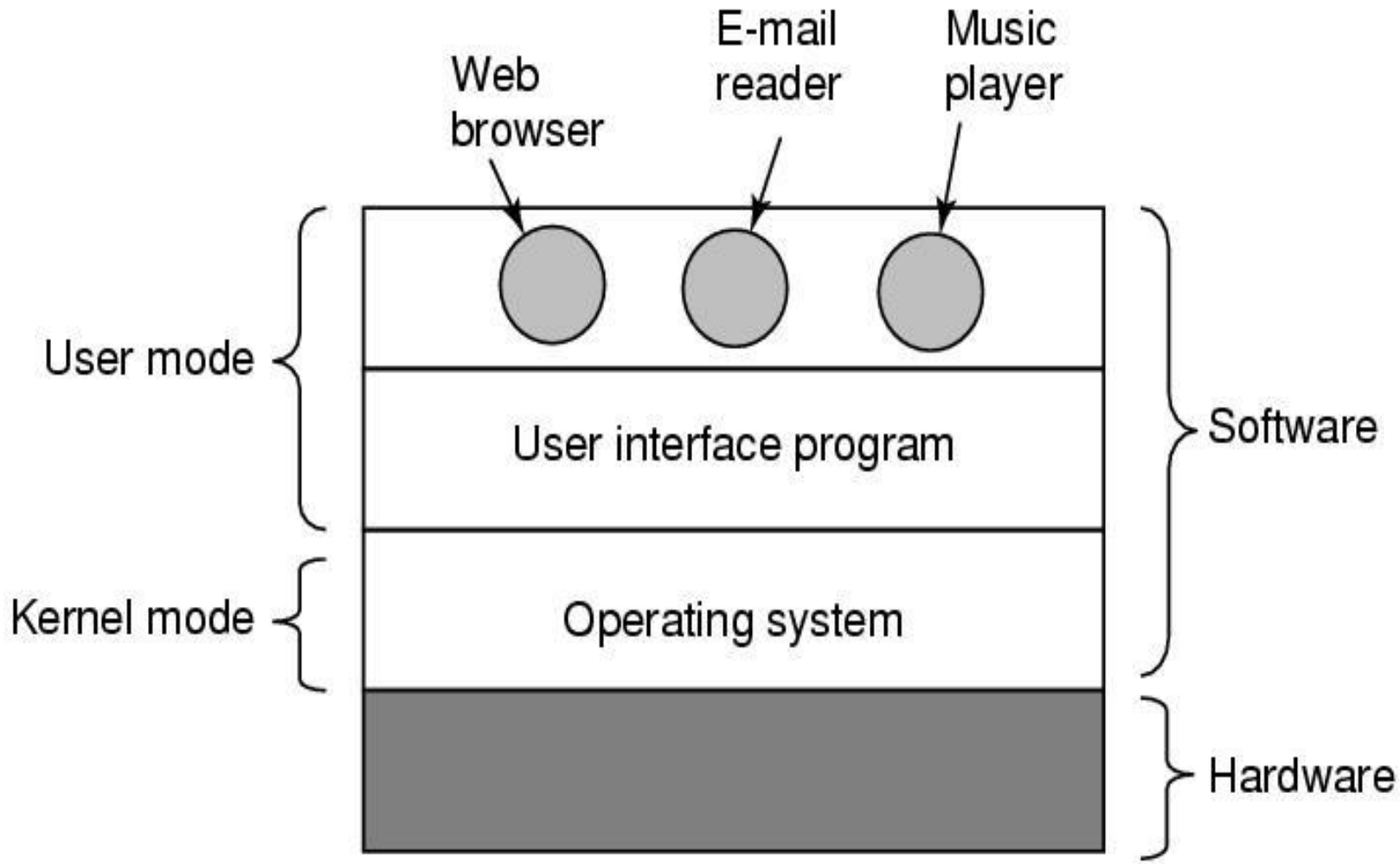
Computer Hardware Organization



What is an Operating System (2)?

- An Operating System is a program that acts as an intermediary/interface between a user of a computer and the computer hardware.
- OS goals:
  - Control/execute user/application programs.
  - Make the computer system convenient to use.
  - Ease the solving of user problems.
  - Use the computer hardware in an efficient manner.

Where does the OS fit in?



Services provided by an OS

- Facilities for program creation
  - editors, compilers, linkers, debuggers, etc.
- Program execution
  - loading in memory, I/O and file initialization.
- Access to I/O and files
  - deals with the specifics of I/O and file formats.
- System access
  - resolves conflicts for resource contention.
  - protection in access to resources and data.

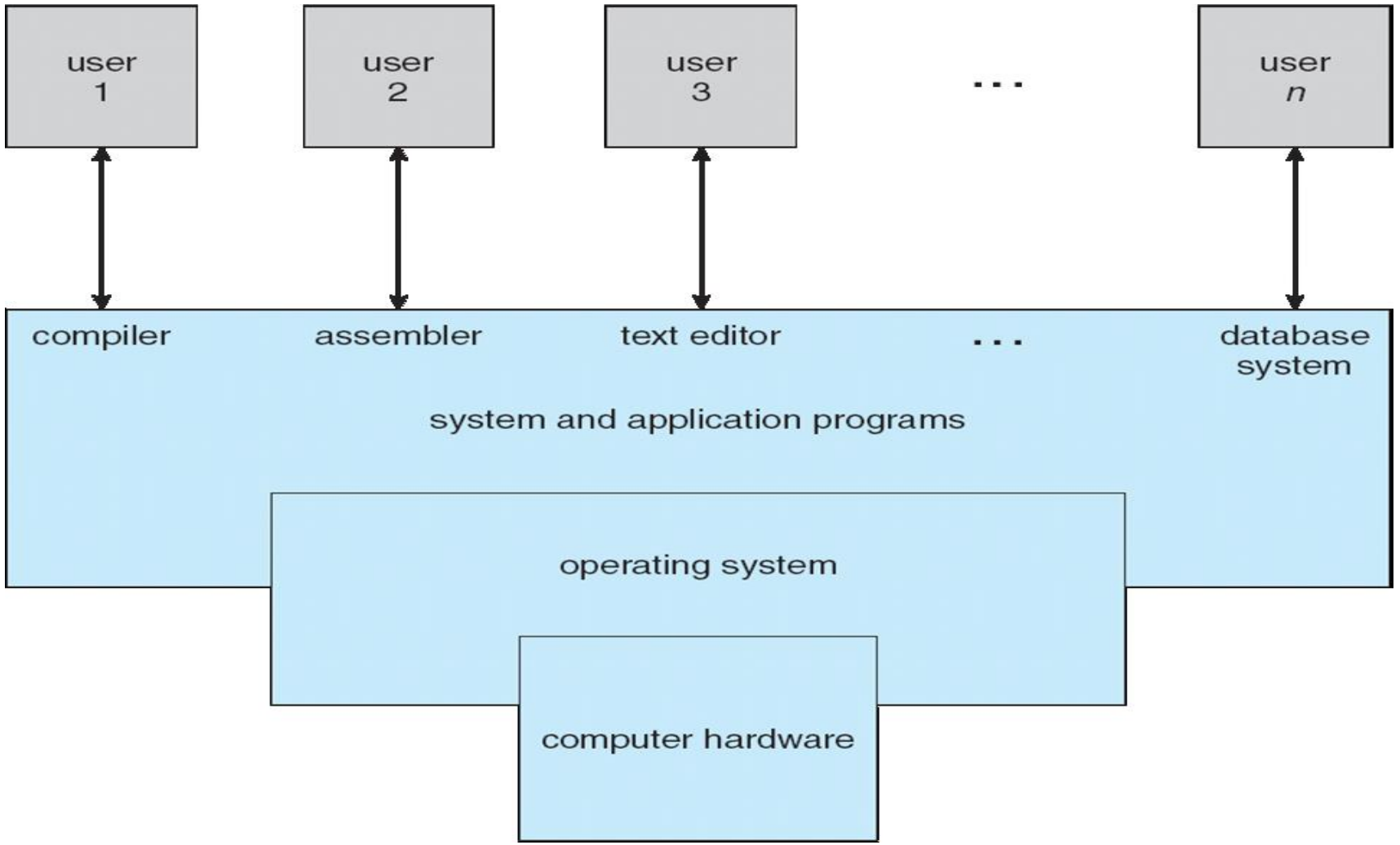
Why are Operating Systems Important?

- Important to understand and know how to correctly use when writing user applications.
- Large and complex systems that have a high economic impact and result in interesting problems of management.
- Few actually involved in OS design and implementation but nevertheless many general techniques to be learned and applied.
- Combines concepts from many other areas of Computer Science: Architecture, Languages, Data Structures, Algorithms, etc.

