Sample Study Guide Questions and Answers, Introduction to Linux

1. **Describe Linux software.**

   Linux is the latest in "free" operating systems that began with Bell Laboratories version 6 UNIX (1977), Douglas Comer's Xinu (1983), Andrew Tanenbaum's Minix (1986), and FreeBSD (1990). Linux initially identified the operating system (OS) kernel proper, but now refers to any one of a number of software distributions that have bundled other "free" software including the GNU Free Software Foundation and the XFree86 Project software. These Linux distributions contain up to 2 gigabytes of data and include the OS kernel, internetworking, X-window GUI, utilities, GUI applications, documentation, and complete source code. Some estimate the installed Linux base at 7.5 million CPUs.

2. **Explain what is meant by "Linux ideals."**

   In addition to software, Linux also refers to a world-wide community of private companies, governmental agencies, independent professional programmers, and computer enthusiasts that number in the 10's of thousands (some say 250,000) whom communicate via the Internet. The Linux community is unique in that it has no official organizational structure other than the agreement to share an open software system.

   The Linux community believes that the profession of programming is just as much about the dynamic process of people solving problems as it is about the sale of static software "products." The Linux community, therefore, takes a revolutionary position in today's computer software industry by saying that there is a set of fundamental software tools that cannot be owned (restricted) by any one company or individual. Thus, the computer professional now has at his or her disposal a rich set of system and graphic application tools that permit the crafting of software solutions that go beyond the basic software tools presently employed by the programmer community.

   In other words, the Linux community seeks to raise the standards of programming performance for the whole profession by saying that there are many challenges before us and we cannot (will not be able to) solve these advanced and complex problems with every professional reinventing and controlling their own version of basic software tools. Instead of rewriting the same code over and over, let's agree that, as a group, we can all be more efficient and productive if we share the operating system, internetworking, GUI, and basic applications.

   As an aside, it turns out that these Linux ideals are practiced (to a lesser extent) by all industries that agree upon shared information or "standards." The automotive industry, for example, agrees to use the steering wheel, brakes, gas pedal, etc. in the same form.

3. **What is meant by the term "free" software.**

   As the adage goes "there is no such thing as a free lunch," and there is no such thing as free software. Linux is free in the sense that there is no fee for the open software. Linux, however, has strict licensing agreements (the GNU General Public License) and an implicit (built-in) cost for labor and distribution fees regardless of how the software is installed or upgraded. So the "free" in free software is really freedom to create software. The real cost of Linux software occurs when a highly-paid professional programmer improves the software and releases the improvements (effort) free-of-charge to the Linux community.

4. **Raising software industry standards sound great, but we all know it's money that makes the world go around. So what really motivates these people?**

   The August 10, 1998 Forbes cover story, featured Linux and the freeware programmers Linus Torvalds, Richard Stallman, Larry Wall, Brian Behledorf, *et. al.* (Article sidebar and issue table of contents). Scott McCormack (article author) addressed the motivation conundrum by saying that these programmers are more interested in status than money. By contributing to Linux, programmers earn the respect of their peers and they can even make a "decent" living using the skills they honed in the development of Linux.

   The argument has merit, but this explanation does not take into account the motivation of tens of thousands of other programmers that work very hard on Linux and will never achieve the same status of Torvalds or Stallman. A more parsimonious explanation is that these people need (want) sophisticated software (an OS, networking, GUI etc.) to build even more sophisticated applications. But the applications always interact in unpredictable ways with the underlying support software and, therefore, they must have complete **control** over their software tools to build these advanced applications.

   And it is the **control** of software that traditional software "product" people do not (want to) understand. As corporations begin to understand the resulting productivity and profitability of controlling your own software from the ground up, Linux will be the OS of choice, and OSs as software products will disappear.

5. **So where is all this "sophisticated" and "advanced" software? Linux does not run any "good" software.**

   If by "good" software you mean traditional PC software, it's true, you will not see Microsoft products on Linux. (Aside from the fact that Linux can emulate MS-DOS, most of MS-Windows 3.x, and about half of MS-Windows 9x 32-bit software, it is not clear how the Linux "desk top" environment will evolve over the next few years.) Most likely, Linux will establish its own niche using various native office suite products. In spite of the fact that many PC software houses (such as Corel) are building Linux versions of their software, it is difficult foreseeing Linux offering a serious challenge to Microsoft products.

