Linux Academy

Study Guide

Linux Professional Institute Certification Level 1 Exam 1

Contents

Prerequisites 1

Linux - CentOS 6, CentOS7 (or other SysVInit or Systemd Distribution) 1

Topic 101 - System Architecture 1

101.1 Determine and Configure Hardware Settings (Filesystems and Device Files) 1

101.1 Determine and Configure Hardware Settings (Tools and Utilities to Explore System Devices) 2

101.2 Boot the System (Boot Process - sysvinit - Power On to System Prompt) 4

101.2 Boot the System (Boot Process - systemd and upstart - Power On to System Prompt) 6

101.3 Change Runlevels/Boot Targets and Shutdown or Reboot the System (systemd, sysvinit and upstart) 9

Topic 102 - Linux Installation and Package Management 10

102.1 Design Hard Disk Layout (Layout and Installation Example) 10

102.1 Design Hard Disk Layout (LVM Basics and Swap) 12

102.2 Install a Boot Manager (Working with GRUB Legacy) 13

102.2 Install a Boot Manager (Working with GRUB2 Boot Loader) 14

102.3 Manage Shared Libraries (Shared Library Overview and Configuring Library Locations) 15

102.4 Use Debian Package Management (Repositories and Package Managers) 17

102.4 Use Debian Package Management (Using the dpkg Utility) 19

102.5 Use RPM and YUM Package Management (YUM Package Management) 20

102.5 Use RPM and YUM Package Management (Using the RPM Utility) 21

Topic 103 - GNU and Unix Commands 24

103.1 Work on the Command Line (Shells, Sourcing and Special Characters) 24

103.1 Work on the Command Line (Environment Variables - Displaying, Setting and Using) 27

103.2 Process Text Streams Using Filters (Textutils Package Commands - sort, nl, wc, expand, cut, paste, join, uniq, head, and tail) 29

103.2 Process Text Streams Using Filters (Textutils Package Commands - split, cat, od, pr, fmt, tr, sed, more, and less) 31

103.3 Perform Basic File Management (Commands and Terms to Know - ls, cd, pwd, mkdir, file, globbing, touch, stat, cp, and mv) 34

103.3 Perform Basic File Management (Commands and Terms to Know - dd, rmdir, rm, and find) 38

103.3 Perform Basic File Management (Archiving and Compression Utilities) 40

103.4 Use Streams, Pipes and Redirects 42

103.5 Create, Monitor and Kill Processes (Terms and Utilities - PID, ps, pstree, free, uptime, signals, kill, killall, pkill, pgrep) 44

103.5 Create, Monitor and Kill Processes (Terms and Utilities - jobs, bg, fg, &, priority, nohup, and screen) 47

103.6 Modify Process Execution Priorities (Using nice, renice, and top) 48

103.7 Search Text Files Using Regular Expressions (Using grep, egrep, fgrep, sed, and regex) 50

103.8 Perform Basic File Editing Operations Using VI 52

Topic 104 - Devices, Linux Filesystems, and the Filesystem Hierarchy Standard 56

104.1 Create Partitions and Filesystems (Using Partitioning Tools - fdisk, gdisk, and parted) 56

104.1 Create Partitions and Filesystems (Filesystem Types and Creating Them on Partitions) 59

104.2 Maintain the Integrity of Filesystems 62

104.3 Control Mounting and Unmounting of Filesystems (Mounting Filesystems Manually and Automatically) 64

104.4 Manage Disk Quotas 66

104.5 Manage File Permissions and Ownership 68

104.6 Create and Change Hard and Symbolic Links 72

104.7 Find System Files and Place Files in the Correct Location (The Filesystem Hierarchy Standard) 73

104.7 Find System Files and Place Files in the Correct Location (Using find, locate, updatedb, /etc/updatedb.conf, whereis, which, and type) 75

APPENDIX A - Configuration File Samples 77

Prerequisites

Linux - CentOS 6, CentOS7 (or other SysVInit or Systemd Distribution)

Topic 101 - System Architecture

101.1 Determine and Configure Hardware Settings (Filesystems and Device Files)

- Everything is a file
  - Linux treats everything on a system (hardware devices included) as a file
- `procfs`
  - Information about system hardware and the state of the system is contained in this 'pseudofilesystem'
    - NOTE: a 'pseudofilesystem' is something that looks like a system device but is not really associated with real hardware, it just exposes kernel information in human readable format
  - `/proc`
    - Usually where the `procfs` filesystem is mounted
    - Within this directory, you will find directories that correspond in label (number) to the associated PID (Process ID) of running processes
    - Additionally, other files and directories that contain information about the system or hardware in it are:
      - `/proc/mounts` - a file (or link to another file) that contains information on all the filesystems that are mounted locally
      - `/proc/interrupts` - shows information about the interrupts in use in the system and what they are associated with (the hardware that is using them)
      - `/proc/ioports` - one or more addresses that identify a device, and the kernel module associated with them
      - `/proc/dma` - (Direct Memory Access), values used by hardware to access system memory directly (i.e. without involving the CPU)
      - `/proc/usb` - USB device IDs and kernel modules associated with them

        - `/proc/pci` - PCI device IDs and kernel modules associated with them
- `sysfs`
    - Another 'pseudofilesystem' that can contain information about system hardware (like procfs)
    - Was designed to address some of the problems with the procfs method (unstructured data and the fact that both hardware and process information was consolidated in the structure)
    - `/sys`
        - Where the 'sysfs' filesystem is mounted
        - Within this directory, you will directories that correspond to system hardware and kernel modules associated with the system (physical or virtual)
        - NOTE: process data (PID) does NOT exist in this method
- `udev`
    - Device manager for the kernel
    - `/dev`
        - Directory containing device data (using the information from `/sys` or sysfs)
        - Devices added or removed on a live system will engage the `udev` system which will detect and configure it and then make it available for use (often called 'hotplug' or 'hotswap')
    - D-Bus
        - Related to the `udev` system
        - Allows desktop applications to send messages to other applications and/or receive messages from the Linux kernel
            - i.e. when mounting a CD-ROM, the 'udev' system uses this to notify the running window manager to display an icon on the desktop

101.1 Determine and Configure Hardware Settings (Tools and Utilities to Explore System Devices)

- `lsmod`
    - Lists the kernel modules (drivers) that are loaded into memory (along with their dependencies)
    - will not load a driver if missing
- `lscpu`
    - Displays a summary of the CPU and its features/configuration

  - `-a` (or `--all`) - will display offline and online CPUS
  - `-b` (or `--online`) - only online CPU information
  - `-c` (or `--offline`) - only offline CPU information
  - `-e` (`--extended`) `=[list]` - display the information in a more readable format
    - `[list]` - column(s) to include (default is all - examples include 'cpu' or 'node')
  - `-p` (or `--parse`) - provides the information displayed in a comma delimited form (that can be logged and used by other applications for reporting)
- `lspci`
  - Displays information about the PCI bus and associated devices that are 'plugged in' there
  - In the event that the kernel does not recognize (or support by default) a device, it will show up here and contain information (slot, device ID, memory range, etc) that can be used to find device drivers/modules
  - `-v` (or `-vv` or `-vvv`) - Controls the level of verbosity of the information displayed
  - `-m` (or `-mm`) - Displays the PCI information in a more “easily parseable” format
    - `-vmm` - nicely formatted listing of all PCI hardware on the system
  - `-t` - display a tree view of the PCI devices on the system (JUST the device slot/port)
    - `-tvmm` - associate the device names with the IDs in a tree view
- `lsscsi`
  - Displays information on any SCSI devices detected
    - NOTE: modern distributions will display SATA devices detected via this command
  - `-c` (or `--classic`) - output is the same as running `cat /proc/scsi/scsi`
  - `-d` (or `--device`) - provides additional 'major/minor' device numbers behind each detected device
  - `-g` (or `--generic`) - generic SCSI device file name (sg)
  - `-l` (or `--long`) - addtional information for each device
  - `-s` (or `--size`) - print the disk size in a more readable format
  - `-v` (or `-vv` or `-vvv`) - controls the level of detail (verbosity)
- `lsdev`
  - Shows all the devices that are recognized by the running system kernel

- `lsraid`
  - Displays any RAID (Redundant Array of Inexpensive Disks) on your system
  - NOTE: not installed on most systems, unavailable outside of some RAID drivers/configuration software
- `lsusb`
  - Displays USB device IDs and general information about detected devices
  - `-s [bus][:][device #]` - shows the specified bus and device number information (optional)
  - `-d [vendor]:[product #]` - show only devices with the indicated vendor and product ID (hexadecimal format)
  - `-D [device]` - skip scanning the `/dev/bus/usb` directory, display only info about the device indicated
    - NOTE: only root can issue this parameter, the format requested is path based, not device ID (i.e. `/dev/bus/usb/001/001`)
  - `-t` - display the hierarchy in a tree view
  - `-v` - verbose device output
- `lsblk`
  - Display block devices (disks) that are attached to the running system
  - `-a` (or `--all`) - list empty devices as well (off by default)
  - `-d` (or `--nodeps`) - print only top level device information
    - For example - `lsblk --nodeps /dev/sdb` (only print device information on the `/dev/sdb` disk)
  - `-e` (`--exclude`) `[list]` - exclude the indicated devices (one or more, comma separated)
  - `-f` (or `--fs`) - include information on filesystems
  - `-i` (or `--ascii`) - use ASCII characters for tree view
  - `-l` (or `--list`) - output list format
  - `-t` - tree view of disk devices and partitions

101.2 Boot the System (Boot Process - sysvinit - Power On to System Prompt)

- `sysvinit`

- Older (but valid for exam) system and service management
- init script driven

- System boot order (notes on differences between sysvinit, systemd and upstart are provided as appropriate)
- General boot order is as follows:

1. Power on
2. BIOS loads (NOTE: modern systems this also includes EFI)
3. BIOS find and hands off to the primary (or chosen) disks 'boot sector'
4. Boot sector provides the MBR (Master Boot Record), within the first 512 bytes of the active (or first, depending on installation) drive (can be a floppy, USB or hard drive of any kind)
5. Boot loader is executed
6. LILO/GRUB/GRUB2 boot loader begins
7. USER INPUT STEP - depending on boot loader configuration, the user may be able to choose from a menu of potential boot types/kernel versions or allow the default to proceed
8. Linux kernel is read and executed
9. Device initialization, module loading, and the initial RAM disk (initrd) is loaded
10. Root filesystem is mounted
11. The init program is loaded (and becomes the first PID - `/sbin/init`)
12. `/etc/inittab` is read and the appropriate runlevel script(s) are run
    - Debian/Ubuntu - `/etc/init.d/rc#`
    - Red Hat/CentOS - `/etc/rc.d/init.d/rc.sysinit`
13. Modules indicated within the init scripts are loaded
14. Root filesystem is checked
15. Remaining local filesystems are mounted
16. Network devices are started
17. Remote filesystems are mounted (if configured)
18. init process re-reads `/etc/inittab` and changes to the default runlevel indicated and executes the appropriate scripts
19. runlevel scripts executed in numeric order (#service for the runlevel indicated in `/etc/inittab`)

20. tty sessions are loaded as listed in `/etc/inittab`
21. Login prompt is displayed, system is ready for login

- sysvinit runlevels
    - 0 - HALT (shutdown)
    - 1 - single user
    - 2 - multi-user (no networking or remote filesystems)
    - 3 - full multi-user (with network)
    - 4 - unused
    - 5 - X11 (graphical desktop mode)
    - 6 - reboot

101.2 Boot the System (Boot Process - systemd and upstart - Power On to System Prompt)

- Systemd
    - Default on most modern Linux distributions
    - Designed to replace the sysvinit method of managing system initialization and start up
    - More efficient and allows parallel operation
    - Fewer dependencies
    - Service prioritization and precedence
    - Reduces start up time
    - Concept of 'units' replacing the sysvinit init scripts
        - Service is 'name.service' rather than just an init script name
    - Major components to be aware of
        - `systemd` - manager for the services and systems
        - `systemctl` - primary command for controlling service start/stop/restart/status
        - `systemd-analyze` - display performance information on system bootup, allows for trace/debugging information
        - `logind` - replaces consolekit, supports X Windows managers
        - `consoled` - console daemon, replaces virtual terminals

    - `journald` - logging system, uses binary logs (although it can be replaced by other logging daemons)
    - `networkd` - networking support daemon
- NOTE: runlevels in systemd systems are indicated in 'component groups (cgroup)'
    - Allows for dependencies and tagging, saves on memory and resource allocation
- `/usr/lib/systemd/system`
    - the 'units' installed
- `/etc/systemd/system` - units that have priority over other units and are controlled by sysadmin
- `/run/systemd/system` - units created at runtime and may have priority over non-runtime (or installed) units
- Service and extensions
    - .service - system service
    - .swap - swap device or file
    - .socket - IPC socket
    - .target - unit (one or a group of)
    - .snapshot - saved state instance of the systemd manager
    - .slice - units grouped together in a hierarchy used to manage processes
    - .timer - timer
    - .mount - mount point on the file system (local or remote)
    - .automount - automount point on the filesystem
    - .scope - externally created process
    - .path - file or directory on file system
    - .device - device in use by the system kernel
- systemd runlevels
    - 0 - poweroff.target - shutdown the system
    - 1 - rescue.target - single user/rescue shell
    - 2 - multi-user.target - non-graphical, but full network, multi-user
    - 3 - multi-user.target - non-graphical, but full network, multi-user