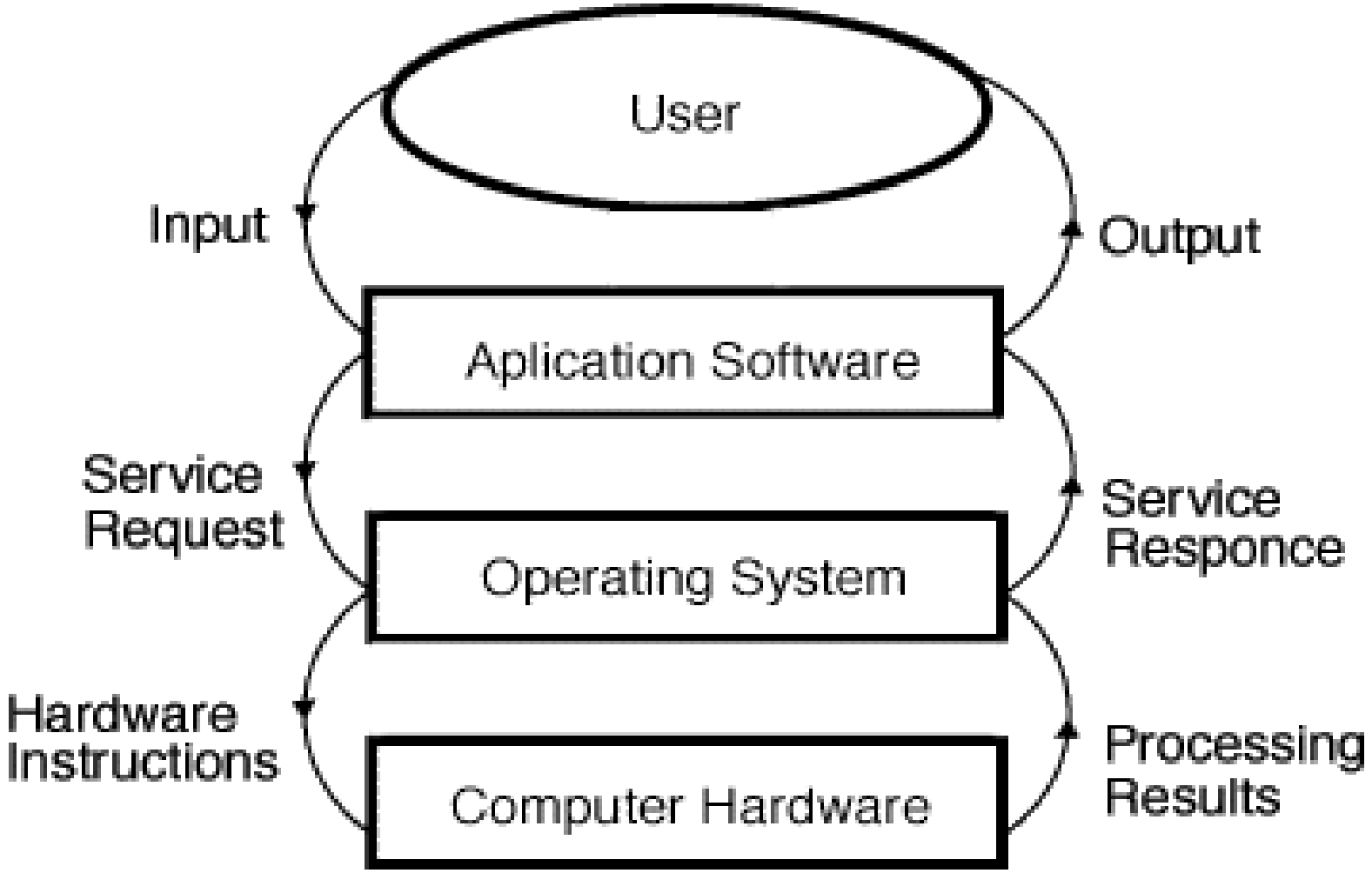
Computer System Components

1. Hardware – provides basic computing resources (CPU, Memory, I/O devices, Communication).
2. Operating System – controls and coordinates use of the hardware among various application programs for various users.
3. System & Application Programs – ways in which the system resources are used to solve computing problems of the users (Word processors, Compilers, Web browsers, Database systems, Video games).
4. Users – (People, Machines, other computers).

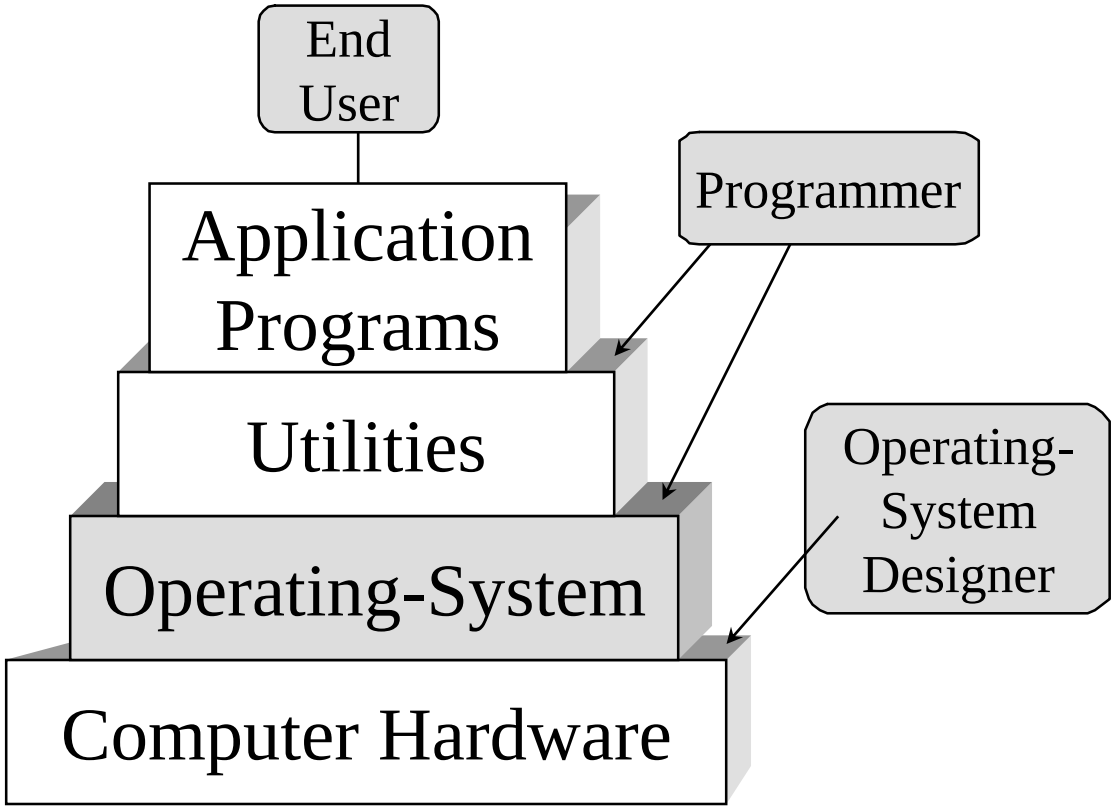
Static View of System Components



Dynamic View of System Components



Layers of a Computer System



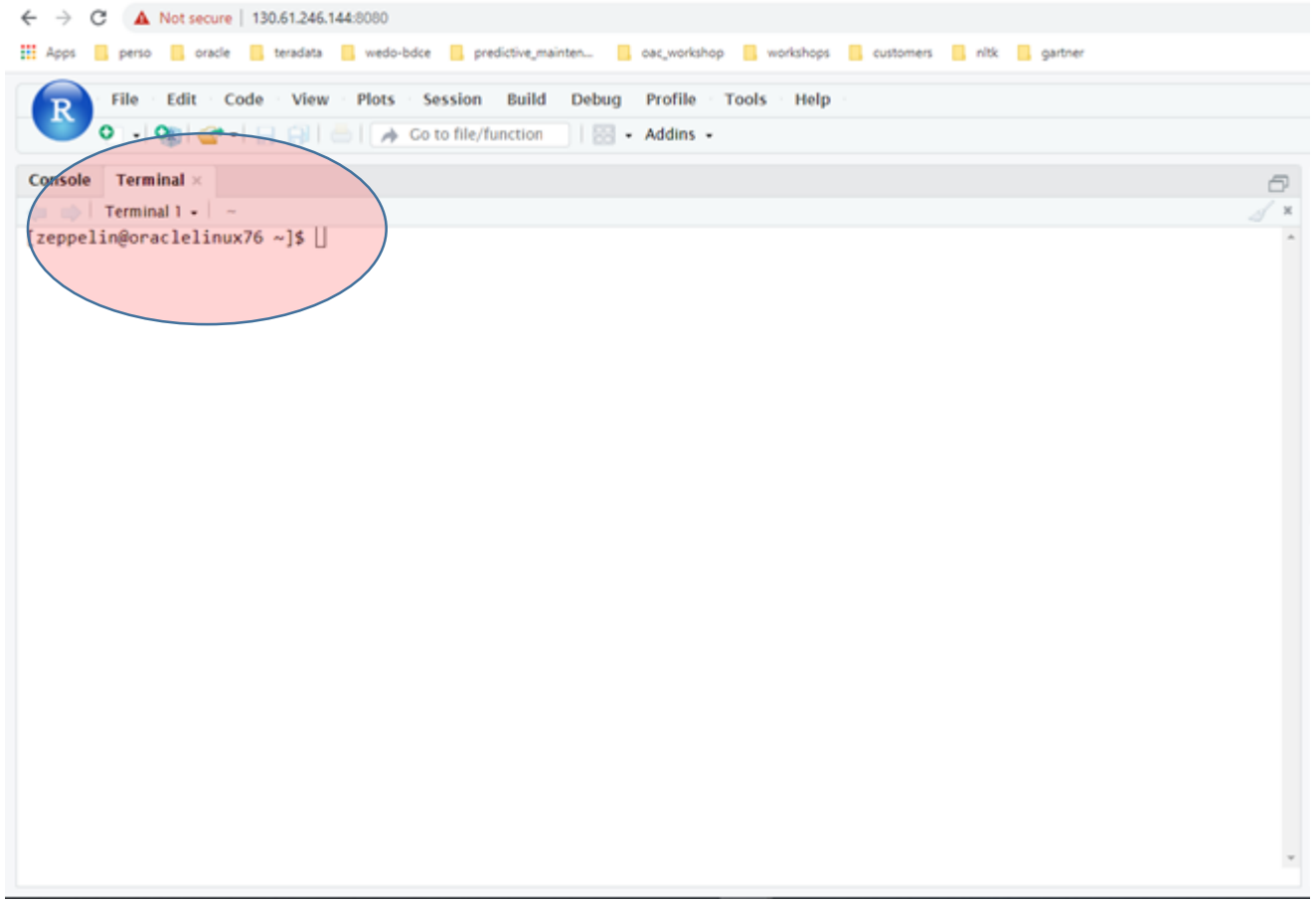
ie
SCHOOL OF
HUMAN SCIENCES
& TECHNOLOGY

Linux OS

Environment for Hands-on-Lab

- IP: 130.61.246.144
- RSTUDIO: 8080
- USERS: workshop001 / ….etc…. / workshop020
- PWD: Welcome2020#
- URL: http://130.61.246.144:8080/

Lab 1: check environment



- Connect to Terminal
- Execute command in Terminal

You need help ?