   Although a GUI, such as KDE, may make Linux look like a PC, make no mistake, Linux "is" Unix but with more functionality than other Unix implementations and a fresh set of GUIs! Linux is a very successful Unix hybrid that integrates System V and BSD versions and, as a result, Linux accepts almost all traditional Unix applications with little or no modification. An interesting side effect of porting Unix software to Linux, is that Linux drops Unix software prices by a factor of 100.

In addition to the thousands of Unix applications that run on Linux, programmers integrate and "mask" Linux with their software product. If you are providing a end-product, then do not bother the end-user with the operating system. Simply start your application upon bootup and use Linux behind the scenes to provide services to the application.

i. The USPS handwriting scanners.
ii. Northern Italy transportation authority.
iii. US military.
iv. Pharmacy data management system.

6. **Explain how Linux differs from proprietary operating systems.**

First, there is no charge for the Linux software as there is for a typical proprietary OS. With a commercial OS you receive a static product (with the possibility of patches) and a license that restricts the number of CPUs and the number of users allowed on the CPU. With Linux you forget about licenses and warranties and become part of an interactive process that continually strives to develop a better operating system. You can stop this process at any time by freezing the current version of Linux on a given box or you can continue to participate in the addition of new software components and software upgrades.

Second, the simple truth is that Linux offers MORE functionality than ANY other operating system (See the internal Microsoft analysis of Linux). It has preemptive multi-tasking, multi-user kernel with centralized and dynamic virtual buffer management, demand loading shared libraries, shared copy-on-write executables, loadable kernel modules, symmetrical multi-processing, and sophisticated networking. Besides its native fast file system, Linux provides transparent access to a host of other hard disk formats such as Windows-NTFS. Through the XFree86 Project, Linux provides an advanced GUI with graphic applications, through the GNU Free Software Foundation, Linux also offers advanced compilers and advanced Unix utilities.

Third, Linux is fast. Performance has always been a key concern for Linux developers. Depending on the benchmark, Linux can be four to ten times faster than Sun's Solaris. MS-Window users always marvel when they see Linux running twice as fast as Windows 9x or NT. Linux is so fast that old 486 CPUs are usually brought out of closets, installed with Linux, and placed around as network firewalls and miscellaneous network servers.

7. **Define "virtual consoles" and explain their use and significance.**

Virtual consoles first appeared in Minix and are also in FreeBSD. Virtual consoles provide access to multiple login sessions regardless of the application executing in the current session. At the lowest level of the Linux operating system, 1 of four to twelve consoles can be selected by typing <alt>Fx, where <alt> is the depression of the "Alternate" key and Fx is one of the "Function" keys. Thus, the user at the console may quickly save the contents of one login session and switch to an another. This lets the user exploit the multi-tasking OS by executing separate tasks on the various virtual consoles. If desired, the user may even switch from the GUI to a fresh command line session by using the <alt>Fx keys. But most significantly, virtual consoles permit the user to switch from a "locked up" session to another screen and, in turn, kill the locked process without having to reboot the operating system.

As an aside. Linux is also known for its stability. Once installed and setup. Linux systems run for years. Regardless of the number of users and processes, the only time a Linux system reboots is after a power outage.

8. **Describe the distinction between WYSIWYG and mark-up word processing and how this impacts the Linux user.**

   As part of its Unix heritage, Linux provides a long line of typesetting or mark-up languages (e.g., nroff, groff, TEX, LATEX, and HTML). In these word processing schemes, text creation and display are separate processes in which the user writes the information in a plain text file and directs how the text is to be displayed with text-based formating commands for use by the subsequent display processor. The PC user, on the other hand, usually employs a What You See Is What You Get (WYSIWYG) word processor where the text is displayed as it is entered. To accomplish this, the WYSIWYG word processor must embed non-printing binary information into the text file.

   This distinction reflects a fundamental design issue for Unix (Linux) users since Unix is predicated on employing all of its utilities (support programs) for processing text files. Embedding binary information into the text file allows only one program, the word processor that created it, to access the text.

   A user that moves between the PC and Unix environments must be aware of this distinction and use the various programs accordingly. For example, a user familiar with Microsoft Word or Corel's Word Perfect may continue to work in the same WYSIWYG environment on Linux. But if files are to be used elsewhere in the system, they must be exported as "MS-DOS Text" files in which each paragraph is transformed into one long line.