    - 4 - multi-user.target - non-graphical, but full network, multi-user
    - 5 - graphical.target - full graphical desktop, multi-user
    - 6 - reboot.target - reboot
- Service target dependencies
    - Method of indicating dependencies between units or groups of units on the system
    - Requires
        - Only after the requirement is met, will any other 'wanted' units be started
    - Wants
        - Only executed upon completion of 'requires' units
    - NOTE: see `/usr/lib/systemd/system/graphical.target` for example
- Boot process changes
    - `/etc/inittab` is not read (and is no longer in use)
    - Instead, a 'default.target' is started (which is linked to the current 'default' target - multi-user, graphical, etc)
    - A tree of 'requires' and 'wants' is then followed throughout the process
- `runlevel`
    - Displays the current runlevel (and the previous, if available)
- `systemctl get-default`
    - Displays the current default runlevel target
- `systemctl set-default [new.target]`
    - Will set the default runlevel target to the indicated value
    - Creates a link from 'default.target' to the indicated target
        - Manual method example - `ln -sf /usr/lib/systemd/system/multi-user.target /etc/systemd/system/default.target`
- `systemctl list-units --type=target`
    - List all the active system targets
- `systemctl isolate [runlevel.target]`
    - Allows you to set the runlevel of the system without changing the default
    - For example - `systemctl isolate graphical.target` would change the system to

the full desktop graphical mode

- `upstart`
  - An asynchronous boot and shutdown process
  - Designed to be easy to implement as it maintains sysinit compatibility (and can run sysvinit scripts unchanged)
  - Considered easily extensible

101.3 Change Runlevels/Boot Targets and Shutdown or Reboot the System (systemd, sysvinit and upstart)

- `sysvinit`
  - `runlevel`
    - Displays the current runlevel (and the previous, if available)
  - `init`
    - Can be run to change the system runlevel
      - For example - `init 5` would change the system runlevel from whatever it was when run to the runlevel 5 or graphical mode, if configured
  - `telinit`
    - Same as the `init` command (note that this is a legacy command not typically used)
  - default target
    - Changed in the `/etc/inittab` file
      - For example - `id:3:initdefault` would set the default runlevel value to 3, or multi-user target
  - Rebooting or shutting down the system
    - `reboot` - will reboot the system
    - `shutdown` - will reboot OR halt the system
      - `-h` - halt the system (shut it down)
      - `-r` - reboot the system
      - `-P` - power off (if ACPI is available)
      - `-c` - cancel shutdown
      - `-k [message]` - broadcasts a 'wall' message to logged in users

        - `[option]` - indicates when the shutdown takes place (in seconds, minutes, specific time or 'now')
    - `halt`
        - `-f` (or `--force`) - does not shutdown, only halts
        - `-p` (or `--poweroff`) - power off after shutdown (if ACPI is available)
        - `-w` (or `--wtmp-only`) - only LOGS the shutdown but does not perform the shutdown or reboot
        - `--verbose` - displays more information for troubleshooting
- systemd
    - In addition to any of the sysvinit methods, you can use `systemctl`
    - `systemctl halt` - halt, do not power off
    - `systemctl poweroff` - halt and power off (if ACPI is available)
    - `systemctl reboot` - halt and reboot
- `upstart`
    - Supports all sysvinit methods but not systemd methods
- `wall`
    - Allows you to broadcast a message to all logged in users, limited to 20 lines of text
    - For example - `wall "This system is going offline in five minutes - wrap it up and save your files"`
    - NOTE: this message will appear on every terminal, often overwriting or interrupting terminal text

Topic 102 - Linux Installation and Package Management

102.1 Design Hard Disk Layout (Layout and Installation Example)

- Single disk installation
    - Linux installed on a single physical disk, consisting of one or more partitions
- Multi-disk installation

    - Linux installed on multiple physical disks, each consisting of one or more partitions
- Device naming conventions (for local SATA or SCSI disks)
    - `/dev/sda` - first physical disk (drive)
    - `/dev/sda2` - first physical disk, second partition
- Other device types
    - hda (IDE drives)
        - `/dev/hdc2` - third IDE drive, second partition
    - scd (CDROM)
        - `/dev/scd0` (first CDROM drive)
            - CDROM does not have partitions
- Linux filesystem layout
    - Everything is a file in Linux
    - Although you have many files/folders, any folder (with a few exceptions) can be mounted on any drive/partition
- Common filesystem mounting scheme (folders)
    - `/` - the 'root' filesystem is mounted on a device/partition and contains all other folders (some of which are commonly mounted on other partitions or drives)
    - `/var` - log files, shared files/directories, runtime information, binary data files
    - `/home` - contains user home directories
    - `/opt` - generally third party applications are installed here
    - `/boot` - the boot loader configuration and kernel files are contained here (in older systems, the first partition meant to be in the first 512mb of space on the first drive)
    - `swap` - dedicated partition for memory swap
- Filesystem layout
    - Create your scheme for your specific purpose
    - Shared development system with constrained disk space; for example, you may want to create a separate partition for the `/home` directory so that users copying files or installing software/builds do not cause the root partition to fill up (which can cause issues of stability)

102.1 Design Hard Disk Layout (LVM Basics and Swap)

- LVM
  - Logical Volume Manager
  - Allows the creation of 'groups' of disks or partitions that can be assembled into a single (or multiple) filesystems
  - PV
    - Physical Volume - the basic starting point or unit of storage
    - Corresponds to a disk or partition on the system
      - Either local to the system (like `/dev/sdb1`) or a SAN (storage area network) block device
  - VG
    - Volume Group
    - Combination of one or more PVs to create a pool of available storage
      - PE - Physical Extants
      - Pieces that allow easy allocation of storage from a PV
  - LV
    - Logical Volume
    - Each volume group can be divided into one or more Logical Volumes, each can be formatted with a particular filesystem type and then mounted on the system
  - Advantages of LVM
    - Flexibility - you can increase the size of a logical volume simply by resizing it, pulling more space from the volume group, if the volume group is out, you can allocate additional physical volumes to the volume group, making it available to the logical volume(s)
    - Snapshots - 'point in time backup' of the state of any logical volume that can be used for backup, restore, testing, movement, etc. without affecting the 'live' logical volume
- Swap (virtual memory)
  - Partitions - a dedicated partition formatted specially as swap space (usually during install, but can be done anytime you add a disk or partition)
  - Files - specially created file that is used on a filesystem as swap space for the system

102.2 Install a Boot Manager (Working with GRUB Legacy)

- GRUB (legacy)
  - Grand Unified Boot Loader
  - `/boot`
    - The 'boot volume' for GRUB
    - The kernel (and related files) are stored here
      - Kernel config file
      - Kernel initrd (ramdisk) file
      - Default boot message
      - Module 'symvers' - version file containing all kernel module symbols
      - System map file
      - The kernel itself (vmlinuz)
  - `/boot/grub`
    - Legacy GRUB configuration directory
      - Various 'stages' for the boot process
      - `menu.lst`/`grub.conf`
        - GRUB options and builds the menu of choices displayed on boot
        - Key settings:
          - `default=[#]` - determines the menu option that will be started if nothing else is chosen
          - `timeout=[#]` - the amount of time, in seconds, you have to choose another option
        - sections (menu list entries)
          - `title [description]` - a text based description of the menu entry
          - `root [(hd#,#)]` - the hard drive device number and partition of the root drive (where the kernel is)
          - `kernel [/boot/vmlinuz-# options root=LABEL=/ options]` - the path to the kernel, read options on the kernel, its label and other boot options passed to the kernel on system start
          - `initrd [/boot/initrd#.img]` - path to the initial RAM disk image

for the chosen menu entry

- Changes to the menu list are applied on the next system boot and can be seen

- Editing the boot menu entry with 'e' on the menu list during boot loader display
- GRUB command line can be accessed with 'c' at the menu
- Reinstalling GRUB
  - `grub-install [device]`
    - Device can be identified by drive path (`/dev/hda` or `/dev/hd0`)
- `grub`
  - Application that will show the GRUB prompt
  - Allows you to explore the environment (finding the primary device path and number)

102.2 Install a Boot Manager (Working with GRUB2 Boot Loader)

- GRUB2
  - Grand Unified Boot Loader (improved)
  - `/boot`
    - The 'boot' volume for GRUB2
    - The kernel (and related files) are stored here
      - Kernel config file
      - Kernel initrd (ramdisk) file
      - Default boot message
      - Module 'symvers' - version file containing all kernel module symbols
      - System map file
      - The kernel itself (vmlinuz)
  - `/boot/grub`
    - Compatibility with GRUB directory, sometimes will contain the splash image for the boot menu
  - `/boot/grub2`
    - `grub.cfg`

        - Generated menu file, containing a more complex configuration that is a combination of a number of files when the `grub2-mkconfig` utility is used
        - Editing changes here will be lost during the next run of the `grub2-mkconfig` utility
    - `/boot/grub2/efi`
        - EFI extensions for GRUB2 on modern distributions (out of scope for exam)
    - `/boot/grub2/fonts`
        - Default (and potentially custom) font(s) for the GRUB2 menu
    - `/boot/grub2/themes`
        - Themes for the GRUB2 graphical menu
    - `/etc/grub.d`
        - Numbered files that, when the `grub2-mkconfig` utility is run, are all concatenated (in numeric order) to make the `/boot/grub2/grub.cfg` (menu) file
    - `/etc/default/grub`
        - File containing the GRUB2 specific configurations (timeouts, default option, options for the command line)
            - NOTE: these options are specific to GRUB2 behavior in general and do not apply to any menu entry
    - Editing the boot menu entry with 'e' on the menu list during boot loader display
    - Grub command line can be accessed with 'c' at the menu
    - `grub2-mkconfig` (sometimes `grub-mkconfig`)
        - Used to generate the GRUB configuration file (menu)
        - `-o [/path/grub.cfg]` - output to the specified location and file
    - `grub2-install`
        - Installs GRUB2 to the indicated location (drive master boot record or full path indicated on drive)
        - `[[--boot-directory=/path/to/install] [/dev/drive]]`

102.3 Manage Shared Libraries (Shared Library Overview and Configuring Library Locations)

- Shared libraries

    - Reusable functions that other applications can use by 'linking' to them within the library file
    - Any shared library filename has a '.so' file extension (may also have version information after the .so)
        - The 'so' in the extension stands for 'shared object'
    - `/lib`
    - `/usr/lib/*` (`/usr/lib64/*` - for 64 bit libraries)
    - `/usr/local/lib/*`
    - `/usr/share/*`
        - Common locations for system shared libraries
    - Soft links
        - Many times, specific library versions are linked to a more generic name
        - For example - `libstdc++.so > libstdc++.so.6.0.19`
- Linking
    - Static linking - the application contains a full copy of the library that is used
        - Advantages - version control (the application will have the exact library version it expects with known interfaces)
        - Disadvantages - size (increases the size of the application to be distributed since it includes a full copy of any statically linked library), upgrades (the application needs to be recompiled and relinked anytime the library required is updated)
    - Dynamic linking - the application uses the library externally using 'stubs', the library is installed on the system (OS) itself, but not installed with the application
        - Advantages - size (the application is smaller because the libraries are separate from it), upgrades (no recompile or relinking needs to be done when/if the library is updated)
- `ld.so`
    - Any time an application needs to use a shared library, the `ld.so` service is called
    - Also known as 'dynamic linker'
- `ldd [filename]`
    - Shows a list of all libraries the indicated dynamic executable application requires (and if they are present)
- `ldconfig`
    - Configures the dynamic linker run-time bindings (creates links and caches the most recent

shared libraries found)

- Looks as specified on the command line
- Looks as specified in the `ld.so.conf` file

- Generates the `/etc/ld.so.cache` file, a binary file listing libraries on the system as detected (used by `ld.so` as needed)
- Commonly run after system updates (automatically or manually) or after third party application installations (that may have added libraries)

- `/etc/ld.so.conf`
  - Standard list of library file locations that are listed in the file
  - NOTE: if a line exists, 'include ld.so.conf.d/*.conf', it will also read all the files as indicated, looking for libraries in those locations as well (newer method)
  - Format of entries are simply the path to the directory that the linker needs to scan
    - For example - `/usr/lib/qt-3.3/lib`
- `LD_LIBRARY_PATH`
  - An environment variable that can be user-specific or system wide, that specifies a path (in typical format, colon (:) delimited), that the `ld.so` and `ldconfig` application(s) can look for libraries
  - For example - `export LD_LIBRARY_PATH=/opt/myapp/lib:/opt/anotherapp/lib`
    - Would create a session environment variable when executed on the command line that would allow `ldconfig` to find libraries in the indicated paths (in addition to the values in `/etc/ld.so.conf`)

102.4 Use Debian Package Management (Repositories and Package Managers)

- Repositories
  - Remote locations that you can download and install packages from using any of the package management tools below
  - `/etc/apt/sources.list`
    - File containing one or more remote repository locations for package download and installation
  - `/etc/apt/sources.list.d`
    - Individual files that define one or more remote repository locations for package download

and installation

- Each may contain lines for both packages (binaries) and another for package sources
- For example - `deb http://archive.canonical.com/ubuntu [nickname] partner`
  - This will add the 'partner' remote repository for the indicated (nickname) Ubuntu distribution, see the `apt-get update` command for pulling the package list into the local system cache

- `apt`
  - Stands for Advanced Package Tool
  - A set of tools (apt-*) for working with full Debian packages (and their dependencies)
- `apt-get`
  - Package manager that can install, reinstall or remove packages and all their dependencies at the same time
  - `update` - read remote repository package listing (refresh)
  - `upgrade` - upgrade the existing system and installed software with any new versions reported as available in the configured repositories
  - `install [packagename]` - install the indicated package (not a .deb file, a general package name) as well as all its dependencies (if available and they do not cause conflicts)
  - `dist-upgrade` - upgrades all packages on the system at one time to the next available version of Ubuntu/Debian in the repositories
  - `remove [packagename]` - removes the package's files but any configuration files and directories will be left alone
  - `purge [packagename]` - removes the package's files AND any configuration files and directories associated with it
  - `-d` (or `--download-only`) `[packagename]` - will only download the .deb package and place it in `/var/cache/apt/archives`
- `apt-cache`
  - Allows you to interact with the cache of available packages from all configured repositories (once updated via the `apt-get update` command above)
  - `search [value]` - search for packages matching the indicated value
  - `show [packagename]` - show available information about the indicated package
  - `showpkg [packagename]` - additional technical information about the package (dependencies, services, etc)

- `aptitude`
  - similar to `apt` (in fact, it uses it in the background), has a graphical front end
- `synaptic`
  - similar to `apt` (in fact, it uses it in the background), has a graphical front end