- **The Linux equivalent of HELP is `man` (manual)**
  - Use `man -k <keyword>` to find all commands with that keyword
  - Use `man <command>` to display help for that command
    - Output is presented a page at a time. Use `b` for to scroll backward, `f` or a space to scroll forward and `q` to quit

Common command

- `pwd` - **print (display) the working directory**
- `cd <dir>` - **change the current working directory to *dir***
- `ls` - **list the files in the current working directory**
- `ls -l` - **list the files in the current working directory in long format**

- `who` **or** `w`
  - List who is currently logged on to the system
- `whoami`
  - Report what user you are logged on as
- `ps`
  - List your processes on the system
- `ps aux`
  - List all the processes on the system
- `echo "A string to be echoed"`
  - Echo a string (or list of arguments) to the terminal

Who's Logged On Right Now?

- The `w` command lists all users logged on right now

```
 5:16pm  up 2 days,  8:46,  1 user,  load average: 0.00, 0.00, 0.00
USER     TTY      FROM              LOGIN@   IDLE   JCPU   PCPU  WHAT
neale    ttyp0    websurfer.reston  4:28pm  1.00s  0.52s  0.18s  w
```

Execute Next commands

- Get help on the `ls` command
- Find out who else is on the system
- What is your current directory
- Redirect the output of the `ls -l /` command to `ls.output` and see what you get

Execute Process Command

- **ps -ef | more**
- **ps aux**
- **ps –e f**
- **top (tape “q” to quit the process)**

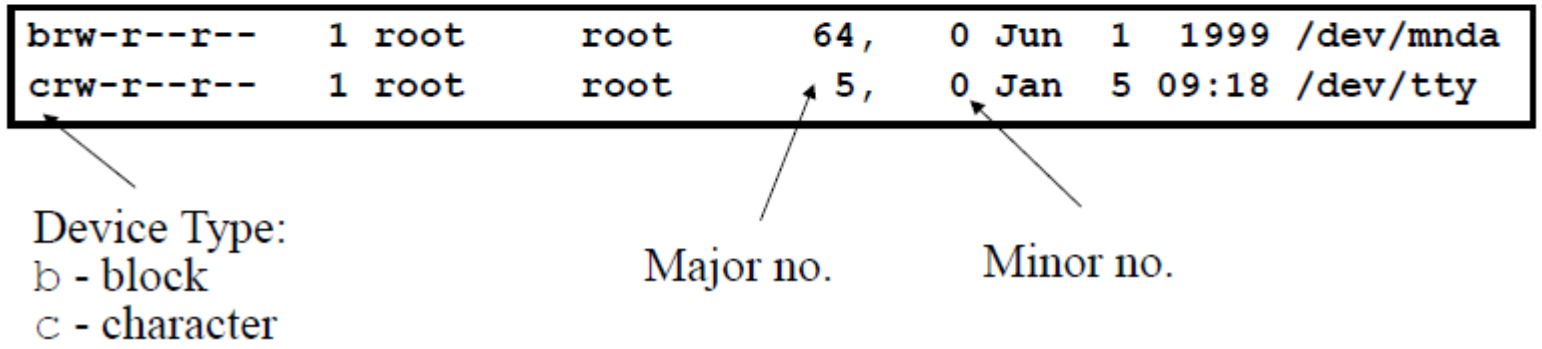
Linux Device Handling

- **Devices are the way Linux talks to the world**
- **Devices are special files in the `/dev` directory (try `ls /dev`)**

/dev/ttyx	TTY devices
/dev/hdb	IDE hard drive
/dev/hdb1	Partition 1 on the IDE hard drive
/dev/dasda	ECKD/CKD/FBA DASD
/dev/dasda1	Partition 1 on DASD
/dev/null	The null device (“hole”)
/dev/zero	An endless stream of zeroes
/dev/mouse	Mouse (not /390)

Device and Drivers

- **Each `/dev` file has a major and minor number**
  - Major defines the device type
  - Minor defines device within that type
  - Drivers register a device type



Special Files - /proc

- **Information about internal Linux processes are accessible to users via the `/proc` file system (in memory)**

/proc/cpuinfo	**CPU Information**
/proc/interrupts	**Interrupt usage**
/proc/version	**Kernel version**
/proc/modules	**Active modules**

```
cat /proc/cpuinfo
vendor_id       : IBM/S390
# processors    : 1
bogomips per cpu: 86.83
processor 0: version = FF, identification = 045226, machine = 9672
```

File System

- **You can view what file systems are mounted using either:**
  - mount
  - df –h
  - cat /etc/fstab

mount

- Mounts a file system that lives on a device to the main file tree
- Start at Root file system
  - Mount to root
  - Mount to points currently defined to root
- `/etc/fstab` used to establish boot time mounting

```
/dev/dasda1     /              ext2    defaults,errors=remount-ro 0 1
/dev/dasdb1     /bin           ext2    defaults,errors=remount-ro 0 1
/dev/dasdc1     /usr           ext2    defaults,errors=remount-ro 0 1
/dev/dasdd1     /usr/local     ext2    defaults,errors=remount-ro 0 1
/dev/dasde1     /usr/man       ext2    defaults,errors=remount-ro 0 1
/dev/dasdf1     /home          ext2    defaults,errors=remount-ro 0 1
/dev/dasdg1     swap           swap    defaults    0       0
none            /proc          proc    defaults    0       0
```

Environment Variables

- **Using Environment Variables:**
  - `echo $VAR`
  - `cd $VAR`
  - `cd $HOME`
  - `echo "You are running on $SYSTEMNAME"`
- **Displaying - use the following commands:**
  - `set` (displays local & environment variables)
  - `export`
- **Variables can be retrieved by a script or a program**

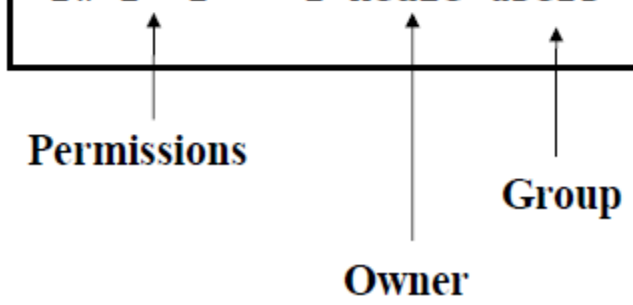
Creating file and directories

- **Files can be created in a number of ways**
  - The output of a command
  - Being edited using vi or your favorite editor
  - By using the `touch` command which creates an empty file or updates the modification and access time information of an existing file
- **Directories are created using the `mkdir` command**

File Permissions

- **The long version of a file listing (`ls -l`) will display the file permissions:**

```
-rwxrwxr-x  1 rvdheij  rvdheij      5224 Dec 30 03:22 hello
-rw-rw-r--  1 rvdheij  rvdheij       221 Dec 30 03:59 hello.c
-rw-rw-r--  1 rvdheij  rvdheij      1514 Dec 30 03:59 hello.s
drwxrwxr-x  7 rvdheij  rvdheij      1024 Dec 31 14:52 posixuft
:
-rw-r--r--  1 neale users       1039 2009-09-10 12:47 a.a
drwxr-xr-x  5 neale users       4096 2011-08-16 20:34 benchmark
drwxr-xr-x  2 neale users       4096 2009-07-30 08:55 bin
drwxr-xr-x  3 neale users       4096 2009-05-16 12:17 BINUTILS
-rw-r--r--  1 neale users       3776 2012-02-24 09:32 bluefin.cs
```



File Commands

- `cp <fromfile> <tofile>`
  - Copy from the <fromfile> to the <tofile>
- `mv <fromfile> <tofile>`
  - Move/rename the <fromfile> to the <tofile>
- `rm <file>`
  - Remove the file named <file>
- `mkdir <newdir>`
  - Make a new directory called <newdir>
- `rmdir <dir>`
  - Remove an (empty) directory

Change File Permissions

- **Use the `chmod` command to change file permissions**
  - The permissions are encoded as an octal number

User			Group			Other		
Read r	Write w	Execute x	Read r	Write w	Execute x	Read r	Write w	Execute x
400	200	100	40	20	10	4	2	1

```
chmod 0755 file  # Owner=rwx Group=r-x Other=r-x
chmod 0500 file2 # Owner=r-x Group=--- Other=---
chmod 0644 file3 # Owner=rw- Group=r-- Other=r--

chmod +x  file   # Add execute permission to file for all
chmod u-r file   # Remove read permission for owner
chmod a+w file   # Add write permission for everyone
```

More Commands

- `awk` - a file processing language that is well suited to data manipulation and retrieval of information from text files
- `chown` - sets the user ID (UID) to owner for the files and directories named by pathname arguments. This command is useful when from test to production

```
chown -R apache:httpd /usr/local/apache
```

More Commands

- `diff` - attempts to determine the minimal set of changes needed to convert a file specified by the first argument into the file specified by the second argument
- `find` - Searches a given file hierarchy specified by path, finding files that match the criteria given by expression

Search Command

- `grep` - **Searches files for one or more pattern arguments. It does plain string, basic regular expression, and extended regular expression searching**

```
find ./ -name "*.c" | xargs grep -i "fork"
```

In this example, we look for files with an extension "c" (that is, C source files). The filenames we find are passed to the xargs command which takes these names and constructs a command line of the form: `grep -i fork <file.1>…<file.n>`. This command will search the files for the occurrence of the string "fork". The "-i" flag makes the search case insensitve.