   From a casual use perspective the two designs achieve the same end, but over time the user learns that a mark-up design does offer advantages over a WYSIWYG design. Mark-up text can be created with any number of editors, text files can be easily processed by other applications, and mark-up actually provides more control over the display than the set of functions provides inside a given WYSIWYG word processor. This process is similar to the transition that experienced users have when they migrate from mouse action to keyboard short-cut keys to speed their interaction with the machine.

9. **Describe the main entry points into Linux assistance.**

   The main Linux information pages are:

   - The Linux Documentation Project (LDP). The LDP provides links to a wealth of Linux documentation, including HOWTOs, FAQs, and online books.
   - Linux Applications & Utilities List provides an incomplete list of programs for Linux, but it has the major applications.
   - Your news reader can provide access to the Linux newsgroups. The entry point is "comp.os.linux.x" where "x" is one of 10 subgroups.
   - DejaNews provides access to old Usenet postings where someone may have posted solution to your problem.

   The Official Linux Web Page is a well-organized site designed for basic information on Linux, and for users to keep up on news. Its subdirectories include:

- Linux distributions
- Compatible computers
- How to install Linux
- Year 2000 (Y2K) compliance

10. **Describe the "Linux Software Release" and give four examples with tradeoffs of each example.**

    The Linux community has no "official" software release. Instead, there are various "distributions." There are currently seven distributions generally available and an eighth projected distribution. Four of the most common distributions are: Redhat, Debian, Slackware, and S.u.S.E.

    - The Redhat distribution is mass-marketed and, therefore, the most commonly available. The Redhat goal is to provide an easy-to-install, pre-configured system complete with many applications, including a Web server and X-Windows environment. Redhat is primarily a CD-ROM distribution to be sold over the Internet and in bookstores for $49.95. Although one can download Redhat from the Internet it usually requires about three days and a lot more time than using the CD-ROM version.
    - The Debian distribution is the oldest and probably the least well-known. Debian can be found on CD-ROM, but was designed to be distributed over the Internet. The primary goal of the Debian distribution is to provide an easy upgrade technique as well as high-level security components and is, therefore, preferred by Linux "power users." With Debian, one installs a basic Linux kernel and then uses Debian select (dselect) utility to download the various components or upgrades over the Internet. As one installs a Debian distribution and selects from among the 1,200 plus software packages available over the Internet, one begins to truly sense the magnitude of the Linux-wide effort.
    - The Slackware distribution is probably the second-best known Linux distribution. Its bundled software has the Carnegie Mellon Andrew toolkit which supports interactive multimedia, like Redhat, Slackware is a CD-ROM based distribution.
    - S.u.S.E. (pronounced su-sa) personnel work closely with the XFree86 Project and its distribution has the most recent modifications to, and device drivers for, the X11R6 windows system. The S.u.S.E. CD-ROM distribution also has the most bundled applications with a full installation requiring over 2 GB of disk space.

11. **Describe the general steps in installing Linux from three types of media.**

    0. CD-ROM -- Offers the most ease and convenience when installing Linux. If the BIOS permits, the CD may be booted and the system installed from the CD-ROM. Typically, however, the BIOS cannot directly boot the CD-ROM and boot floppies must be constructed by employing a DOS-based program "rawrite" to copy a minimal Linux kernel image file (and install program) onto the floppy disks. After the floppy disk boots, the hard disk is partitioned and formatted with Linux native file systems and a swap partition. Next, the install program can read Linux from the IDE or SCSI CD-ROM and copy it to the hard disk. Removing the floppies allows Linux to be booted from the hard disk.
    1. LAN-based Hard Disk -- When installing from the Internet, a local (LAN) hard disk has been configured with the Linux software. The first step is to format floppy boot disks from the distribution and use these disks to launch the initial version of Linux. Next the hard disk is partitioned and formatted with Linux native file systems and a swap partition. After the OS and executable versions of the applications are down loaded from the LAN, Linux can be booted from the hard disk.
    2. Floppy Disk -- A third but painful alternative is to store the distribution on floppy disks by repeating the above steps of booting and repartitioning the hard disk. But this time you must read a minimum of 50 floppies onto the hard disk before you can reboot the system.