102.4 Use Debian Package Management (Using the dpkg Utility)

- Debian packages (*.deb)
  - Contains an application, default configuration files, documentation and system instructions on how/where all the pieces of the application are to be installed on the system as well as a full list of dependencies
- `dpkg`
  - Application responsible for the installation, removal and querying of Debian packages on the system
  - `--info [packagename]` - will display the package contents
  - `--status [packagename]` - an abbreviated display of the `--info` option
  - `-i (or --install) [packagename]` - will install the indicated package name on the system
    - NOTE: if there are missing dependencies, the installation will fail, dependencies should either be installed first or at the same time in the same command (optionally, use one of the package management tools below)
  - `-r` (or `--remove`) `[packagename]` - removes the package's files but any configuration files and directories will be left alone
  - `-P` (or `--purge`) `[packagename]` - removes the package's files AND any configuration files and directories associated with it
  - `-L` (or `--listfile`) `[packagename]` - list all files that were installed with the indicated package
  - `-S` (or `--search`) `[filename]` - search for the indicated installed file within the package database for all mentions of the indicated file or string
  - `--force-reinstreq [packagename]` - allows you to force the removal of a package marked as requiring reinstallation
  - `--force-depends [packagename]` - used with `-i` or `--install` to force the installation of a package with missing dependencies

    - `--force-conflicts [packagename]` - used with `-i` or `--install` to force the installation of a package with conflicting dependencies
        - NOTE: forcing installations in this manner may leave your system in an unusable state, be sure you know what you are doing
    - `-l` (or `--list`) `[packagename or partial name]` - list the installed package(s) that match the indicated value
- `dpkg-reconfigure [packagename]`
    - Some applications will include a tool for configuring the application during installation (mail servers, database servers, etc), if you need to rerun that configuration tool for a package, you can cause it to run with this command

102.5 Use RPM and YUM Package Management (YUM Package Management)

- Repositories
    - Remote locations that you can download and install packages from using any of the package management tools below
- `/etc/yum.conf`
    - Default yum configuration file, may contain some repository definitions
- `/etc/yum.repos.d`
    - Individual '*.repo' files containing the location and options for remote repositories
    - See Appendix A for an example of a repository file from this directory (third party repository)
        - NOTE: in the repository configuration log, the location is defined as either a 'baseurl' (direct link) to a repository or a 'metalink', which will return a list of mirror sites to use
        - NOTE: only repositories that are set to 'ENABLED=1' in the repo file will be used by the yum utility
- `/var/log/yum.log`
    - Default yum transaction log file (defined in the `/etc/yum.conf file`)
- `yum`
    - The Yellowdog Updater Modified
    - Sits on top of RPM, manages dependencies in addition to the package being installed
    - `update`/`upgrade` - will read the configured repositories and local cache, and upgrade all installed system applications (note - this does NOT do a distribution version upgrade)

    - `--enablerepo [reponame]` - enable a disabled repo for the current transaction only
    - `install [packagename]` - installs the package, along with all required dependencies
        - `--downloadonly` - when added to install, will just download the indicated package
        - `/var/cache/yum/[architecture]/[version]/base/packages` - the directory the package will be downloaded to
    - `remove [packagename]` - removes the package, does NOT remove previously installed dependencies
        - to remove unused dependencies (not necessarily related to the removal of a package, but anything not in use over time), you can execute
            - `yum autoremove`
    - `-y` - added to any yum transaction, will skip the affirmation step (ability to press Y/N before it continues)
    - Can be used directly to download a package rather than installing it by answering the 'Y/N' prompt with 'd'
        - `/var/cache/yum/[architecture]/[version]/base/packages` - the directory the package will be downloaded to
- `yumdownloader`
    - Allows you to download a package only, but provides additional functionality than doing so through yum (see above)
    - `--source` - download only source RPM
    - `--urls` - display the URL of the files without downloading
    - `--destdir` - allows you to indicate the directory to store the package download
    - `--resolve` - includes any dependencies

102.5 Use RPM and YUM Package Management (Using the RPM Utility)

- RPM database
    - Database files containing the packages and associated files installed by them on the system
- `/var/lib/rpm`
    - The RPM database directory
- rpm packages

    - Will have the *.rpm extension
    - Contains source or binary application files, build instructions (if source), name of the software, version information, build date and system of build, description, dependencies, and checksums
- Source RPM
    - Source files that are needed to build and install the indicated package
- `rpm`
    - Originally, was the RedHat Package Manager
    - `--rebuilddb` - rebuilds the local RPM database in the `/var/lib/rpm directory`
        - NOTE: only root can do this (or user with `sudo` command privileges)
    - Installation, removal, querying, etc can be done with long package names or short package names
    - `-q [long/short packagename]` - shows all packages meeting the indicated values that are installed
        - `-qi [long/short packagename]` - detailed information about the indicated values that are installed
        - `-ql [long/short packagename]` - a listing of all files installed with the package
        - `-qip` or `-qlp [package.rpm]` - show the appropriate details about a file NOT installed as named
        - `--changelog [long/short packagename]` - display the changelog for the indicated package (must be used with `-q`)
        - `-qc [long/short packagename]` - display all configuration files for the indicated package
        - `-qRp [package.rpm]` - show the requirements of the indicated rpm file
        - `-qf [/path/of/file/to/check]` - will show the package that the indicated file dependency belongs to
    - `-K` (or `--checksig`) `[package.rpm]` - validate (using the package checksums) for the indicated package
        - The result of 'NOT OK' could mean a problem, but it may just mean the signature key needs to be imported
    - `--import [URL of keyfile to import]` - import the indicated keyfile from the package maintainer, rerun the validation command once the key is installed
    - `-i` (or `--install`) `[package.rpm]` - install the indicated package (this will fail if dependencies are not met)

- `-v` (or `--verbose`) `[package.rpm]` - print verbose information (used in conjunction with other options)
- `-h` (or `--hash`) `[package.rpm]` - during activity, print a 'status' bar using hash '#' character, used in conjunction with other options
    - For example - `rpm -ivh telnet-0.17-60.el6.x86_64.rpm`
        - Will install the indicated package, with hash status under the command, in verbose mode
- `--force` - option added during install or removal to force installation/removal even if there are conflicts with existing files or packages
- `--replacefiles` - option added during install (a bit less risky), that just replaces duplicate files
- `--nodeps` - option added during install or removal to force the installation/removal of the package without the indicated dependencies
    - NOTE: forcing installations without resolving the conflicts between files or dependencies can leave your system in an unusable state
- `-V [long/short packagename]` - verify the integrity of a package, will compare the package's current installation and configuration to the default package installation and configuration
    - No output means no differences are found
    - If the package is not installed, an error message will appear indicating so
    - Differences found will be output as a nine character output
        - In order - (S)ize change, (M)ode changed, (5) MD5 sum differs, (D)evice major/minor number match issue, (L)ink path problem, (U)ser ownership change, (G)roup ownership change, (T)ime of modification change, (P) capability set change
        - NOTE: they only appear IF that category has changed
        - For example - `rpm -V vsftpd`
            - Output - `S.5....T.  c /etc/vsftpd/ftpusers`
                - In this case, the configuration file (c) `/etc/vsftpd/ftpusers` has changed in size (S), MD5 sum (5) and modification time (T) since it's installation with the package, the other categories have remained the same
- `-Va` - verify ALL installated packages on the system
- `-Vac` - verify ALL installed package configuration files on the system (often captured for later comparison)

- `-U` (or `--upgrade`) `[packagename]` - will install or upgrade the indicated package name, while removing previous versions if they exist
- `-F` (or `--freshen`) `[packagename]` - will ONLY upgrade the indicated package if it is already installed, if not installed, nothing will be done
- `-e` (or `--erase`) `[short/long packagename]` - erases the indicated package name
    - NOTE: if short name is used and multiple matches for that short name are found, an error will be displayed and the long name must be used OR --allmatches must be added to the command (CAUTION)

- `rpm2cpio`
    - Extracts the contents of the *.rpm package and streams it to standard out (or it can be redirected to a file)
    - The `cpio` utility can then be used to extract the files, which can allow the installation on non-RPM based systems
        - For example - `rpm2cpio telnet-0.17-60.el6.x86_64.rpm > telnet.cpio`
            - Will unpack the rpm file indicated and create the cpio archive 'telnet.cpio'
            - This can be extracted then with `cpio -idmv < telnet.cpio`, effectively 'installing' it in the local directory (see output)

Topic 103 - GNU and Unix Commands

103.1 Work on the Command Line (Shells, Sourcing and Special Characters)

- Bash
    - The default Linux shell used to interpret the commands and scripts on our system
- Other shells
    - ash, c-shell, ksh, zsh
- Special shells
    - `/bin/false`, `/etc/nologin` (caution - this will NOT exist, if you add it, you won't be able to login)
- Script execution
    - Bash reads the contents of the file indicated and, if the proper permissions are set, the commands within it are executed one by one (like they were typed on the command line), but in

a 'new' shell

- Any variables set within the execution space of the shell during the script, will be gone (called 'going out of scope') once the script is complete and exits back to the command prompt
- For example - `./myscript.sh`
  - Will execute the `myscript.sh` script from the current working directory (the `./` in our command)

- Script sourcing
  - Bash reads the contents of the file indicated and, if the proper permissions are set, the commands within it are executed one by one (like they were typed on the command line), but in the CURRENT shell's environment
  - Allows anything set within that shell to remain set (i.e. environment variables) once it has completed
  - For example - `source myscript.sh` or `. myscript.sh`
- `/etc/profile`
  - System wide configuration script that will affect each user's environment (provided they use the bash shell)
  - It is a 'sourced' script and is executed each time a user logs in through a login shell
- `/etc/bashrc`
  - System wide configuration script that will affect each user's environment (provided they use the bash shell)
  - It is a 'sourced' script and is executed each time a user logs in through a login shell OR they execute a non-login session (see below)
- `~/.bash_profile`
  - Another 'sourced' script that is executed when a user logs in through a login shell, but it ONLY affects the environment for the user logging in (it is located in their home directory, indicated by the home directory shortcut of '~' in the path)
  - Sometimes is called `~/.profile` or `~/.bash_login`; however, if any exist in ADDITION to `~/.bash_profile`, only the `~/.bash_profile` will execute
  - Typically used to set environment variable specific to the user (like adding the local home directory `~/bin` to the PATH variable)
- `~/.bashrc`
  - Another 'sourced' script, one that is usually called by the '~/.bash_profile' script
  - Will source the `/etc/bashrc` file if it exists

    - Typically used to customize the shell prompt, keyboard shortcuts, command aliases, etc
- `~/.bash_logout`
    - Another 'sourced' script, one that is executed when the `logout` or `exit` commands are executed
- Session order example:
    1. user log in
    2. `/etc/profile` (sourced)
    3. user `~/.bash_profile` (sourced)
    4. user `~/.bashrc` (sourced) from `~/.bash_profile`
    5. user performs their tasks as needed
    6. user logs out with `logout` or `exit`
    7. user `~/.bash_logout` (sourced)
- 'Non-login' session
    - When the root or other user uses the `su` command to become another user (or the root user), it does not load the full environment of that user by default
- Built in environment settings
    - Command completion
        - Typing any command in the system path or any valid path, partially, and pressing the <TAB> key will attempt to 'autocomplete' the remainder of the command or path
    - Special characters
        - ~ - user home directory (example - /home/user)
        - \ - escape character
        - $ - identifies a variable
        - ? - single character 'wildcard'
        - * - 0 to n 'wildcards'
        - & - send a process to the background
        - && - execute second command only if the first is successful (example - /bin/true && echo 'It Works')
        - | - pipe, for sending command output to another command

    - || - execute second command only if the first is unsuccessful (example - /bin/false || echo 'Got Here')
    - ; - executes multiple commands on the same line
    - [] - defines a range of characters (numbers or letters)
    - > - redirect standard output to a file
    - < - redirect standard input to a program
- Command exit code
    - every command succeeds or fails and provides an exit code that you can see after
        - `echo $?` (0 for success, 1 or higher for failure)

103.1 Work on the Command Line (Environment Variables - Displaying, Setting and Using)

- Environment variables
    - 'Shortcuts' that allow you to indicate paths, commands, aliases or other information with an easy to use and remember shortcut designation
- `env`
    - Command to list all environment variables set for the current session EXCEPT for shell settings
- `set`
    - Used to view any shell settings or shell variables for the session
    - Can be used to set options on or off
        - `-o [option]` - turns the option ON
        - `+o [option]` - turns the option OFF
            - Options available
                - `vi` or `emacs` - keymap style for command line
                - `hashall` - default ON, enables hash table of commands and locations for repeat use
                - `history` - default ON, allows HISTFILE to be read to find history file (see below)
                - `monitor` - runs background processes in a different group and let the console know when they are done
                - `noclobber` - default OFF, will not allow the '>' redirect to overwrite an existing

file

    - `noexec` - default OFF, scripts will run a syntax check when enable on all commands when run
    - `notify` - terminated jobs notify the console immediately instead of during the next execution of the 'jobs' command
    - `verbose` - echoes any command to the terminal/screen before they are run

- `shopt`
    - Used to view any shell settings or shell variables for the session
- `PATH`
    - An environment variable containing a colon separated listing of directories that are searched (in order) for command executable files (i.e. files with execute permissions)
    - Absolute path - `/home/user/command`
    - Relative path - `./command`
    - Usually the PATH environment variable is initially set (for the system) in the `/etc/profile` file
    - The PATH can be modified on the command line or any other executable sourced script (i`/home/user/.bashrc` for example)
        - For example (in `/etc/profile`) - `export PATH=/sbin:/bin:/usr/bin:/usr/local/bin`
            - Sets the general 'bin' directory locations for executables on the system
        - Example of PATH modified in `/home/user/.bashrc` - `export PATH=$PATH:/home/user/bin`
            - Takes the existing value (not including $PATH will overwrite the current value, losing it) and appends the `/home/user/bin` so that the user can place their own scripts in that directory and execute without having to provide an absolute path to them
- `HISTFILE`
    - `history` command uses this value to display a list of previously run commands
    - By default, the value for this for any user will be `/home/user/.bash_history`
    - Variables that control the bash shell's use of history are:
        - `HISTCMD` - index of the current command
        - `HISTCONTROL` - when set to `ignorespace`, any command preceded by a blank space will NOT be recorded in the history file, when set to `ignoredups`, two consecutive lines

that are a duplicate will have one ignored

- `HISTFILESIZE` - the number of lines that can be used to hold previous commands (default is 500)