Kill Process

- **kill - sends a signal to a process or process _group**
  - You can only kill your own processes unless you are root

```
UID         PID   PPID   C STIME TTY          TIME CMD
root       6715   6692   2 14:34 ttyp0    00:00:00 sleep 10h
root       6716   6692   0 14:34 ttyp0    00:00:00 ps -ef
[root@penguinvm log]# kill 6715
[1]+  Terminated              sleep 10h
```

Replace String

- **sed - applies a set of editing subcommands contained in a script to each argument input file**

```
find ./ -name "*.c,v" | sed 's/,v//g' | xargs grep "PATH"
```

This finds all files in the current and subsequent directories with an extension of c,v. sed then strips the ,v off the results of the find command. xargs then uses the results of sed and builds a grep command which searches for occurrences of the word PATH in the C source files.

Archive command

- **`tar` - manipulates archives**
  - An archive is a single file that contains the complete contents of a set of other files; an archive preserves the directory hierarchy that contained the original files.

```
tar -tzf imap-4.7.tar.gz
imap-4.7/
imap-4.7/src/
imap-4.7/src/c-client/
imap-4.7/src/c-client/env.h
imap-4.7/src/c-client/fs.h
```

Viewing Files

- `cat` **"Concatenate"**
- `more` **Display one page at a time**
- `less` **Variant of** `more`
- **Editors**
  - `vi` Visual editor, the default
  - `the` XEDIT/KEDIT/ISPF clone
  - `xedit` X windows text editor
  - `emacs` Extensible, Customizable Self-Documenting Display Editor
  - `pico` Simple display-oriented text editor
  - `nedit` X windows Motif text editor

ie
SCHOOL OF
HUMAN SCIENCES
& TECHNOLOGY

Windows OS

Windows Terminal: cmd

Command Prompt

```
c:\Windows\WinSxS\amd64_microsoft-windows-p..rastructureconsumer_31bf3856ad364e35_10.0.17763.1_none_a4a27884889b78d9\Rul
es.System.CPU.xml

C:\Users\operard>help dir
Displays a list of files and subdirectories in a directory.

DIR [drive:][path][filename] [/A[[:]attributes]] [/B] [/C] [/D] [/L] [/N]
  [/O[[:]sortorder]] [/P] [/Q] [/R] [/S] [/T[[:]timefield]] [/W] [/X] [/4]

  [drive:][path][filename]
              Specifies drive, directory, and/or files to list.

  /A          Displays files with specified attributes.
  attributes   D  Directories                R  Read-only files
               H  Hidden files               A  Files ready for archiving
               S  System files               I  Not content indexed files
               L  Reparse Points             O  Offline files
               -  Prefix meaning not
  /B          Uses bare format (no heading information or summary).
  /C          Display the thousand separator in file sizes.  This is the
              default.  Use /-C to disable display of separator.
  /D          Same as wide but files are list sorted by column.
  /L          Uses lowercase.
  /N          New long list format where filenames are on the far right.
  /O          List by files in sorted order.
  sortorder    N  By name (alphabetic)       S  By size (smallest first)
               E  By extension (alphabetic)  D  By date/time (oldest first)
               G  Group directories first    -  Prefix to reverse order
  /P          Pauses after each screenful of information.
  /Q          Display the owner of the file.
```

You need help ?

- The Windows Command for Help:

- help [<Command>]
- [<Command>] /?

- Example: help dir

Common command

Windows Command	Comments
cd	print the working directory
cd <dir>	
dir	List the files in the current working directory
tasklist	List your processes on the system
echo “a string to be echoed”	Echo a string to the terminal

Execute Next commands

- Execute: **help dir**
- Execute: **dir**
- Redirect the output of **dir /A** command to **dir_output.txt**
  - **dir /A > dir_output.txt**
- Execute: **notepad dir_output.txt**

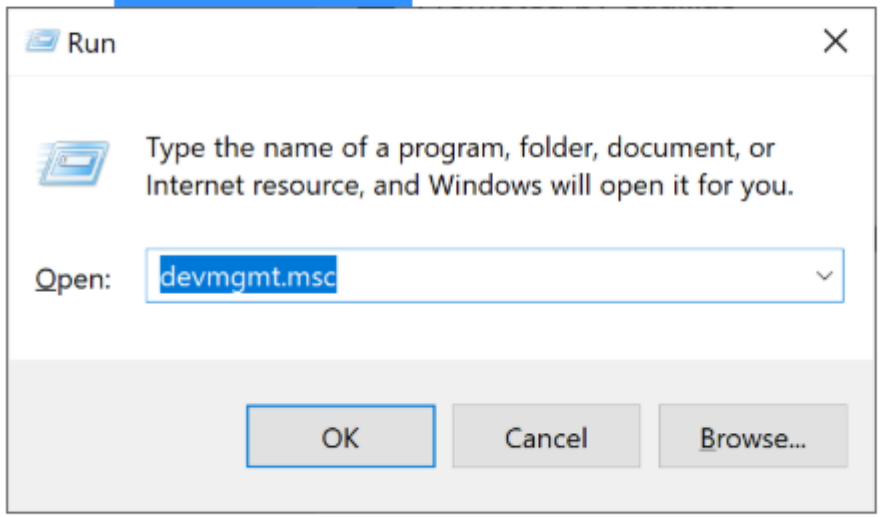
Execute Process Command

- Execute to show your processes**: tasklist | more**

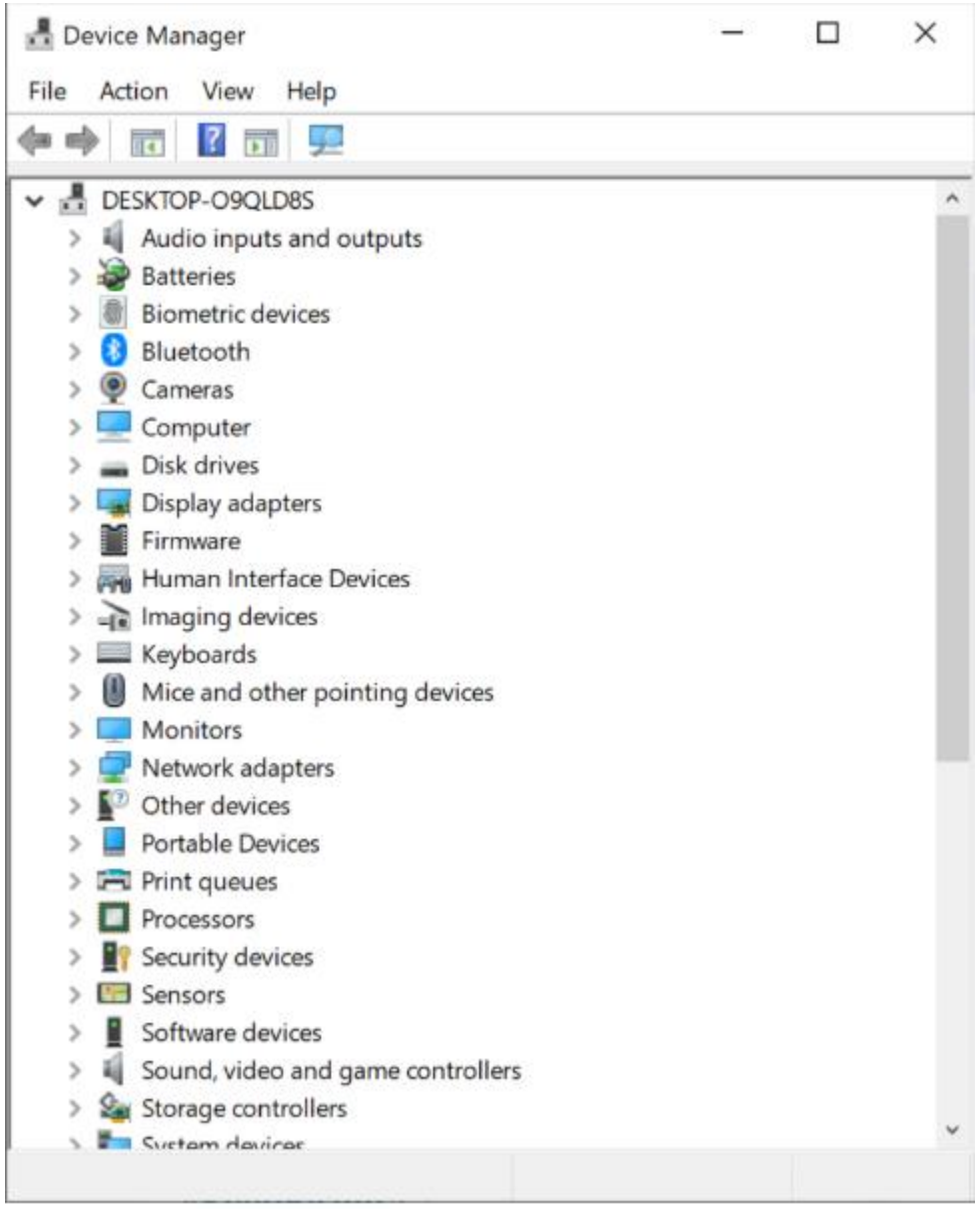
Windows Device Manager

To Start Device Manager

- Open the “Run” dialog box by pressing and holding the Windows key, then press the R key (“Run”).
- Type `devmgmt.msc`.