12. **Describe the basic steps in any Linux installation.**
    0. Disk allocation -- Decide on single Linux install or a "dual boot" with other operating systems. Decide on the number and size of Linux partitions.
    1. First boot -- Boot the minimal kernel and install program (usually from the boot floppies).
    2. Create Linux partitions -- Use the fdisk (or disk druid) program to write native and swap partitions.
    3. Copy software -- Copy the software and (default) configuration files onto the newly partitioned hard disk and re-boot. Sometimes the newly created hard disk is updated with files from an older Linux or Unix box and the hard disk is moved to the older box (but see the next point for a caution).
13. **Contrast the BIOS versus the Linux Kernel and explain how they interact with one another.**

    All of today's computers employ firmware or Read Only Memory (ROM) that the CPU executes upon power-up. The firmware performs basic diagnostics and allows the user to configure basic system functions like the time-of-day. The PC firmware is referred to as the Basic Input Output System (BIOS) and also contains routines to access the hard disk and other peripherals. The Linux loader (LILO) uses the BIOS to boot Linux or another OS. But once the Linux kernel takes over, it no longer uses the BIOS to access the hard disk or other peripherals. There are many reasons for this, but the key problem is that the BIOS does not use interrupts and, therefore, could not support a preemptive multi-tasking OS.

14. **Explain the conflict and solution of installing GB hard disks into a Linux box where the older BIOS can only access 504 MB (Intel 486 CPU, 33 MHz, with IDE controller).**

    Even though the Linux kernel may access large gigabyte (GB) disk drives, the older Basic Input Output System (BIOS) read-only memory may be only able reach the first 504 MB (up to cylinder 1023) of disk space. Moreover, the Linux loader (called LILO) program must use the BIOS to read the master boot record, the root file system, and the "vmlinuz" compressed kernel image. This means that the kernel image must be within the 1st 504 MB from the start of the hard disk.

    The best way to solve this problem is to make the first partition small (10 MB) and mount the small partition as "/boot" for kernel images, then create a very large "root" partition for everything else. To do this, just have the distribution package create and mount the small partition as `/boot` and let it copy the kernel images into that default directory.

    A second alternative is to make the root partition less than 504 MB to allow the BIOS (LILO) to reach the kernel image anywhere within the fully-indexed partition, but this leads to possible problems described below.

    A note of caution. If you move the hard disk from one box to another and reboot Linux, be aware that the BIOS disk settings (heads, cylinders, and sectors) must be the same or LILO will not be able to find the vmlinuz kernel image.

15. **Define "partition" and describe the types "primary," "extended," and "logical."**

    In the old days (early 1980's and before), the operating system (OS) had no difficulty reaching any part of the hard disk. But as disk sizes grew by a factor of tens and hundreds, they soon out stripped the maximum size of the file system indexing mechanism. The solution was to let the firmware (the BIOS) to logically divide the hard disk into smaller logical drives that the OS file system could reach. Thus, OSs have no knowledge of a disk partition. Even though the re-partition program (fdisk) runs in the OS, it only communicates with the firmware (BIOS).

Primary, extended, and logical partitions are specific to the PC BIOS and part of the first disk sector (physical record) or master boot record (MBR). A primary partition is one of the four originally defined table entries in the MBR. One of the four must be marked "active" so that the MS-DOS boot code will know on which partition to find the secondary OS boot program. The BIOS also supports a user-defined extended partition table (more sectors) in which any number of logical partitions may be defined.

As an aside, LILO (as well as others) will ignore the "active" flag and allow the operator to choose among partitions to decide which OS will be booted.

16. **Describe the Unix multi-volume file system and its historical configuration.**

Unix employs a logical hierarchical view of all volumes (*i.e.*, a partition or separate hard disk) with the boundaries among volumes being transparent (invisible). Volume boundaries can only be seen by invoking a separate utility (called mount), examining a system file (/etc/fstab), or invoking a utility to report the amount of free disk space on each volume (called df).

Historically, Unix exploited multiple disk drives by separating independent files among the drives. For example, system programs (/bin) would be held on one drive, user files (/usr) would be stored on a second drive, swapped processes (swapped blocks did not have a file system name) would be saved on a third drive, and temporary files (/tmp) used in editing, compilation, etc. would be placed on a fourth drive. In this way the file manager would automatically overlap seek requests among disk drives since while waiting for one drive, another process would be scheduled that would request information from a second drive and so on.

A second reason for multi-volume file systems is that hard disk sizes have grown faster than file system managers. It is not unusual for Unix file managers to be unable to index the whole hard disk and, as result, have to divide the disk into two or more partitions.