- Environment variable change
  - All variables can be changed in one of two ways:

```
1. Overwrite (export VARIABLE=newValue)

2. Append (export VARIABLE=$VARIABLE:newValue)
```

- Using environment variables on the command line
  - `echo [$ + variable name]` - will display the current value
    - For example - `echo $HOME`
      - Will display the current user's home directory as set in that variable
    - For example - `cd $HOME (or cd ~/)`
      - Will change your shell to the current user's home directory (see list of special characters in the shell above)

103.2 Process Text Streams Using Filters (Textutils Package Commands - sort, nl, wc, expand, cut, paste, join, uniq, head, and tail)

- `sort`
  - Will sort (numerically or alphabetically, starting by default at column 0) each line in the indicated file
  - `sort [filename]` - output a default sort of each line (top to bottom) on the screen
  - `-k[#] [filename]` - perform the sort at the start of the second delimited field (the delimiter by default is a space or tab)
  - `-n [filename]` - sort the file numerically (paying attention to the entire number, not just column 0)
- `nl`
  - Numbering the lines in a file (or input stream), can be all lines or just lines with data
  - `nl [filename]` - will number all lines with data (default)

    - `-ba [filename]` - will number all lines, even if the line is empty
- `wc`
    - Abbreviation for 'word count', but can do more than that
    - `-l` - number of lines
    - `-w` - number of words
    - `-c` - number of characters (bytes)
    - Is used as a complement to the output of another command
        - For example - `cat /etc/passwd | wc -l`
            - This will output the `/etc/passwd` file, filtering it through the `wc` command and displaying a count of the number of lines in the file
- `expand`
    - Changes tabs in a file to a specific number of spaces (default 8)
    - `-t` (or `--tabs=[#]`) `[#]` - converts tabs to # of spaces indicated
- `cut`
    - Command that will allow the extraction of field(s) or column(s) of data from a particular location in the indicated file
    - `-c [#][-#] [filename]` - will display only the column (or range of columns) from the indicated filename
    - `-d[delimiter]` - sets the delimiter to use when dealing with fields (default - TAB)
    - `-f [field1[,field2[,field3]]] [filename]` - identify the field numbers (identified by the delimiter) to display from the indicated filename
- `paste`
    - Combines two files together without removing any data (concatenation style)
- `join`
    - Combines two files together but removes redundant fields (database style) - based on the first field (key field)
    - `-t[character]` - use the indicated character as the field separator (delimiter)
- `uniq`
    - Allows you to extract only the 'unique' lines of data from a file
    - Apply a sort to the file before running this command (particularly on large files)

    - `-u [filename]` - print ONLY the unique lines in the file
    - `-d [filename]` - print an example of each line that is duplicated in a file
    - `-D [filename]` - to print ALL instances of duplicate lines in a file
- `head`
    - Similar to the `cat` command in that it will display contents of a file, but it displays only a certain number of lines from the top (default 10)
    - `-n [#] [filename]` - display 'n' number of lines, beginning at the top, of the indicated file
- `tail`
    - Although similar to the `cat` command in that it will display contents of a file, it is the polar opposite of the `head` command, only displaying a certain number of lines from the bottom of the file (default 10)
    - `-n [#] [filename]` - display 'n' number of lines, beginning at the bottom, of the indicated file
    - `-f [filename]` - display the indicated file, (default 10) last entries to begin with, but will then 'follow' the files output, adding additional values to the display as they are written

103.2 Process Text Streams Using Filters (Textutils Package Commands - split, cat, od, pr, fmt, tr, sed, more, and less)

- `split`
    - Allows you to take a file with a number of records and split it into multiple files that contain 'part' of the original data
    - `-a [#]` - when the split file(s) are created named them 'x#' (i.e. `-a 5` would create the file 'xaaaaa', alphabetical order for any subsequent files)
    - `-b [#][b/k/m]` - the new file(s) contain the indicated number of bytes/kilobytes/megabytes
    - `- [#]` - the new file(s) contain the indicated number of lines of bytes
    - `-l [#]` - the new file(s) contain the indicated number of lines
- `cat`
    - Display a file, top to bottom, in full, to standard output
    - `tac`
        - Same functionality, in reverse, display a file, bottom to top, in full, to standard output
- `od`

    - Allows the 'safe' display of a binary file to the standard output/terminal (without having to recover with a 'clear' or 'rest' of the terminal)
    - `-a` - display 'named' binary file
    - `-d` - decimal format
    - `-f` - floating point format
    - `-o` - octal format
    - `-x` - hexadecimal format
- `pr`
    - Commonly used to 'format' a source file or other text only files to be printed
    - Adds a header with the date of the 'job', the file, and pagination (page number) at the top
    - `--columns=[#]` - format text data in the file into the indicated number of columns for printing
    - `-a` (or `--across`) - print columns across instead of down (used in conjunction with --columns)
    - `-d` (or `--double-space`) - double space the line output
    - `-h` (or `--header=[text]`) - customizes the header text, replacing the filename with the indicate [text]
    - `-t` (or `--omit-header`) - strangely enough, omits the header from the output
- `fmt`
    - Also used to format files for printing, however, is limited to wrapping longer lines of output
    - Commonly used in combination with the `pr` utility (see above)
    - `-[#]` - the desired width to break each line at
    - `-s` - split long lines (over 50 characters) without filling
    - For example - `fmt -40 mytext.txt | pr --columns=2`
        - Will split each line at 40 characters (or as close as a whole word will allow) and the prepare a printing with header in two neat columns
- `tr`
    - Translate, allows you to change one or more characters (matching a pattern) in a file or stream
    - Not intended to substitute entire words or phrases
    - "[value(s)-from-range]' '['value(s)-to-range']' [filename]' - provide one or more character(s) via a range (like 'a-f') to substitute with other characters in a range (like '1-6')

- Cannot be used to operate on a file directly, the file intended has to be treated as a stream
  - For example - `tr 'A' 'a' < myfile.txt`
    - Would translate all uppercase 'A' to lowercase 'a' in the stream provided by redirecting the contents of 'myfile.txt' as an input to the command
- `[:upper][:lower] < [filename]` - convert all uppercase to lowercase in the input stream (can be reversed)

- `sed`
  - A very powerful stream editor that can quickly process and perform complex actions on streams of text (can be an entire course in and of itself, we are concentrating on common use cases for the exam)
  - Makes use of 'regular expressions', often called 'regex', to determine what to perform its operations on in the stream (and these can be EXTREMELY complex, but flexible, regex is another topic that could be an entire course)
  - `s/` - the most commonly used basic option, it substitutes what comes next
  - `[value(s)]/` - the value(s) to search for in the stream
  - `[value(s)]/` - the value(s) to replace the first value(s) with
  - `g` - when the command is ended with, replaces ALL instances of the first value with the second, if omitted, will only replace the first occurrence of the first value with the second
  - The sed application can use chaining of options for multiple replacements
    - For example - `sed 's/a/A/g' filename.txt`
      - Will replace every instance of lowercase 'a' with uppercase 'A' (omit the 'g' at the end to replace only the first)
    - For example - `sed 's/me/ME/g' ; 's/ME/YOU/g' filename.txt`
      - Will replace every instance of lowercase 'me' with uppercase 'ME' and immediately after, replace 'ME' with 'YOU' from the 'filename.txt'
  - `-e` - Can be used to chain multiple filters together (instead of the ';' character)
  - NOTE: if your first or second values do not contain spaces, it is not required to use single quotes; however, using single quotes will always work and is a good habit to be in
  - `-f [scriptlistfile] [filename]` - can use a file containing multiple lines of options to perform on the indicated filename
  - NOTE: sed treats the file as a stream and it outputs to standard output, it does not change the file itself
  - `-pg` - display only the changed lines (instead of 'g' which displays all)

    - `-n` (or `--quiet`) - suppresses the printing of 'pattern space', can prevent lines from appearing multiple times in output
    - `[#]{-/,][#]/` - provide a single, list or range of line numbers to perform the substitution on
- `more`
    - Allows paginate paging through text files (one screen at a time)
    - `-d` - prompt to 'space to continue' or 'q to quit' at each screen (rather than having to CTL-C or know to press 'q')
    - `-num [#]` - specify the screen size (in lines), the default is 50 or the size of the terminal (if it is a GUI terminal providing that information)
    - `-p` - clear the screen before starting the first screen display
    - NOTE: only allows the forward movement through a file (cannot go back to previous screens)
- `less`
    - Similar in function to `more`; however, also allows movement through the file backwards (previous values), is considered faster as it does not read the entire file before starting display
    - NOTE: options listed below are during the file display (keyboard shortcuts)
        - space - moves to the next full screen of text
        - q - quits the display
        - d[#] - scrolls through the next number of lines indicated (no number indicated, defaults to one full screen)
        - b[#] - scrolls backward through the next number of line indicated (no number indicated, defaults to one full screen)

103.3 Perform Basic File Management (Commands and Terms to Know - ls, cd, pwd, mkdir, file, globbing, touch, stat, cp, and mv)

- `ls`
    - 'list' directories, files or both
    - `-l` - long listing, containing permissions, owner, size and date/timestamp
    - `-a` - list files, including 'hidden' files (files that start with a .)
    - `-d` - list just the directory without listing the files in it
    - `-i` - display the inode numbers for the file or directory

- `-h` - add 'human-readable' format to filesizes and details
- NOTE: all options can be combined
  - For example - `ls -ailh /home/user`
    - Would display the files and directories in i`/home/user`, including hidden files, their permissions, owner, group owner, size (in human readable format) and inode numbers

- `cd`
  - Change directory
  - Can be used as an absolute path, a relative path, or values in variables as part of the path
  - `~` - a special character that means 'home directory'
    - NOTE: be careful, going to your home directory is 'cd ~`, but` cd ~user' is the home directory of that user
      - 'cd ~/tmp' - your home directory's tmp subdirectory
      - 'cd ~tmp' - the home directory of the 'tmp' user (note the missing /)
  - `.` - a reference to the current directory (`cd .`)
  - `..` - moves you up one directory (`cd ..`)
    - Can be chained in a relative path
    - For example - `cd ../../mydirectory`
      - This would move up TWO directories in the tree and into the 'mydirectory' directory (assuming it exists)
- `pwd`
  - Display the current directory you are in
- `mkdir`
  - Make a directory (if it does not already exist)
  - `-p` - make all directories in the indicated path, if they do not already exist
    - For example - `mkdir -p /home/user/dir1/dir2/dir3/dir4`
      - Will create ALL directories in the indicated path if they do not exist
- `file`
  - Can determine the type of file the indicated file is
    - For example - `file myfile.txt`

        - Assuming it was a text file, would display `myfile.txt: ASCII text`
    - Can be run against an entire set of files in a directory, which will list the type for every one
        - For example - `file /home/user/*`
            - Would display all files in `/home/user` and their file type
- Globbing
    - A wildcard or list operator/character that matches one or more files based on a defined/indicated pattern
        - For example - `ls -al /usr/bin/*.sh`
            - Will display the appropriate attributes of all files in `/usr/bin` that end with '.sh' (any number of characters before that)
        - For example - `ls -al /usr/bin/*.?h`
            - Will display the appropriate attributes of all files in `/usr/bin` that end with '.[one character]h' (any number of characters before that)
        - For example - `ls -al /usr/bin/[a-f]*.sh`
            - Will display the appropriate attributes of all files in `/usr/bin/` that begin with characters between 'a through f' with any number of characters after, ending with the '.sh' extension
- `touch`
    - Creates a file (if it does not exist) or updates the date/timestamp on the file if it does
    - Often used to create a new file that an application or logging utility expects to exist
    - Can be used with absolute or relative path names
        - NOTE: you must have permissions to create/update a file in the location indicated
    - `-t [date] [filename]` - sets the date/time as indicated for the file
        - Date/time format - yyyymmddhhmm (y=year, m=month, d=day, h=hour, m=minute)
    - `-r [referencefile] [filename]` - applies the indicated date/time in the 'referencefile' to the 'filename' (or multiple files)
        - Used to define a 'starting point' for files without having to remember the value
- `stat`
    - Displays a file (or file system) status
    - `-f` (or `--file-system`) - display the file system status instead of file status

    - `-t` (or `--terse`) - display the information in terse (short) form
- `cp`
    - Copy files and/or directories from one place to another
    - `-d` - do not follow symbolic links, just copy the link
    - `-f` - force overwrite (if the file already exists in the destination)
    - `-i` - ask before overwriting an existing file
    - `-l` - create a hard link to the original
    - `-s` - create a symbolic link to the original
    - `-r` (or `-R`) - recursively apply any options to other directories and subdirectories in the indicated path
    - `-u` - update copies only if the original is newer than the existing file (or create the destination file if it does not exist already)
    - `-x` - do not include any files/directories from other filesystems (network shares)
    - NOTE: copy can be done with absolute or relative path names
- `mv`
    - Moves a file or directory from one location to another (unlike copy, which duplicates it in the next location)
    - Can be used with absolute or relative path names
    - NOTE: files or directories moved within the current location is effectively using the move command as a rename (since a file or directory cannot have a duplicate name in the same location)
    - `-i` - check to see whether the destination already exists, will prompt to overwrite
    - `-u` - do not overwrite the destination file or directory if it is newer than the original
    - `-f` - do not prompt for directory entry changes (NOTE: often the default in distributions)
    - NOTE: care must be taken when moving the contents of one directory into another
        - For example - `mv mydir1 mydir2`
            - This can have two difference behaviors:
                - If 'mydir2' does not exist, it effectively renames 'mydir1' to 'mydir2'
                - If 'mydir2' does exist, it moves 'mydir1' INTO 'mydir2'
        - For example - `mv mydir1/* mydir2`