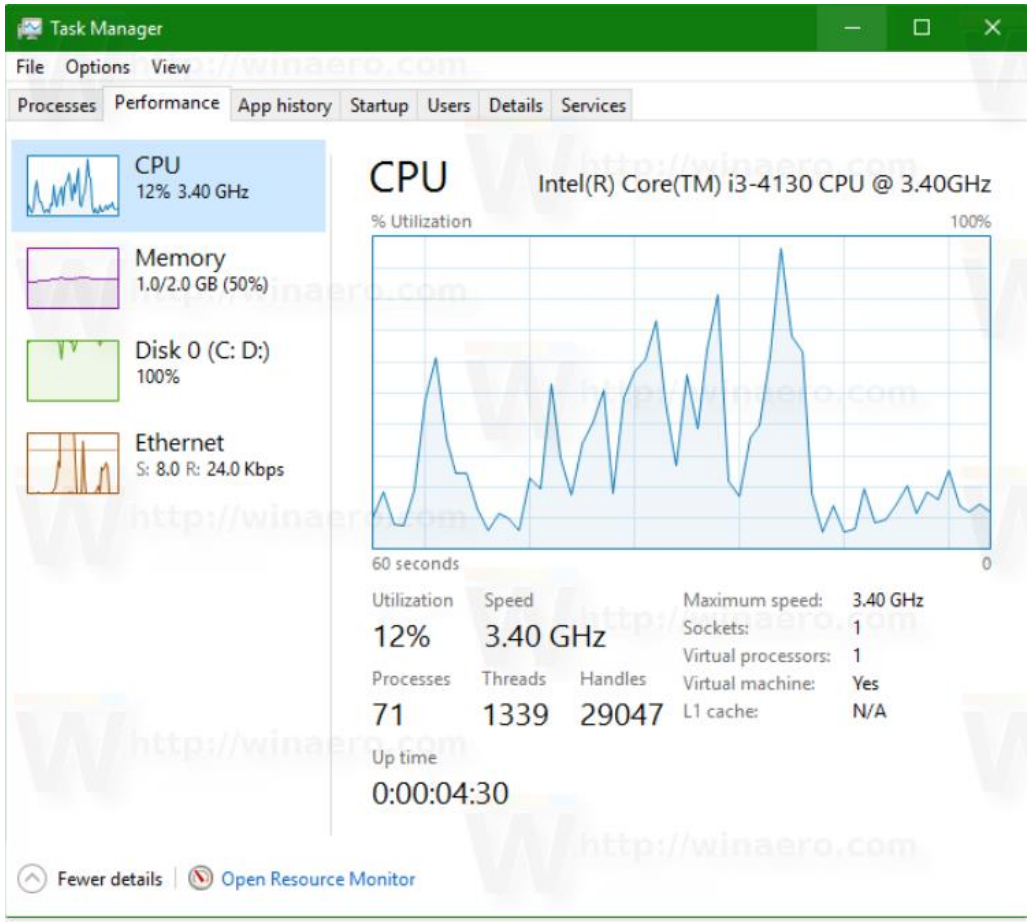


- Click OK.



Check Windows CPU Info

- Execute next command:
  - wmic cpu get caption, deviceid, name, numberofcores, maxclockspeed, status
- Or use the TaskManager tool:



Windows File System

- **You can view what file systems are mounted using either:**
  - **mountvol c: /L**
  - **If you have another drive D:**
    - **mountvol d: /L**

Environment Variables

- Display variablew:
  - Execute: **set**
  - set variable
  - set variable=string
  - set "variable=string“
  - set "variable=“
- Using Environment Variables:
  - echo %TEMP%
  - cd %TEMP%
  - echo “the env var TEMP is: %TEMP%”

Creating file and directories

- Create a new file:
  - Execute: **type nul > your_file.txt**
  - Execute: **notepad your_file.txt**

- Create a directory
  - Execute: **mkdir test**

- Remove a directory
  - Execute: **rmdir test**

File Commands

- Copy a file to other file:
  - **copy <fromfile> <tofile>**
- Move or rename a file:
  - **move <fromfile> <tofile>**
- Remove a file:
  - **del <file>**
- Create/delete a directory:
  - **mkdir <directory>**
  - **rmdir <directory>**

More Commands

- To compare 2 files (fc):
  - **fc** C:\Users\Martin\Desktop\FCsample.txt C:\Users\Martin\Desktop\FCexercise.txt
- To search a file (for example secret.doc):
  - **dir secret.doc /s /p**
- To search a text in a file:
  - **find "martin hendrikx" C:\Users\Martin\Desktop\exercise.txt**

Kill Process

- To kill a process:
  - **taskkill**

Check File Permissions in Windows

- Read the next link:
  - https://protechguides.com/how-to-check-folder-permissions-on-windows/

Compress command

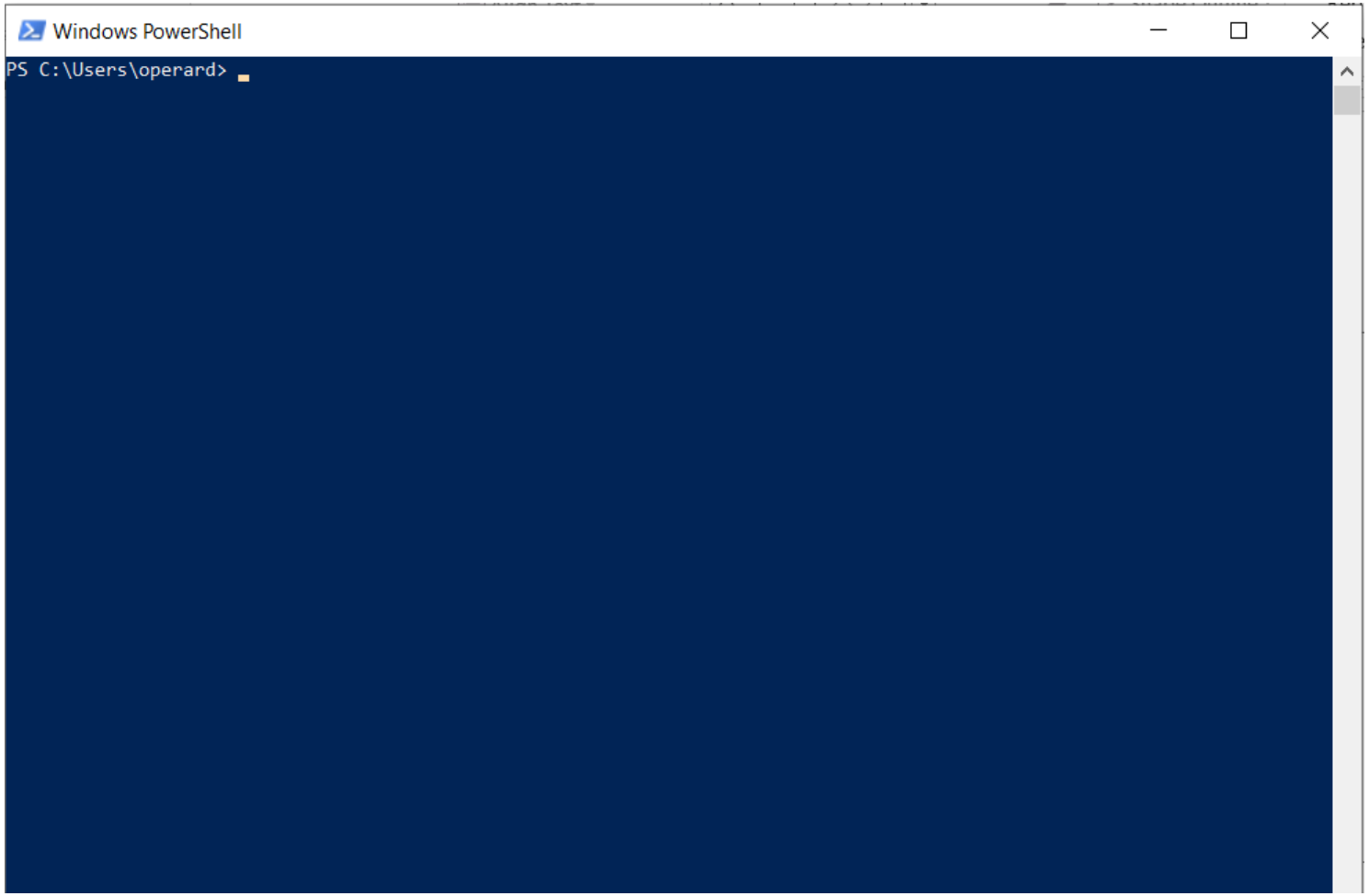
- Here is the *unzip*: http://gnuwin32.sourceforge.net/packages/unzip.htm
- There is a ZIP command as well: http://gnuwin32.sourceforge.net/packages/zip.htm
- Use winzip, zip, gzip, 7-Zip, winrar, …etc….

Viewing Files

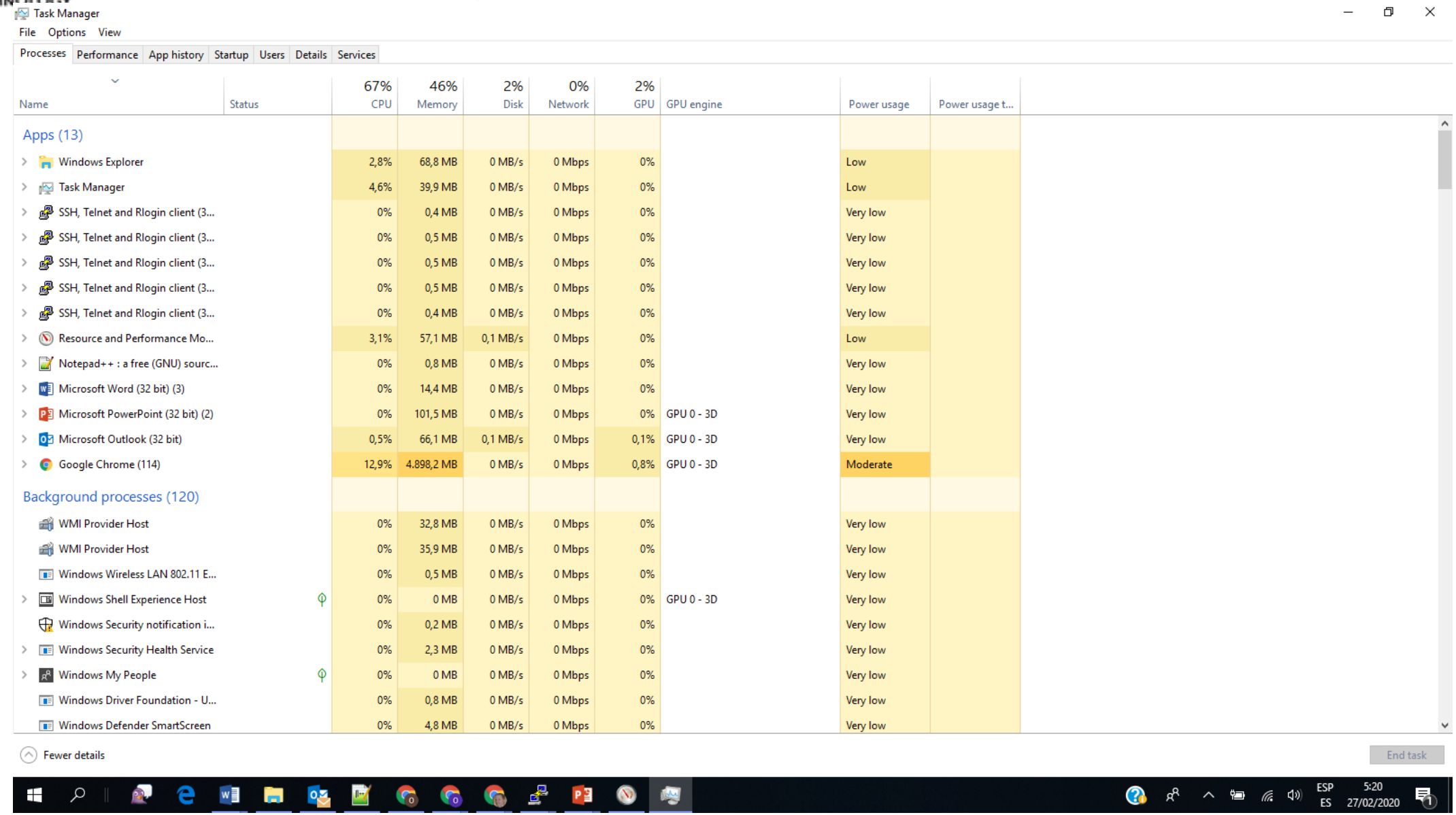
- Concatenate:
  - Use **copy** command
  - Use **type** command:
    - echo hi > a.txt
    - echo bye > b.txt
    - type a.txt b.txt > c.txt
    - type c.txt
- Display one pageat a time:
  - more
- Editors:
  - Notepad
  - Notepad++