- This will move the file(s) within 'mydir1' to the 'mydir2' directory, unless it does not exist, then it will error

103.3 Perform Basic File Management (Commands and Terms to Know - dd, rmdir, rm, and find)

- `dd`
  - Used to create backup images, specially formatted files for swap space, CD/DVD ISO files and others
  - `if=[image file or device]` - used as an input (as an image file like .img or .iso) or as a device (like /dev/sr0)
  - `of=[image file or device]` - used as an output (as an image file like .img or .iso) or as a device (like `/dev/sdc`)
  - For example - `dd if=/dev/sda of=/dev/sdb`
    - Will back up the full `/dev/sda` disk to `/dev/sdb` (assuming it has the same or greater amount of space)
  - For example - `dd if=/dev/sr0 of=/tmp/cdimage.iso`
    - Will take an image of the CDROM device at /dev/sr0 and create an ISO image file of it at `/tmp/cdimage.iso`
  - For example - `dd if=/dev/sda of=/tmp/mbrbkup.img count=1 bs=512`
    - Will back up the master boot record from `/dev/sda` reading a single (count) block of 512 byts (bs)
    - Reverse to restore it (if and of)
- `rmdir`
  - Remove a directory, but only if it is EMPTY
  - `-p` - remove all directories in a path, as long as they are ALL empty (except for other empty directories)
  - `--ignore-fail-on-non-empty` - allows the removal of directories that do have files or non-empty directories (recursive)
    - NOTE: rarely used, see the `rm` command and options below
- `rm`
  - Remove files (or directories)
  - `-r` - recursively remove files (and subdirectories if the target is a directory)

- `-f` - do not prompt for confirmation on deletion
- `-i` - prompt for EVERY file/directory to be removed

- `find`
  - The best method for 'finding' files based on name, type, or other characteristics on the system (although it can be costly in terms of CPU and IO performance)
  - `[starting path]` - where to begin the search
  - `-[options]` - what type of 'thing' to find
  - `[value(s)]` - what you are trying to find (filename, directory, globbed name, etc)
  - For example - `find / -name "messages"`
    - Will search the file system, beginning with the `/` directory, recursively trying to find any file or folder with the name 'messages'
  - Options you can find:
    - group - files/directories belonging to the specified group
    - user - files/directories belonging to the specified user
    - newer - files/directories newer than the indicated file
    - name - files/directories whose names match exactly (case included) the indicated name
    - iname - files/directories whose names match (case exclusive) the indicated name
    - mtime - files/directories matching the indicated modification time (+n days old)
    - atime - files/directories matching the indicated number of days since accessed
    - ctime - files/directories matching the indicated number of days since last changed
  - '-exec [cmd] {} [output] \;'
    - Allows the execution of [cmd] on all values found, fed to that command ({} as the placeholder for each), with any associated output of each
    - For example - `find /home/user/bin -name "*.sh" -exec cp -f {} /home/user/backup/bin \;`
      - This will recursively search the `/home/user/bin` directory for any files ending in .sh and cp them, overwriting without confirmation if they already exist, to the `/home/user/backup/bin` directory
    - NOTE
      - Multiple values can be used

- For example - `find /var -name "*.log" -group "apache"`
  - Will recursively search the `/var` directory for any files ending in .log that are owned by the group 'apache'

103.3 Perform Basic File Management (Archiving and Compression Utilities)

- `tar`
  - Tape archive (developed to backup file systems to a tape device originally)
  - One of the most common utilities to back up and (through extensions), compress files and directories
  - Can use various compression types (gzip and bzip)
  - `-c` - create the archive
  - `-t` - displays the contents of the archive
  - `-x` - extract the contents of the archive
    - NOTE: one of the three options above is ALWAYS required
  - `-f` - the name of the file to create
  - `-j` - compress/uncompress with bzip2 (not always available by default but is the best compression method)
  - `-z` - compress/uncompress with gzip (available by default and is most commonly used compression method)
  - `-v` - verbose messages (this output approximates the output of `ls -al` as the archive is being created/unarchived/viewed)
  - NOTE: many options are made to use together in a single option list
    - For example - `tar -cvjf mybackup.tar.bz2 /home/user`
      - Will create a file called `mybackup.tar.bz2` in the current directory, containing all the files in `/home/user`, compressed with the bzip2 compression utility
  - One or more files or directories can be compressed in an archive by passing a list of them with a space between each
- `cpio`
  - Usually used by receiving input from a file or another command and then sends the files to either standard output (default) or a file if redirected
  - `-o` (or `--create`) - runs in copy-out mode (by default, to standard output or the screen)

    - `-O [archivefile]` - creates the indicated file instead of using standard output
    - For example - `ls -al /home/user/bin/*.sh | cpio -o > scriptbkup.cpio`
        - Would list all files in `/home/usr/bin` that end in .sh and pipe that output to the `cpio` command running in copy-out mode to the terminal and redirected into a file called `scriptbkup.cpio`
    - For example - `ls -al /home/user/bin/*.sh | cpio -O scriptbkup.cpio`
        - The same as the above command, but creates the archive file indicated directly without redirecting standard output
    - `-d` (or `--make-directories`) - make leading directories if needed
    - `-i` (or `--extract`) - extracts the content in copy-in mode (by default, to standard output or the screen)
    - `-I [archivefile]` - use the archive file indicated rather than standard input from a command or redirected file
    - `-v` - verbose messages
- `gzip` (and/or gzip2)
    - Compression utility for files and directories
    - `-r` - recursive, include all files and directories
    - CAUTION: compresses the original file and replaces it with the new compressed version
        - For example - `gzip /home/user/myfile.txt`
            - This would compress the `/home/user/myfile.txt` file, leaving `/home/user/myfile.txt.gz` in its place (the uncompressed version is removed)
- `gunzip`
    - The gzip extraction utility for compressed files
- `bzip2`
    - A better (in terms of size) compression utility for files and directories
    - CAUTION: compresses the original file and replaces it with the new compressed version
        - For example - `bzip2 /home/user/myfile.txt`
            - This would compress the `/home/user/myfile.txt` file, leaving `/home/user/myfile.txt.bz` (or .bz2) in its place (the uncompressed version is removed)
- `xz`
    - New(er) compression utility for files and directories

- Better performance than bzip2 (but not better compression)
- `-z` (or `--compress`) - compress the file indicated
- `-d` (or `--uncompress` or `--decompress`) - decompress the file indicated
- CAUTION: compresses the original file and replaces it with the new compressed version
  - For example - `xz -z /home/user/myfile.txt`
    - This would compress the `/home/user/myfile.txt` file, leaving `/home/user/myfile.txt.xz` in its place (the uncompressed version is removed)

103.4 Use Streams, Pipes and Redirects

- Standard input stream
  - A stream that provide input to terminals, files, applications, and/or utilities (something all programs are assumed to be able to accept in some form)
- Standard output stream
  - A stream that provides output to terminals, files, applications, and/or utilities
- Standard error stream
  - A stream of error messages that are output to terminals, files, applications, and/or utilities (considered to be a superset of the standard output stream)
- Redirection
  - The process of taking a stream and sending it 'somewhere else', other than the default
- Devices
  - Standard input - `/dev/stdin`
  - Standard output - `/dev/stdout`
  - Standard error - `/dev/stderr`
- File descriptors and related files
  - Standard input - file descriptor 0 - file `/proc/self/fd/0`
  - Standard output - file descriptor 1 - file `/proc/self/fd/1`
  - Standard error - file descriptor 2 - file `/proc/self/fd/2`
- Redirectors
  - `|` - pipe, for sending command output to another command
    - For example - `cat /var/log/messages | more`

    - `>` - redirect standard output to a file or device (creating or overwriting the destination if a file)
        - For example - `find /user -name "*.sh" > output.txt`
    - `>>` - redirect standard output to a file or device (appending to the destination if a file)
        - For example - `find /user -name "*.txt" >> output.txt`
    - `<` - redirect standard input to a program
        - For example - `sort < /home/user/listfile.txt`
        - NOTE: similar behavior to what you would get with `cat /home/user/listfile.txt | sort`
    - NOTE - redirects can happen by default (based on the redirector) or by using the file descriptor with the redirect
- Redirecting standard error
    - Commonly, stderr is redirected to a file (logging) or to a special device called `/dev/null` (effectively discarding it)
    - This allows you to clear up errors from the normal standard output
    - For example - `find / -iname "*.sh" 2> /dev/null`
        - Will display the results of the found matches without displaying error messages related to permissions
- Chaining redirects
    - For example - `find / -iname "*.sh" 2> /dev/null > output.txt`
        - The same as the above example, but matches will be redirected to the output.txt file instead of displayed on the screen
    - For example - `sort < listfile | nl`
        - Redirect the 'listfile' as an input stream to the `sort` command, piping that output to the `nl` command to add line numbers
- Special combination - `2>&1`
    - Often used to throw away all output for a job or process (like a cron job) where attempting to display or capture the data would cause issues
    - For example - `find / -iname "*.sh" > /dev/null 2>&1`
        - Redirects standard error to standard output (2>&1>) and the whole output stream is redirected to `/dev/null` (discarded)
- `tee`

- Accepts a standard input stream and sends one (identical) output stream to an indicated file
- Often used when you want to capture the output of an application but also need to see the results on screen
  - For example - `find / -name "*.sh" | tee visibleresults.txt`
    - Will find all files ending in .sh from the root partition, piping those results as an input stream to `tee` which then opens two standard output streams, sending one to the console and one to the indicated file `visibleresults.txt`

- `xargs`
  - Takes an input stream (the result(s) of another command - commonly the `find` command), and then feeds them to another command as indicated
  - For example - `find / -name "*.sh" | xargs ls -al > myresults.txt`
    - Will find all files ending in .sh and then xargs will take that output and feed it to the `ls -al` command to display the details of each file and redirect output to the `myresults.txt` file

103.5 Create, Monitor and Kill Processes (Terms and Utilities - PID, ps, pstree, free, uptime, signals, kill, killall, pkill, pgrep)

- PID
  - Process identification - a unique number (on the system) that can be used to refer to a running process
- `ps`
  - Displays processes that were started and are running as the current or indicated user
  - `-a` - display all running processes on the system, for any user
  - `-u` - display user information for the displayed processes
  - `-x` - display processes without an associated tty (terminal)
  - NOTE: these options are often used together and can be used with or without the preceding `-`
  - For example - `ps aux`
    - Will show all running processes, by any user, displaying the process owner and including processes not tied to a terminal
    - This is the Linux syntax for viewing these items
  - For example - `ps -ef`

        - Will show all running processes, by any user, displaying the process owner and including processes not tied to a terminal
        - This is the Unix syntax for viewing these items
    - By the transitive property, despite formatting differences, `ps aux` = `ps -ef`
- `pstree`
    - Show a tree view (hierarchical ASCII or ncurses display) of running processes
    - `-A` - display the tree using ASCII characters
    - `-a` - display the processes including any parameters used
    - `-p` - show PIDs
- `free`
    - Provides information about system memory (including total, in use, shared, buffers, and cached, etc)
    - `-b` (or `--bytes`) - displays the memory in bytes
    - `-k` (or `--kilo`) - displays the memory in kilobytes (default)
    - `-m` (or `--mega`) - displays the memory in megabytes
    - `-g` (or `--giga`) - displays the memory in gigabytes
    - `-h` (or `--human`) - displays the memory if a more 'human-readable' format
    - `-c` (or `--count`) - the number of times to display the output (must be used with `-s` option)
    - `-s` (or `--seconds`) - how many seconds between each display output (used with `-c`)
    - `-t` (or `--total`) - display a line showing each columns totals
    - `-l` (or `--lohi`) - display low and high memory statistics
- `uptime`
    - Displays the amount of time the system has been running since the last boot/reboot
    - Also provides information on how many users are currently on the system and the average load on the system in the last 1, 5 and 15 minutes (three values)
- Signals
    - What is sent to a process that it then reacts to, depending on the type of signal
        - SIGHUP - signal 1 - shutdown and restart the process (hang up)
        - SIGINT - signal 2 - interrupts a process (CTL-C)

    - SIGKILL - signal 9 - kill the process (cannot be ignored or caught)
    - SIGTERM - signal 15 - terminate (the process can ignore or 'catch' the signal)
    - SIGSTOP - signal 19 - stop (cannot be ignored or caught)
    - SIGTSTP - signal 20 - terminal stop (CTL-Z)
- `kill`
    - Sends a 'nice' kill command to the indicated PID (default SIGTERM signal 15)
    - This method allows the process the ability to 'cleanly' stop (terminate, release memory, close open files, etc)
    - `-[signal #]` - attempts to kill the process with the indicated signal
    - `-1` (`-HUP`) - special reference to ask a process to restart (in order to reread a configuration file or implement a change)
    - `-9` (`-KILL`) - kill/stop/end/dump right now (most often will kill off even zombie or hung processes)
- `killall`
    - Kills all instances of the named process (useful when there are multiple processes using the same process name/application name - example 'httpd')
- `pkill`
    - Process kill, kills processes based on name, ID, user, session or terminal (tty)
    - NOTE: if multiple criteria is used, ALL options must match for it to apply to a process
    - `-signal` (or `--signal`) `[#]` - send the signal number to the matched process
    - `-t [terminal]` - match the indicated terminal/tty
    - `-U` (or `--uid`) `[user]` - match the user ID
- `pgrep`
    - Allows for the testing (debugging) of the pkill command (displays what will happen)
    - Displays the process ID that will be affected
    - Takes ALL `pkill` options to be able to determine what PID will be affected
    - NOTE: if multiple options are supplied, how they are listed determines how they are interpreted
        - For example - `pgrep -u root,apache httpd`
            - Will display any httpd process owned by root OR apache

    - For example - `pgrep -u root apache`
        - Will display only processes owned by root AND apache