New Command Tool in Windows 10

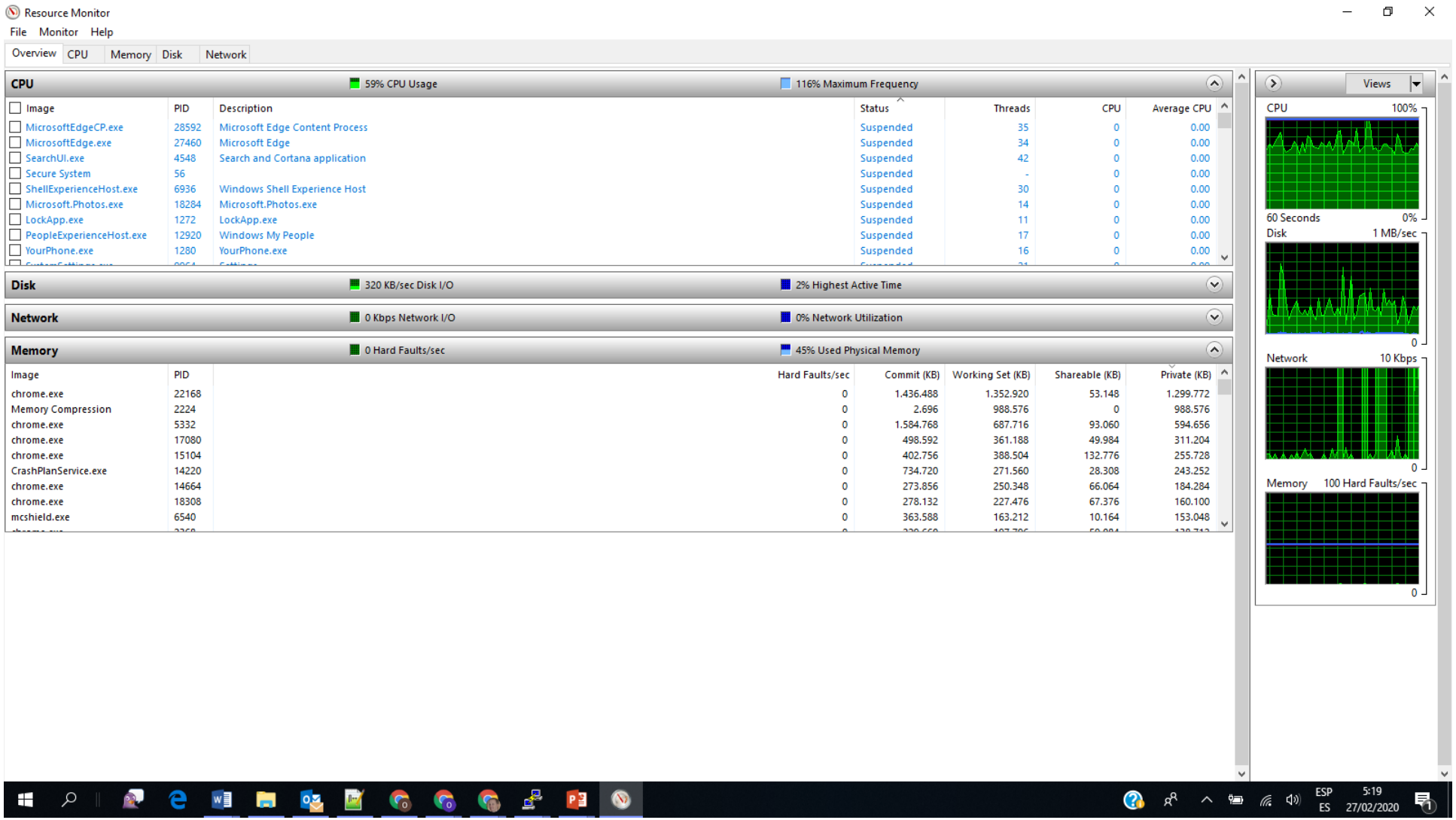
- Windows powershell



Task Manager



Execute Process Command: resmon



PSTools Suite

- https://docs.microsoft.com/en-us/sysinternals/downloads/pstools

Process Monitor

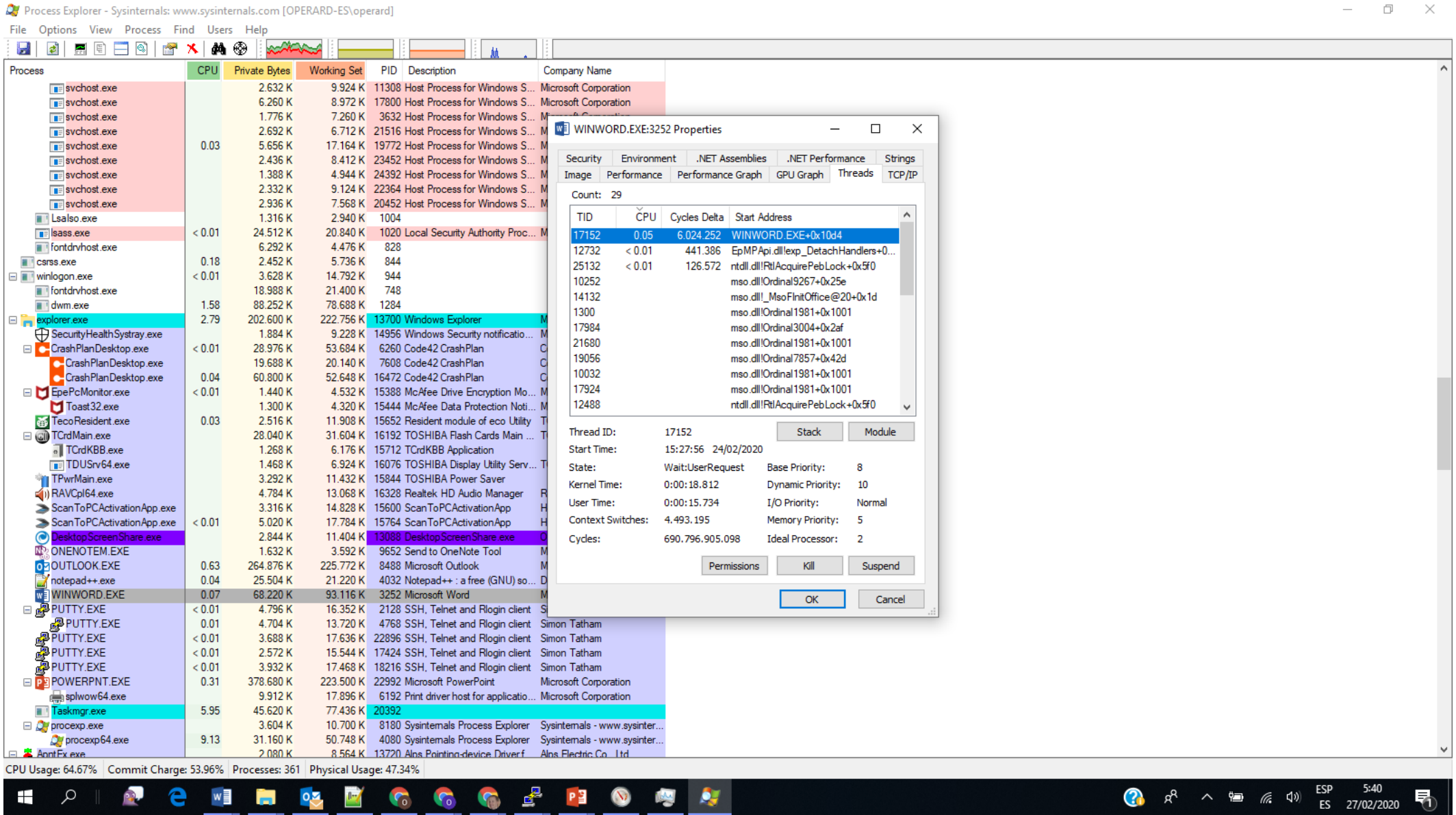
- https://docs.microsoft.com/en-us/sysinternals/downloads/procmon

ProcessExplorer

- https://docs.microsoft.com/en-us/sysinternals/downloads/process-explorer

SCHOOL OF HUMAN SCIENCES & TECHNOLOGY

Check Threads



ie
SCHOOL OF
HUMAN SCIENCES
& TECHNOLOGY

Mac OS

You need help ?

- **The Linux equivalent of HELP is `man` (manual)**
  - Use `man -k <keyword>` to find all commands with that keyword
  - Use `man <command>` to display help for that command
    - Output is presented a page at a time. Use `b` for to scroll backward, `f` or a space to scroll forward and `q` to quit

Common command

- `pwd` - **print (display) the working directory**
- `cd <dir>` - **change the current working directory to *dir***
- `ls` - **list the files in the current working directory**
- `ls -l` - **list the files in the current working directory in long format**

- `who` **or** `w`
  - List who is currently logged on to the system
- `whoami`
  - Report what user you are logged on as
- `ps`
  - List your processes on the system
- `ps aux`
  - List all the processes on the system
- `echo "A string to be echoed"`
  - Echo a string (or list of arguments) to the terminal

Who's Logged On Right Now?

- The `w` command lists all users logged on right now

```
 5:16pm  up 2 days,  8:46,  1 user,  load average: 0.00, 0.00, 0.00
USER     TTY      FROM              LOGIN@   IDLE   JCPU   PCPU  WHAT
neale    ttyp0    websurfer.reston  4:28pm  1.00s  0.52s  0.18s  w
```

Execute Next commands

- Get help on the `ls` command
- Find out who else is on the system
- What is your current directory
- Redirect the output of the `ls -l /` command to `ls.output` and see what you get

Execute Process Command

- **ps -ef | more**
- **ps aux**
- **ps –e f** **NOT WORKING in MacOS**
- **top (tape “q” to quit the process)**

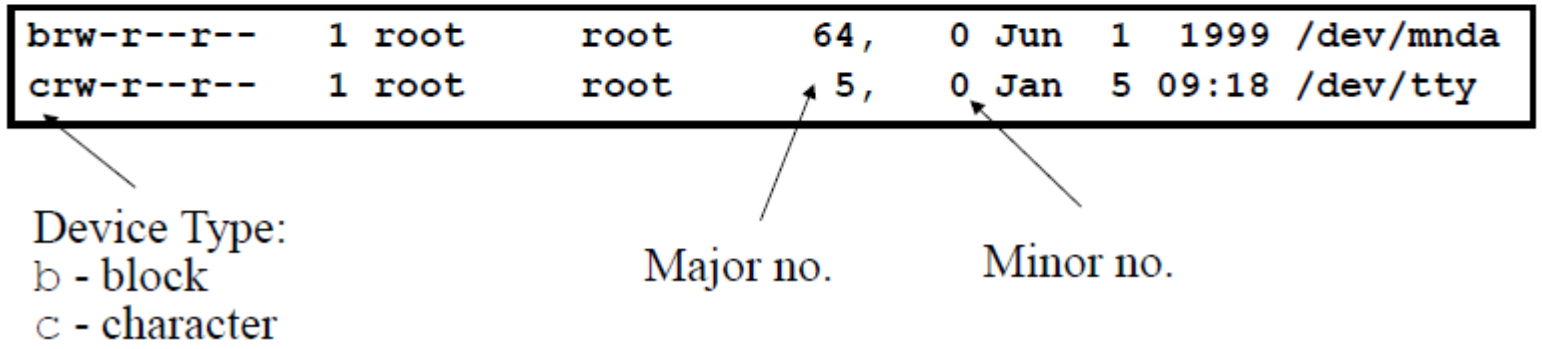
Linux Device Handling

- **Devices are the way Linux talks to the world**
- **Devices are special files in the `/dev` directory (try `ls /dev`)**

/dev/ttyx	TTY devices
/dev/hdb	IDE hard drive
/dev/hdb1	Partition 1 on the IDE hard drive
/dev/dasda	ECKD/CKD/FBA DASD
/dev/dasda1	Partition 1 on DASD
/dev/null	The null device (“hole”)
/dev/zero	An endless stream of zeroes
/dev/mouse	Mouse (not /390)

Device and Drivers

- **Each `/dev` file has a major and minor number**
  - Major defines the device type
  - Minor defines device within that type
  - Drivers register a device type



Special Files - /proc

- **Information about internal Linux processes are accessible to users via the `/proc` file system (in memory)**

/proc/cpuinfo	**CPU Information**
/proc/interrupts	**Interrupt usage**
/proc/version	**Kernel version**
/proc/modules	**Active modules**

NOT WORKING in MacOS

```
cat /proc/cpuinfo
vendor_id       : IBM/S390
# processors    : 1
bogomips per cpu: 86.83
processor 0: version = FF, identification = 045226, machine = 9672
```

Check CPU Info

- sysctl -n machdep.cpu.brand_string
- sysctl -a | grep machdep.cpu
- sysctl -a | grep machdep.cpu | grep core_count
- sysctl -a | grep machdep.cpu | grep thread_count

File System

- **You can view what file systems are mounted using either:**
  - mount
  - df –h
  - cat /etc/fstab

mount

- Mounts a file system that lives on a device to the main file tree
- Start at Root file system
  - Mount to root
  - Mount to points currently defined to root
- `/etc/fstab` used to establish boot time mounting

```
/dev/dasda1     /              ext2    defaults,errors=remount-ro 0 1
/dev/dasdb1     /bin           ext2    defaults,errors=remount-ro 0 1
/dev/dasdc1     /usr           ext2    defaults,errors=remount-ro 0 1
/dev/dasdd1     /usr/local     ext2    defaults,errors=remount-ro 0 1
/dev/dasde1     /usr/man       ext2    defaults,errors=remount-ro 0 1
/dev/dasdf1     /home          ext2    defaults,errors=remount-ro 0 1
/dev/dasdg1     swap           swap    defaults    0       0
none            /proc          proc    defaults    0       0
```

Environment Variables

- **Using Environment Variables:**
  - `echo $VAR`
  - `cd $VAR`
  - `cd $HOME`
  - `echo "You are running on $SYSTEMNAME"`
- **Displaying - use the following commands:**
  - `set` (displays local & environment variables)
  - `export`
- **Variables can be retrieved by a script or a program**