103.5 Create, Monitor and Kill Processes (Terms and Utilities - jobs, bg, fg, &, priority, nohup, and screen)

- `jobs`
    - Will display the status of jobs that have been paused
    - Display:
        - `[#]- Stopped [cmd]` - indicates the previous job or the next to last job to be operated on
        - `[#]+ Stopped [cmd]` - indicates the current job with any commands (`bg` or `fg`) that will act on it
- `bg`
    - Sends the indicated job to the background (commonly used after the above `jobs` command to place a paused job into the background so it becomes active again)
- `fg`
    - Bring the indicated job to the foreground (for bringing a previously `bg` process to the foreground for use)
- `&`
    - Places a program into the background when run after a command
        - For example - `vim &`
            - Will run and place the vim editor into the background
- `priority`
    - Scheduling and priorities are what Linux uses to run multiple applications/services/process on a single machine in a 'multi-tasking like' manner
    - Default process priority is '0'
    - Range of priorities is '0' to '19' and '0' to '-20' (the 'lower' a process is, the more resources it will receive, with -20 being the highest possible priority)
        - Any user can start processes with priorities from 0 to 19
        - Only root can start processes with priorities from 0 to -20
- `nohup`

- Allows you to start an application or process at the command line/terminal and then stop/exit/logout of the terminal or session, while leaving the application or process running
- For example - `nohup find / -name "somefile" > output.txt`
  - This potentially long running file command (assume on a very large filesystem) would stop if we log off the machine, close the terminal - `nohup` allows it to continue if we stop/exit/lose connection

- `screen`
  - Text-based session manager that can run multiple shell sessions (independent of each other) within a single window
  - Sessions can be used for local or remote connectivity (as you would multiple terminals)
  - Supports copy/paste between sessions
  - Scrolling buffer per session
  - Supports output logging if configured
  - Ability to detach (often called "abandon") a window, allows programs launched within to keep running
  - Options once `screen` is started
    - ctl+a then c - open a new session
    - ctl+a then p - move back to previous session
    - ctl+a then n - move to next session
    - ctl+a then " - display a list of available sessions (numeric identification)
      - After the numbers are shown, pressing the corresponding number and enter will switch to that session
    - ctl+a then d - detach from screen (processes still running)
    - ctl+a then x - lock the screen
      - Will prompt (twice) for a password to unlock
    - `screen -ls` - show screens
    - `screen -r` - reattach to the screen

103.6 Modify Process Execution Priorities (Using nice, renice, and top)

- `nice`

    - Allows a user to start a process with a priority lower than the default (again '0' is default, nice will start with '10')
        - For example - `nice myscript.sh`
            - Will start the myscript.sh with a lower priority of '10'
    - `-n [#] [program]` - allows you to change the default priority of the indicated program
- `renice`
    - Changes the priority of a process
    - Commonly used to lower a process priority that may be consuming too many resources or to raise a process priority to that it can complete faster
    - `+[#] [PID]` - change the priority to a higher number (which is a lower priority)
        - All users can lower the priority of a running process (raise the number)
    - `-[#] [PID]` - change the priority to a lower number (which is a higher priority)
        - Only root can raise the priority of a running process (lower the number)
- `top`
    - Configuration is read from `/etc/toprc` (or local `/home/user/.toprc` for user configurations)
    - Can be used to display (actively) the 'top' running processes
    - Can also be used to alter the priority of running processes
    - `d [#]` - run and update the processes display every '#' of seconds
    - `i` - show only active processes
    - `-b` - run in batch mode
    - `-n [#]` - run '#' of updates and exit
        - For example - `top -b -n 5 > output.txt`
            - Will run top in batch mode and update 5 times, write the results to the `output.txt` file and exit
    - While running, the following keyboard shortcuts apply:
        - spacebar - immediate update
        - h - displays help screen
        - k - prompts for PID to kill

- i - display or ignore inactive/idle/zombie processes (toggle)
- n - prompt for the number of processes to display on screen
- r - prompt for the PID to apply 'renice' to and the new priority (remember - higher number, lower priority)
- R - sort process IDs from high to low (reverse of default)

103.7 Search Text Files Using Regular Expressions (Using grep, egrep, fgrep, sed, and regex)

- `grep`
  - A utility to find strings and phrases in file, streams or directories
  - Stands for - global regular expression print
  - `-c` - count of matches found only
  - `-C [#]` - encloses the string match with '#' of lines of context (before and after data)
  - `-E [ext. regex]` - use the indicated extended regular expression for finding a match
  - `-F [fixed regex]` - use the indicated fixed regular expression for finding a match
  - `-H` - displays the filename of each matching string or phrase
  - `-h` - prevents the filename from being displayed
  - `-i` - ignore case
  - `-l` - show only the filename, not the matched string/phrase
  - `-L` - show only filenames that do NOT contain a match
  - `-w` - match only lines containing the whole string or phrase
  - `-x` - show only exact whole line matches to the entire string or phrase
  - `-v` - show only those lines in a file that do NOT match the string or phrase (the opposite of default)
  - for example - `find / -name "*.sh" -exec grep -iH "modprobe" {} \;`
    - This would find any file, recursively, starting in root and ending in .sh, passing that to the `grep` command, which would ignore case and display the file name and matching content from any file containing the word 'modprobe'
  - For example - `grep -cv "/bin/bash" /etc/passwd`
    - Would display only lines that do NOT contain the value '/bin/bash' and display the count

(i.e. you would know the number of user accounts defined that do not have bash listed as the default shell)

- `egrep`
  - Grep command without having to specify the `-E`
  - Allows the use of extended regular expressions to match patterns of text
  - For example - 'egrep '^ope(n|r)' /etc/passwd'
    - This would display any entry in the `/etc/passwd` file that starts with 'ope' and then has either 'n' or 'r' as the next letter (followed by anything else after) - i.e. like openvpn or operator
  - Can use operators like OR (|)
    - For example - `egrep '(bin|bash)' /etc/passwd`
      - Would display any lines containing either 'bin' or 'bash'
- `fgrep`
  - Grep command without having to specify the `-F`
  - Allows for the use of a file that contains one or more items to search for (file grep)
  - `-f [itemfile]` - use the indicated file as a list of items to search for in the indicated file
    - For example - `fgrep -f itemfile.txt searchfile.txt`
      - Will use the items in `itemfile.txt` as search parameters for all lines in the `searchfile.txt` and display the matches
- NOTE: all `grep` commands support ranges (i.e. [a-z]) or globs (*) as search parameters
- Regular expressions
  - A method of searching for strings or phrases that are not known precisely using partial strings and special characters
  - Regex characters:
    - . - match any single character
    - ? - match an optional item, but only once
    - * - match from '0 to n' characters in a string
    - + - item MUST be matched at least once but can be matched more
    - {#} - match '#' of times
    - {#,} - match '#' of times, or more

- {#,#} - match between the first and second number (i.e. {3,10} would mean match between 3 and 10 times)
- \< - words that start with what comes after
- ^ - words that start with what comes after
- \> - words that end with what comes after
- $ - words that end with what comes after
- [aA] - words that contain either 'a' or 'A'
  - For example - `grep "d[iou]g" textfile.txt`
    - Would return words that start with 'd' and end with 'g' and have 'i' or 'o' or 'u' in them (i.e. dig, dug, dog)
- [^o] - words that do NOT contain the letter
  - For example - `grep "d[^o]g" textfile.txt`
    - would return words that start with 'd' and end with 'g' and have any letter in between EXCEPT 'o' (i.e. dig, dug)
- ^...$ - any line that contains exactly three characters

- For example - `grep "\<ope[^r]" /etc/passwd`
  - Would display words that begins with 'ope' followed by any letter EXCEPT 'r' in the `/etc/passwd` file (i.e. find 'openvpn' account but not 'operator')
- For example - `grep "nologin\>" /etc/passwd`
  - Would display all words that end containing the letters 'nologin'

103.8 Perform Basic File Editing Operations Using VI

- `vi`/`vim`
  - A common editor available in all Linux distributions
  - Operates in three distinct modes
    - Command mode - used to move your cursor around and perform various operations on text
    - Insert mode - adding/inserting text in your document
    - Ex mode (called LastLine) - advanced mode for search/replace, executing external commands and using split panes
  - `/etc/vimrc`

    - Global vim configuration file
- `/home/user/.vimrc`
    - User specific configuration for vim
- Command mode number lines shortcuts to know for LPIC-1
    - :number/nonumber - turns line numbers on the display on/off
    - :nu/no nu - shortcut for the above
- The screen
    - bottom - contains the 'message line'
        - Contains the full path to the file, the number of lines, the size of the file, current line and column and current cursor position
        - For example `"/etc/passwd" 33L, 3359C 10,1 Top`
            - The `/etc/passwd` file, 33 lines long, 3359 characters, line 10, column 1 and the cursor is in the top half of the document
- Vim will begin in 'command' mode
    - i - invoke the 'insert mode' at the current cursor position where you can begin typing in the document
    - I - move to the beginning of the current line and invoke 'insert mode'
    - a - invoke insert mode, placing the cursor one character to the right of the current position (append)
    - A - move to the end of the current line, placing the cursor one character to the right of the ending position (line append)
    - o - insert a new line under the current line, place the cursor in the first position on the new line in 'insert mode'
    - O - insert a new line above the current line, place the cursor in the first position of the new line in 'insert mode'
    - c[option] - change the text at the current position
        - cw - change the word at the current position
        - cc - change the line at the current position
        - c$ - change from the current position to the end of the line
    - r - replace the character at the current position
    - R - replace text on the same line until you escape the 'insert mode' or until you reach end

of line

- x - delete the character after the cursor
- X - delete the character before the cursor
- dw - delete the word after the cursor
- dd - delete the entire line the cursor is on
- D - delete the text from the current cursor to end of line
- dL - delete the text from the current cursor to the end of the current screen
- dG - delete the text from the current cursor to the end of the file
- d^ - delete all text from the beginning of line to the current cursor
- u - undo the last operation/change
- yy - copy the current line to the buffer (often called yank)
- yw - copy from current cursor to end of current word
- p - paste contents of the buffer after the cursor
- P - paste the contents of the buffer before the cursor
- :e! - undo ALL changes since the last time the file was saved to disk
- :w - write the file to disk (save)
- :q - quit the editor (will warn you if the file has changed and not saved)
- :q! - quit right now, don't worry about if it was changed and saved
- :x - shortcut for save and exit
- ZZ - shortcut for save and exit

- Navigation shortcuts
  - h - one character left
  - j - one line down
  - k - one line up
  - l - one character right
    - any of the above can be repeated by putting the desired number in front
      - For example - 3k (would move 3 lines up)

```
2h (would move 2 characters left)
```

    - ctl-u - move back one half page
    - ctl-b - move back one page
    - ctl-d - move forward one half page
    - ctl-f - move forward one page
    - ctl-G - show the name of the file, lines and position in percentage of the total file length
    - [#]G - move directly to the indicated line
    - [#]W - move 12 words to the right
- Searching
    - Must be in command mode (press ESC key to make sure you are in command mode)
    - /[string] - search from cursor position forward for 'string'
    - ?[string] - search from cursor position back for 'string'
        - N - when used after a search, will find the 'next' occurence of said 'string' in indicated direction
- Replacing
    - must be in command mode (press ESC key to make sure you are in command mode)
    - sed-like syntax
    - s - substitute (in the current line)
        - %s - substitute (in the entire file)
    - /[value to find] - what to search for
    - /[value to substitute]/ - what to replace with
    - g - optional, will replace ALL occurrences rather that just the first
        - For example - `:s/Mar/Apr/g`
            - Will search the line for all occurrences of 'Mar' and replace them all with 'Apr', add a '%' in front of the 's' and then it will replace all occurrences in the entire file
        - NOTE: complex regular expression syntax can be used
- Run commands
    - :! [cmd] - run the indicated command on the command line

Topic 104 - Devices, Linux Filesystems, and the Filesystem Hierarchy Standard

104.1 Create Partitions and Filesystems (Using Partitioning Tools - fdisk, gdisk, and parted)

- General order of preparing a drive for use
  1. Physical installation (or, in the case of virtual machines, allocation of disk to the VM)
  2. Partition the device (using one of several partitioning tools, depending on the size/features of the device)
  3. Format the partition(s) created with the chose filesystem type
  4. Create a system mount point (directory) and determine access permissions
  5. Mount the device/partition on the chosen directory
  6. If intended to be persistent, add an entry to `/etc/fstab`
- `/dev`
  - The standard Linux device directory where disks will have associated entries
  - IDE devices
    - For example - `/dev/hda` (and then partition numbers like `/dev/hda1`)
  - SATA/SCSI devices
    - For example - `/dev/sda` (and then partition numbers like `/dev/sda1`)
- Primary partition
  - Partition(s) that are independent of any extended and logical partitions (see below), typically you can have up to FOUR primary partitions (historically a limit of the MBR) on a disk device (with some exceptions - see GUID below)
- Extended partition
  - Only one of these per drive
  - Think of it like a 'container' for logical partitions
- Logical partition
  - Partitions within an extended partition, usually needed when there is a need for more than four partitions (see primary partition above)

- Swap partitions
  - Partition(s) that are specially formatted and dedicated to virtual memory in support of system memory exhaustion
  - NOTE: should be at least the size of system memory (historical recommendation was 2x system memory - exam topic!)
  - Although can be used as system memory in the event of system memory exhaustion, the system performance degradation can be steep when in use
- Numbering
  - Primary partitions - 1 through 4
  - Logical partitions - 5 through n
- `fdisk` (description)
  - Disk partitioning tool installed with every distribution (standard partitioning rules, no ability to edit GUID drives)
  - Deals with traditional MBR partition tables
    - Limitations - 2TB partition size limit, four partition limit (primary and extended - logical drives were created to address), no disk checksum support
- `gdisk` (description)
  - Disk partitioning tool installed in most modern distributions, capable of creating/modifying GUID tables
  - Addresses limitation of the MBR partition table
- `parted` (description)
  - Disk partitioning tool with an extended feature set (including disk/partition reorganization/resizing)
  - Often used with a GUI front end extension (gparted for example)
- `fdisk` (use)
  - Command line, along with device
    - For example - `fdisk /dev/sda`
  - `-l [device]` - list MBR status and disk partition information for the indicated devices
  - Interactive options (once the tool is started)
    - NOTE: nothing is written to the drive until you explicitly indicate (see option list below)
    - p - display disk/partition summary

        - n - create a new partition
            - p - make primary partition
                - [#] - number of the primary partition (1 through 4)
            - e - make extended partition
                - [#] - number of extended partition (2 through 4 - NOTE: counts as one of the four allowed primary partitions)
            - l - make logical partition
                - [#] - number of logical partition (range will depend on previous choices, but will be the number of the extended partition + 1)
        - NOTE: each partition will be automatically allocated as a 'Linux' partition type (type 83) unless changed
        - t - change partition type
            - [#] - hex code indicating type ('L' will list them)
                - Partition types to know for the exam:
                    - 82 - Linux swap
                    - 83 - Linux
                    - 85 - Linux extended
                    - 8e - Linux LVM
                    - fd - Linux RAID
        - w - write changes to disk
- `gdisk` (use)
    - Similar in functionality but support for additional filesystem types and the GUID partition table
    - `-l [device]` - list GUID (gpt) status and disk partition information for the indicated devices
    - Interactive options (once the tool is started)
        - NOTE: as with disk, nothing is written to the drive until you explicitly indicate (see option list below)
        - p - display disk/partition summary
        - n - create a new partition
            - p - make primary partition

        - [#] - number of the primary partition (1 through 128)
    - NOTE: each partition will be automatically allocated as a 'Linux' partition type (type 8300) unless changed
    - t - change partition type
        - [#] - hex code indicating type ('L' will list them)
            - partition types to know for the exam:
                - 8200 - Linux swap
                - 8300 - Linux
                - 8301 - Linux reserved
                - 8e00 - Linux LVM
                - fd00 - Linux RAID
    - w - write changes to disk
- `parted` (use)
    - Capable of dealing with both legacy MBR and newer GPT drives
    - `-l [device]` - list A LOT of information about the indicated device and partitions
    - Interactive options (once the tool is started)
        - help - list commands available
        - mkpart [volume name] [begin] [end] - create a partition with the indicated volume name at the beginning value and ending at the end value
        - NOTE: although resizing is supported, it is not an exam objective and beyond the scope of our course
        - NOTE: parted does not set filesystem type (and, in all reality, Linux generally does not care, although having the wrong type or none defined may occassionally confuse older applications)

104.1 Create Partitions and Filesystems (Filesystem Types and Creating Them on Partitions)

- Superblock
    - Portion of the disk that can be read and contains filesystem information (size, inode stats, and last time checked)
    - Typically stored on disk in the first sector (and exists in multiple locations to facilitate recovery)

    - First backup block for 'ext' filesystems - 8193 (exam topic!)
- Inode
    - Associated with every file and directory on the system
    - Contains info about the file (except the filename - exam topic!)
    - Includes a list of blocks that make up the file
    - For example - `ls -i /var/log/messages`
        - Will display the inode of the `/var/log/messages` file
    - For example - `stat /var/log/messages`
        - Will display detailed information about the inode associated with the `/var/log/messages` file
    - The number of inodes is set at filesystem creation time and cannot be changed after
        - For example - `df -i`
            - Will show the inodes that are available, used and free on the mounted filesystems
- Filesystem types (exam coverage - there are others)
    - ext2 - Linux extended filesystem (legacy)
    - ext3 - Linux extended filesystem with journaling (logging capability for easier and quicker recovery from disk issues)
    - ext4 - Linux extended filesystem with journaling (including performance enhancements over ext3)
    - xfs - extent filesystem, enhanced performance, particularly on filesystems with many smaller files
    - ReiserFS - one of the first filesystems to introduce journaling and offer dynamic resizing capabilities
    - btrfs - builds on ReiserFS features while adding additional admin features while increasing performance on larger filesystems
    - iso9660 - filesystem specific to CD-ROM
    - udf - filesystem specific to DVD
    - vfat - older DOS partition type (used for compatibility with other operating systems)
- Formatting a drive
    - The process of preparing the size and structures on a partition and applying the filesystem type indicated, that makes it available for mounting and use

- `mkfs`
  - Make filesystem
  - `-t [fstype] [device/partition]` - creates the indicated filesystem type one the chosen disk partition
  - `-b [#]` - create the filesystem with the indicated block size (default - 4096)
  - `-m [#]` - percentage of space reserved for 'root' user
  - `-L [label]` - set the partition volume label
  - `-O [option]` - additional option(s)
    - For example - `mkfs -t ext4 -b 8192 -m 10 -L LargeData -O sparse_super /dev/sde4`
      - Would create an 'ext4' partition, with a block size of '8192' (meaning that the minimum space allocated to ANY file is 8192 - something you may do for a data drive with lots of large files), reserving 10% for root use, and using the 'sparse_super' option (which reduces the number of superblock copies available for recovery)
- `mkfs.[fstype]`
  - Equivalent command for each filesystem type to format the indicated partition
    - For example - `mkfs.ext3 /dev/sda3`
      - Will create the ext3 filesystem on the `/dev/sda3` partition
- `mke2fs`
  - Equivalent command for making an ext2/3/4 filesystem
    - `/etc/mke2fs.conf`
      - Option file for mke2fs behavior
    - `-O [option]` - change the default behavior as configured in the configuration file
- `mkswap`
  - Create swap space out of the indicated partition (or file)
- `mkraid`
  - Creates a RAID array set from the indicated disks
- `mknod`
  - Create 'special' files (i.e. devices)
- `mkisofs` - create ISO filesystem for burning to CD-ROM

104.2 Maintain the Integrity of Filesystems

- `du`
  - Responsible for 'estimating' disk space usage on files and/or directories
  - `-a` (or `--all`) - write counts for all files and not just directories
  - `-c` (or `--total`) - produce a grand total
  - `-h` (or `--human-readable`) - print sizes in readable format (K/M/G/TB)
  - `-s` (or `--summarize`) - display only a summary for any argument
  - NOTE: can be used with file globbing
    - For example - `du -sh /home/user/.bash*`
      - Will provide a human readable summary of each file or directory matching '.bash*' within the /home/user directory
- `df`
  - Reports on file system disk space usage
  - `-a` (or `--all`) - include all filesystems (including 'dummy' filesystems)
  - `--direct` - show stats for a file instead of mount
  - `--total` - print a grand total
  - `-h` (or `--human-readable`) - print sizes in readable format (K/M/G/TB)
  - `-l` (or `--local`) - include only local file systems
  - `-t` (or `--type`) - limit listing to the indicated type
    - For example - `df -lh --total`
      - Will produce a human readable listing of all local filesystems including a total line at the bottom
- `debugfs`
  - Filesystem debugger, can show a vast amount of information about the indicated partition
  - For example - `debugfs /dev/sdb1`
    - Will show a lot of information about any selected file or folder on the indicated drive
  - Interactive (once started)
    - ? - show available commands

    - cd [path] - change to the indicated path (NOTE: must exist on disk debugfs is running on)
    - features - display the filesystem features
    - logdump - display journal contents (if available)
    - ls - shows contents of current directory
    - pwd - shows the working directory
    - open - open a filesystem for debugging
    - stats - show stats for the filesystem
    - undelete - undeletes a file (NOTE: MUST be used immediately after deleting a file before all changes are synced to disk - often fails to find the deleted file on modern distributions as 'sync' is the default filesystem disk setting)
    - quit - exit

- `fsck`
  - Filesystem check
    - `e2fsck` - checks 'ext' filesystem types
    - `reiserfsck` - check 'ReiserFS' filesystem types
    - `dosfsck` - check DOS filesystem types (i.e. VFAT)
  - Uses `/etc/fsab` to check filesystems automatically on boot (those with a '1' in the sixth column in the file), when a filesystem is marked 'unclean' (crashed, corrupt, not cleanly unmounted, etc), a deeper scan will be performed
  - If the filesystem cannot be fixed automatically on boot, you will be prompted to 'CTRL-D' to continue or provide root password to perform a deeper scan (rescue prompt displayed)
  - `-A` - `fsck` will iterate through the `/etc/fstab` file and check all filesystems
  - `-C` - display hash/mark progress bar
  - `-N` - dry run, makes no changes but displays what would have been done
  - `-V` - verbose output (be careful, is VERY verbose and may scroll quickly by)
  - `-a` - just do it, does not prompt for confirmation or any other feedback, runs the check and repar non-interactively
  - `-f` - force a check even if the filesystem reports clean
  - Order of events for `fsck`:

    1. Check inodes, blocks, sizes

2. Check the directory structure
3. Check directory connectivity
4. Check the file/directory refernce counts
5. Check group summary info

- NOTE: only run on unmounted filesystems or the action performed may cause instability or corruption

- `tune2fs`
  - Used to set filesystem parameters after creation
  - `-c [#]` - set the maximum number of times a filesystem can be mounted for an `fsck` to happen automatically
  - `-e [option]` - modifies the behavior of the filesystem with the indicated option (continue, remount-ro, panic)
  - `-g [groupname]` - add the indicated group as potential users of the reserved space on a filesystem (usually reserved for root)
- `xfsprogs`
  - Tools and utilities for XFS filesystems
  - Be AWARE of the following utilities and their general function for the exam!
- `xfs_check`
  - XFS filesystem equivalent to `fsck` for checking filesystem
- `xfs_repair`
  - XFS filesystem equivalent to `fsck` for repairing filesystem
- `xfs_metadump`
  - Creates debugging information that can be used by a third party to aid in recover of an XFS filesystem when the repair has failed
- `xfs_growfs`
  - Used to grow an XFS filesystem (however, it can NOT be shrunk)

104.3 Control Mounting and Unmounting of Filesystems (Mounting Filesystems Manually and Automatically)

- `/media`

    - General purpose 'parent' directory that is often used for 'removeable' filesystems (CDs, DVDs, etc)
- `/mnt`
    - General purpose 'parent' directory that is often used for mounting disk/partitions that are NOT part of the filesystem installation or wthin the 'root' structure of the OS (i.e. a backup or purely data drive)
- `blkid`
    - Used to obtain the UUID (universally unique identifier) for the local disk partitions on the system, will also display disk labels (if they have one)
    - Matches the UUIDs obtained from `/dev/disk/by-uuid`
- `mount`
    - Executed alone, will display mounted filesystems (read `/etc/mtab`, also available in `/proc/mounts`)
    - Command used to manually mount a device with an existing filesystem on a mount point (directory)
    - `-a` - mount all filesystems in `/etc/fstab` (if not mounted, will NOT unmount/remount those that are)
    - `-f` - fake mount all filesystems in `/etc/fstab`
    - `-r` - mount the indicated filesystem in read only mode
    - `-t [fstype]` - filesystem type to mount the indicated device as (can be read by the values in the superblock automatically if the kernel supports that filesystem)
    - `-o [option(s)]` - specify one or more options that are outside the defaults (see `/etc/fstab` column 4 below)
    - `-w` - mounts in writeable mode (default)
- `unmount`
    - Like you would think, unmounts the indicated filesystem
    - For example - `unmount /mnt/data`
        - Would unmount the `/mnt/data` directory from whatever device was mounted there
        - NOTE: a filesystem can only be unmounted if it is not in use
    - `-f` - attempt to forcibly unmount the filesystem (even if it is in use or files are open)
- `fuser`

    - If `unmount` indicates a filesystem is in use, this will tell you which user is using it
    - `-m [mount]` - determine who is using the indicated mount
        - For example - `fuser -m /mnt/data`
            - Would display the mount and any PIDs using it along with a letter 'c' if being used as current directory (in other words, a user is just 'in' the directory)
- `/etc/fstab`
    - Single line mount configuration for local (and remote) filesystems that are to be mounted on boot
    - Contents of the columns are:
        1. The device (i.e. `/dev/sda1` OR `LABEL=data` OR `UUID=44d27f92-d3df-4207-80ea-22830afccf03`)
        2. Mount point (i.e. `/` OR `/mnt/data`, etc)
        3. Filesystem - supported filesystem type (i.e. ext3 OR xfs, etc)
        4. Options - comma separated list
            - noauto - do not automatically mount on boot (prevents removeable devices that may cause issues from being accessed on boot)
            - defaults - common disk option made up of rw, setuid, dev, exec, auto, nouser, async
            - user - only the user that mounts the disk (CD-ROM/DVD/Removeable Disk) can unmount it
            - users - any user can unmount the disk (CD-ROM/DVD/Removeable Disk)
            - ro - mount read only
        5. dump - value of '0' will prevent the 'dump' command from affecting it
        6. fsck - value of '1' will indicate the filesystem should be the first checked

- NOTE: `/etc/fstab` is used for ALL types of mounts, local and remote, remote filesystems are out of scope for Exam 1, but will be covered in Exam 2!