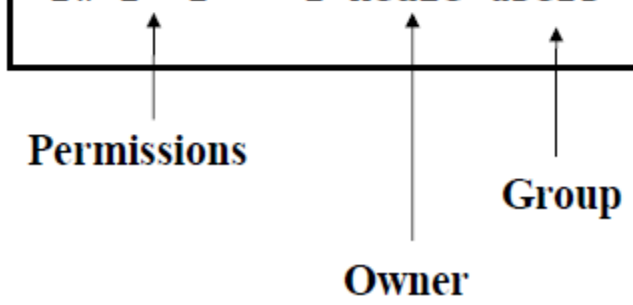
Creating file and directories

- **Files can be created in a number of ways**
  - The output of a command
  - Being edited using vi or your favorite editor
  - By using the `touch` command which creates an empty file or updates the modification and access time information of an existing file
- **Directories are created using the `mkdir` command**

File Permissions

- **The long version of a file listing (`ls -l`) will display the file permissions:**

```
-rwxrwxr-x  1 rvdheij  rvdheij      5224 Dec 30 03:22 hello
-rw-rw-r--  1 rvdheij  rvdheij       221 Dec 30 03:59 hello.c
-rw-rw-r--  1 rvdheij  rvdheij      1514 Dec 30 03:59 hello.s
drwxrwxr-x  7 rvdheij  rvdheij      1024 Dec 31 14:52 posixuft
:
-rw-r--r--  1 neale users       1039 2009-09-10 12:47 a.a
drwxr-xr-x  5 neale users       4096 2011-08-16 20:34 benchmark
drwxr-xr-x  2 neale users       4096 2009-07-30 08:55 bin
drwxr-xr-x  3 neale users       4096 2009-05-16 12:17 BINUTILS
-rw-r--r--  1 neale users       3776 2012-02-24 09:32 bluefin.cs
```



File Commands

- `cp <fromfile> <tofile>`
  - Copy from the <fromfile> to the <tofile>
- `mv <fromfile> <tofile>`
  - Move/rename the <fromfile> to the <tofile>
- `rm <file>`
  - Remove the file named <file>
- `mkdir <newdir>`
  - Make a new directory called <newdir>
- `rmdir <dir>`
  - Remove an (empty) directory

Change File Permissions

- **Use the `chmod` command to change file permissions**
  - The permissions are encoded as an octal number

User			Group			Other		
Read r	Write w	Execute x	Read r	Write w	Execute x	Read r	Write w	Execute x
400	200	100	40	20	10	4	2	1

```
chmod 0755 file  # Owner=rwx Group=r-x Other=r-x
chmod 0500 file2 # Owner=r-x Group=--- Other=---
chmod 0644 file3 # Owner=rw- Group=r-- Other=r--

chmod +x  file   # Add execute permission to file for all
chmod u-r file   # Remove read permission for owner
chmod a+w file   # Add write permission for everyone
```

More Commands

- `awk` - a file processing language that is well suited to data manipulation and retrieval of information from text files
- `chown` - sets the user ID (UID) to owner for the files and directories named by pathname arguments. This command is useful when from test to production

```
chown -R apache:httpd /usr/local/apache
```

More Commands

- `diff` - attempts to determine the minimal set of changes needed to convert a file specified by the first argument into the file specified by the second argument
- `find` - Searches a given file hierarchy specified by path, finding files that match the criteria given by expression

Search Command

- `grep` - **Searches files for one or more pattern arguments. It does plain string, basic regular expression, and extended regular expression searching**

```
find ./ -name "*.c" | xargs grep -i "fork"
```

In this example, we look for files with an extension "c" (that is, C source files). The filenames we find are passed to the xargs command which takes these names and constructs a command line of the form: `grep -i fork <file.1>…<file.n>`. This command will search the files for the occurrence of the string "fork". The "-i" flag makes the search case insensitve.

Kill Process

- **kill - sends a signal to a process or process _group**
  - You can only kill your own processes unless you are root

```
UID         PID   PPID   C STIME TTY          TIME CMD
root       6715   6692   2 14:34 ttyp0    00:00:00 sleep 10h
root       6716   6692   0 14:34 ttyp0    00:00:00 ps -ef
[root@penguinvm log]# kill 6715
[1]+  Terminated              sleep 10h
```

Replace String

- **sed - applies a set of editing subcommands contained in a script to each argument input file**

```
find ./ -name "*.c,v" | sed 's/,v//g' | xargs grep "PATH"
```

This finds all files in the current and subsequent directories with an extension of c,v. sed then strips the ,v off the results of the find command. xargs then uses the results of sed and builds a grep command which searches for occurrences of the word PATH in the C source files.

Archive command

- **`tar` - manipulates archives**
  - An archive is a single file that contains the complete contents of a set of other files; an archive preserves the directory hierarchy that contained the original files.

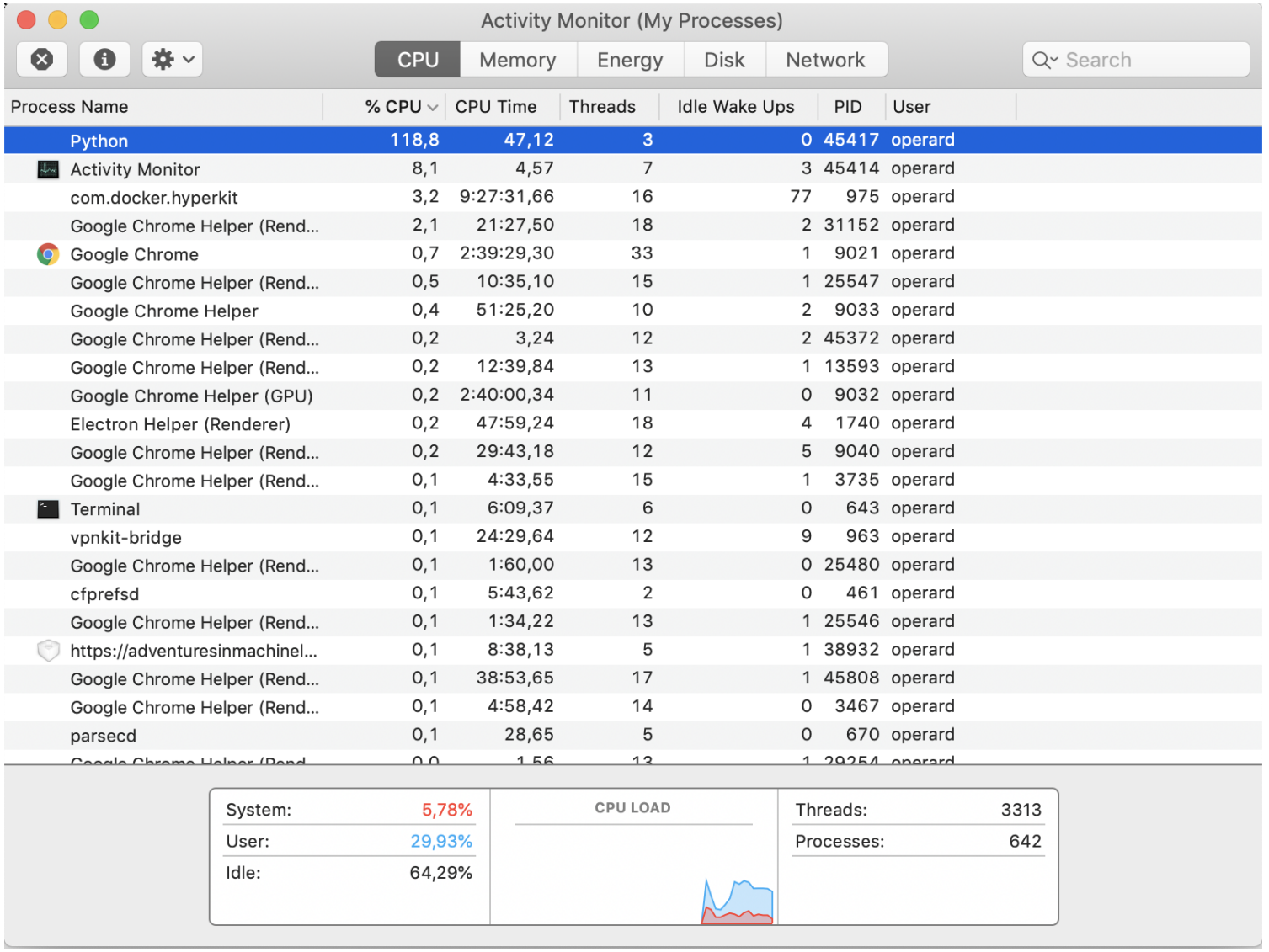
```
tar -tzf imap-4.7.tar.gz
imap-4.7/
imap-4.7/src/
imap-4.7/src/c-client/
imap-4.7/src/c-client/env.h
imap-4.7/src/c-client/fs.h
```

Viewing Files

- `cat` **"Concatenate"**
- `more` **Display one page at a time**
- `less` **Variant of** `more`
- **Editors**
  - `vi` Visual editor, the default
  - `the` XEDIT/KEDIT/ISPF clone
  - `xedit` X windows text editor
  - `emacs` Extensible, Customizable Self-Documenting Display Editor
  - `pico` Simple display-oriented text editor
  - `nedit` X windows Motif text editor

Monitoring Tool

- Use Activity Monitor



SCHOOL OF
HUMAN SCIENCES
& TECHNOLOGY