104.4 Manage Disk Quotas

- `quota`
    - Name of package providing disk quota management
    - Limits a user to using only the configured amount of disk space
    - Command to display any quotas that are in place

    - Related commands and/or options for users/filesystems:
        - `quotaon` - turns quota on
        - `quotaoff` - turns quota off
        - `quotacheck` - verify user usage and update saved values
        - `edquota` - edit user quota amount
        - `aquota.user` - binary containing user quota info
        - `aquota.group` - binary containing group quota info
        - `usrquota` - option in `/etc/fstab` for user quota enablement
        - `grpquota` - option in `/etc/fstab` for group quota enablement
        - `repquota` - used for generating reports on quota usage
- quota types
    - Soft limit - warnings when the limit is exceeded will appear until the end of a grace period
    - Hard limit - cannot be exceeded
    - Grace period - amount of time the soft limit can be exceeded
- NOTE: setting quotas on root is not recommended
    - Typically, quotas are set on the `/home` filesystem (see the earlier lesson on setting up your disk layout during installation)
- Enabling a quota for users and groups
    - `/etc/fstab` entry example
        - `/dev/sdc1 /mnt/data ext4 defaults,usrquota,grpquota 0 0`
            - On boot or filesystem remount, quotas can be enabled
    - Update the 'aquota.*' files
        - For example - `quotacheck -avugc`
            - Will enable user and group quotas for all filesystems indicated in the `/etc/fstab` file and mounted with those options enabled (check with the 'mount' command)
    - Edit quota settings for the intended user
        - For example - `edquota -u user`
            - Will open up the default editor (vi or nano in most cases) with the editable quota

values

- Blocks (presently in use), soft, hard, inodes(presently in use), soft, hard (soft and hard each apply to the directly antecedent 'blocks' or 'inodes')
  - Block limits (soft, hard) will set SPACE limits
  - Inode limits (soft, hard) will set NUMBER OF FILE limits

- Verify the user's new limits
  - For example - `quota user`
- Enable quotas for the filesystem
  - For example (our `/mnt/data` example) - `quotaon /mnt/data`
    - Will enable the configured quotas from the aquota.* files and configured filesystems for the indicated users
- Set grace period for 'soft limits'
  - For example - `equota -t`
    - Will open editor to set the grace time in 'days,hours,minutes, or seconds' for both block and inode limits
- Report on quota usage
  - For example - `repoquota -a`
    - Will create a text report on quota usage for all defined users on enabled filesystems

104.5 Manage File Permissions and Ownership

- UID
  - User ID (unique for each user on each system)
- GID
  - Group ID (unique for each group on each system)
- Permissions are set in 3x3 groups for each file and directory (when provided as a symbolic value - special/sticky bit notwithstanding, see below)
- Permissions are set in 3 numbers for each file and directory (when provided as a numeric value - special/sticky bit notwithstanding, see below)
  - Symbolic permissions
    - [owner rwx][group rwx][everyone rwx] - read/write/execute permissions for user, group, and everyone

- numeric permissions
    - [owner #][group #][everyone #] - read/write/execute permissions for user, group, and everyone

- Permission mapping (file listing examples)
    - Symbolic
        - r - read (view directory contents but not file attributes)
        - w - write (add and delete files, including files that you do not own if you have write privileges on their directory)
        - x - execute (can navigate to the directory [`cd`])
    - Numeric
        - 4 - read (view directory contents but not file attributes)
        - 2 - write (add and delete files, including files that you do not own if you have write privileges on their directory)
        - 1 - execute (can navigate to the directory [`cd` or `changedir`])
    - NOTE: permissions can be set on files or directories using EITHER numeric or alpha notation
        - For example (symbolic) - `-rwxr-xr--  root admins  2456 Mar 25 01:05 myfile.txt`
            - Would indicate that the file `myfile.txt`, owned by root, has owner (root) permissions of 'read/write/execute', group (admins) permissions of 'read/execute' and everyone else permissions of 'read'
        - For example (numeric) - `-754  root admins  2456 Mar 25 01:05 myfile.txt`
            - Would indicate that the file `myfile.txt`, owned by root, has owner (root) permissions of 'read/write/execute (4+2+1)', group (admins) permissions of 'read/execute (4+1)' and everyone else permissions of 'read (4)'
        - In the above listing (symbolic), the initial character before permissions, indicates the object type as follows:
            - - - normal file
            - l - symlink
            - b - block device (like a disk)
            - c - character device
            - d - directory

    - Hierarchy of permissions
        - If user is the owner of the file - the OWNER permissions apply
        - If user is a member of the group owner of the file - the GROUP permissions apply
        - Else the EVERYONE permissions apply
        - NOTE: only ONE set of permissions apply, the first one the user qualifies for
- `chmod`
    - The tool used to manipulate the permissions of files and directories
        - For example - `chmod 755 myscript.sh`
            - Would set the permissions as 'user read/write/execute', 'group read/execute' and 'other read/execute'
    - Can use symbolic values one category at a time (u - user, g - group, a - all) for each permission
        - For example - `chmod g=rx myscript.sh`
            - Would set the permissions as 'group read/execute'
            - NOTE: can also use +/- to add or remove permission values (i.e. g+r-w+x for 'read/execute')
            - NOTE: can set multiple categories of permissions (user/group/all) by inserting a comma between each (i.e. u=rwx,g=rw,a=r)
    - `-R` - recursive, allows you to apply permissions to an entire directory structure, including all other files and directories in it
        - NOTE: be careful in its application, where you place a trailing slash (or not) makes a difference
            - For example - `chmod 744 -R /home/user/data`
                - Sets permissions for the '/home/user/data' directory and all the files and directories within it
            - For example - `chmod 744 -R /home/user/data/*`
                - Sets permissions for the all the files and directories within `/home/user/data` but does NOT change the permissions of the `/home/user/data` directory itself
    - `-c` - report which files were changed
    - `-v` - show all files
    - `-h` - change only symbolic links, not the original
    - `-f` - ignore/don't display error messages

- Special permissions
  - Solves the problem of each file/directory only being able to belong to a single group at any one time
  - Useful in collaborative configurations where you have users in different groups, but some users are members of more than one group
    - Primary group (the first one you are a member of)
    - Secondary groups (your user account appears in `/etc/group`)
  - The 'special permission' representation of 's' takes the place of the execute bit in a list (i.e. rwx becomes rws)
  - SUID - Set User ID
    - Permits a user to access/run a program as if they were the OWNER of the program
    - 4 - value of the SUID permission
    - u+s - symbolic notation
  - SGID - Set Group ID
    - Provides group ownership of any NEW file created in a directory to the group owner of the directory (or when applied to a file, permissions to access/run a program as if they were group owner)
    - g+s - symbolic notation
  - Sticky bit
    - Used to keep users that do not own a file from deleting files in a directory they otherwise have group/all write permissions in (changes the 'write' permission for a directory - delete file can only be done by the owner of the file, the owner of the directory or the root user, even if the user would normally have full write permissions to the directory the files are in)
    - a+t (or +t) - symbolic notation
- `umask`
  - `/etc/bashrc`
    - System wide setting applied to all users unless specified in the local user .bashrc
  - `~/.bashrc`
    - Can be set here for the indicated user, overriding the system setting
  - A method of modifying the default permissions applied to files and directories a user creates
  - Default (no umask) permissios

    - Files - rw-rw-rw- (666)
    - Directories - rwxrwxrwx (777)
- Executing the command `umask` will provide the current default value for the user
    - For example - `umask` (may display - 0022)
        - for this user, this means
            - 0 - special permissions are masked - NOTE: will always be 0 (removed)
            - 0 - user permissions are masked (removed)
            - 2 - group permissions are masked (removed)
            - 2 - other permissions are masked (removed)
        - This would mean that any file created would have default permissions of '644' and directories created by that user would have default permissions of '755'
- default permissions for the active user can be changed at any time temporarily
    - For example - `umask 0027`
        - Would subsequently set created files to 640 and directories to 750 for that user
        - NOTE: this will be reset on session end to default on next session

104.6 Create and Change Hard and Symbolic Links

- Symbolic link
    - Effectively, a shortcut from one file to another
    - Small file (containing its own inode and a path to the original file)
    - Can 'span' filesystems and drives (remote or local) because it has an independency inode
    - Permissions cannot be modified, changes would be (attempted to be) applied to the original (provided the right permissions existed for the user running the command)
    - Removing a symbolic link (sometimes called a 'soft' link), only removes the small link file and associated inode and does not affect the original
    - Removing the original file does NOT remove the link but leaves the link 'orphaned'
- Hard link
    - Adding an additional name to a file or directory that actually points to the ORIGINAL file or directories inode (it actually is the same data on disk)
    - Deleting any ONE of the existing hard links just deletes the reference to the inode, as long as

any other hard link exists (or the original), the data on the inode remains

- As a result of each hard link sharing the same inode however, hard links can NOT 'span' across filesystems or disks (local or remote)
- Completely deleting a file with hard links requires the removal of ALL hard links
- Hard links are useful in make very important files 'permanent' (i.e. placing a hard link in an admin accessible only directory to a directory or file that should never be completely deleted)

- NOTE: remember that the filename and location are NOT stored in an inode
- `ln`
  - Command used to create hard or symbolic links
  - `-s` - creates a symbolic link
    - For example - `ln -s myscript.sh yourscript.sh`
      - Creates a symbolic link called 'yourscript.sh' pointing to 'myscript.sh'
    - For example - `ln myscript.sh yourscript.sh`
      - Creates a hard link (default) called 'yourscript.sh' point to the data in the inode pointed to by 'myscript.sh'
- `ls`
  - `-i [filename]` - can be used to find all the links for a file

104.7 Find System Files and Place Files in the Correct Location (The Filesystem Hierarchy Standard)

- Filesystem Hierarchy Standard (FHS)
  - In short, it defines the directory structure and contents in Unix-like operating systems (which, of course, applies to Linux).
  - This standard is maintained by the Linux Foundation and (as of March 2017) is currently on version 3.0.
  - Most distributions voluntarily choose to follow the FHS and remain in compliance as updates are released, however, some distributions deviate slightly in some ways.
- Directory Structure
  - The primary specification is that all files and directories appear under the 'root' directory (`/`), even if stored on different physical or virtual devices (although some directories do not have to exist at all if the dependent subsystems do not exist, such as X Windows).
  - `/` - Root and root directory of the FHS

- `/bin` - Command binaries intended for all users, must be available in single user mode
- `/boot` - Boot loader files (kernel, initrd, etc)
- `/dev` - Device files
- `/etc` - Host-specific configuration files, no binaries
- `/etc/opt` - Config files for packages stored in /opt
- `/etc/sgml` - Config files for software that processes SGML
- `/etc/X11` - Config files for X Windows
- `/etc/XML` - Config files for software that processes XML
- `/home` - User's home directories
- `/lib` - Libraries essential for binaries in /bin and /sbin
- `/lib[qual]` - Alternate format libraries (i.e. 64bit)
- `/media` - Mount point(s) for removable media (CD, DVD, etc)
- `/mnt` - Temporarily mounted filesystems
- `/opt` - Optional application software (often 3rd party)
- `/proc` - Virtual filesystem providing process and kernel information as files (corresponds to procfs)
- `/root` - Root user home directory
- `/run` - Run-time variable data, information about the running system since last boot
- `/sbin` - Essential system binaries
- `/srv` - Site-specific data server by the local system
- `/sys` - Contains information about the devices connected
- `/tmp` - Temp files, not preserved between reboots
- `/usr` - Secondary hierarchy for read-only user data
- `/usr/bin` - Non-essential command binaries for all users
- `/usr/include` - Standard include files (C/C++ .h files for example)
- `/usr/lib` - Libraries for the binaries in /usr/bin and /usr/sbin
- `/usr/lib[qual]` - Alternate format libraries (i.e. 64bit)
- `/usr/local` - Tertiary hierarchy for local data, specific to host

- `/usr/sbin` - Non-essential system binaries
- `/usr/share` - Architecture independent shared data
- `/usr/src` - Source code (i.e. kernel)
- `/usr/X11R6` - Older X Windows configuration (optional)
- `/var` - Variable files whose content is expected to change during normal system use and operation
- `/var/cache` - Application cache data used as a result of time-consuming I/O or calculations. Can be deleted without loss of data
- `/var/lib` - State information
- `/var/lock` - Lock files that keep track of resources in use
- `/var/log` - Log files
- `/var/mail` - Mailbox files
- `/var/opt` - Variable data from add-on packages stored in /opt
- `/var/run` - Run-time variable data (replaced by /run in 3.0)
- `/var/spool` - Spool for tasks waiting to be processed (print queues for example, /var/spool/mail for another)
- `/var/tmp` - Temporary files to be preserved between restarts

104.7 Find System Files and Place Files in the Correct Location (Using find, locate, updatedb, /etc/updatedb.conf, whereis, which, and type)

- `locate`
  - Find files by name, returning results for the indicated string in any portion of the path or filename the string exists in (important for the exam!)
  - Can use regular expressions, but is often used to locate a known filename or partial name including globbing
  - Uses the `locate` database as updated by the `updatedb` utility
  - `-i [string]` - ignore case on the file being searched for
- `/etc/updatedb.conf`
  - The configuration file for the `updatedb` command managing the `locate` database(s)

    - Contains full or partial filenames and extensions to ignore or 'prune' from the database updates
- `which`
    - Will indicate which command (including the path contained in) will run if indicated command is run without an absolute path provided
    - Can be important if there are multiple potential versions of an application as this will display which path contains the program FIRST in the PATH variable
    - `-a [command]` - display all matches in the PATH, in the order they would be run/appear
- `whereis`
    - Provides 'more' information about a command than `which` which only provides the first matching path for it
    - Provides the name of the command, all locations in PATH and any man page associated with it
    - `-b` - searches for binaries
    - `-m` - searches for manual entries (man pages)
    - `-s` - search for sources
    - `-u` - unusual or undocumented entries
- `type`
    - Will indicate whether a command has been 'extended'
        - For example - `type ll`
            - If you had created an alias for this, you may see something like 'll is aliased to 'ls -al --color=auto"
    - `-a` - list all variants of the indicated command (i.e. `type -a ls`)
- `find`
    - As previously discussed, the best method of finding files in the system (as it does not rely on a database), but can be expensive in terms of performance
    - `-perm -[####]` - find matching files by indicated permission
        - For example - 'find /usr/bin -perm -4000 -exec cp {} /mnt/data/binbkup \;'
            - Would find all SUID enabled files in the `/usr/bin` directory and copy them to the `/mnt/data/binbkup` directory

APPENDIX A - Configuration File Samples

/etc/yum.repos.d (Configuration File Sample - EPEL Repository - epel.repo)

```
[epel-debuginfo]
name=Extra Packages for Enterprise Linux 6 - $basearch - Debug
#baseurl=http://download.fedoraproject.org/pub/epel/6/$basearch/debug
mirrorlist=https://mirrors.fedoraproject.org/metalink?repo=epel-debug-
6&arch=$basearch
failovermethod=priority
enabled=0
gpgkey=file:///etc/pki/rpm-gpg/RPM-GPG-KEY-EPEL-6
gpgcheck=1
[epel-source]
name=Extra Packages for Enterprise Linux 6 - $basearch - Source
#baseurl=http://download.fedoraproject.org/pub/epel/6/SRPMS
mirrorlist=https://mirrors.fedoraproject.org/metalink?repo=epel-source-
6&arch=$basearch
failovermethod=priority
enabled=0
gpgkey=file:///etc/pki/rpm-gpg/RPM-GPG-KEY-EPEL-6
gpgcheck=1
```