17. **Explain how Linux has expanded the multi-volume Unix concept to include a "multi-file system" concept.**

Not only does the Linux "mount" utility show which volumes are mounted on which directories, it also employs a "file type" argument when mounting a volume that subsequently allows the selection of one of many file managers to access the volume. Thus, the native "ext2" file manager will read from /usr while the "ISO9960" file manager is invoked transparently to read the contents of /cdrom, and the "msdos" file manager is automatically invoked to read from the /floppy directory.

As an aside, some administrators install Linux on one partition and MS-DOS on other partitions. Linux can directly view these MS-DOS files when the partition is mounted with the argument `-t msdos`.

18. **Describe the rationale behind typical multi-volume Linux partition configurations.**

Linux has the ability to access very large hard disks (four terabytes) and "non-native" volumes as well. Linux configurations typically employ one or two hard disks and ignore the performance tuning multi-volume principles mentioned above.

A typical Linux configuration only defines a root ("/") and swap partitions. The swap partition is not a file system since it does not have a file manager. Instead, the swapper process discovers the partition at system boot time and accesses this area of the drive as an array of logical blocks that are moved between main memory and the hard disk. The root partition must hold the OS image (vmlinuz) and may contain the remainder of the disk space (assuming there are none of the BIOS constraints mentioned above).

On machines with older BIOSs, a small (less than 500 MB) partition is defined for the boot image "/boot." Then, a second root "/" partition is defined to hold the remainder of the disk.

You can create one large root partition on old BIOS machines if you direct LILO to boot from the floppy disk instead of the hard disk. Linux can still boot the kernel in less than 30 seconds from the floppy disk.

There are many claims for **having** to use multiple volumes (partitions) in a Linux system (such as a safety backup), but these claims do not have any proven merit.

19. **Explain the problems encountered with multiple partitions on one hard disk.**

    Usually the separate partitions are: /, /home, and /usr. The problem is that there are dynamic directories scattered around the file system. For example, the directories /tmp, /usr/tmp, /usr/local/ftp/pub/incoming, /var/spool/mail, and /var/log may grow at unexpected times. The result is that one partition fills up and results in the killing of the application while there is plenty of room elsewhere on the hard disk.

20. **Explain how the number of users and servers interact with partitions to create problems with a "mature" Linux box. Give a solution to the problem.**

    If the root partition is made too small (say 100 MB), unforeseen problems will arise with multiple users and web servers. One problem is that incoming mail is spooled in /var/spool/mail where many users will leave the mail until their mail file grows to several megabytes. A second problem is the system log files in /var/log may grow to large sizes, the result being lost mail and the loss of critical system logging information.

    Even though the above applications (and others) can be reconfigured to write files into the /usr directories, the simplest solution is to create a third partition for "growth" files. For example, /var could be a separate partition.

21. **Define "swapping" and contrast a swap partition versus a swap file.**

    Swapping occurs when there are more processes to be executed than there is physical memory to hold all of the processes. After a typical Redhat distribution install, about 45 processes are loaded. Some are suspended while others are ready-to-run, but only one process can be running. The 45 processes require about 14MB of memory, plus an additional one to 16 megabytes of memory required for the data cache. If the system has eight or 16 MBs of physical memory, then some of these processes or data cache blocks will have to be swapped.

A swap partition is a reserved area of disk space that the swapper process discovers at boot time and uses to automatically save processes and data that could not otherwise fit into physical memory. A swap file is a file name that has been specified in the "swapon" utility (command) that directs the swapper process to use the file in place of the swap partition (or in addition to a pre-defined swap partition). Swap areas are usually defined as partitions since the swapper can directly access the partition faster than requesting blocks from the native file manager, but the swapon command provides flexibility and quick reconfiguration.

The exact amount of total space required is more an art than a science since it depends on the number of users and the memory requirements of programs they run. Swap partitions are limited to 128 MB, but swap files may use any amount of free space. If desired, up to 16 swap partitions may be defined. In our experience, it is rare to see a system with 64 MB of RAM or greater which require much swapping.

22. **Explain how too much main memory (RAM) could reduce performance in Linux or any other operating system.**

The problem arises from limitations of the motherboard cache design. To limit cost, manufactures will use less "tag" RAM than can be addressed by the SIMM sockets. For example, it is not unusual for a motherboard to only have cache tag addresses that go up to 64 MB, but the SIMM sockets may hold up to 128 MB of RAM. In these cases, memory access time will dramatically increase as programs load above the 64 MB boundary (no cache loading will occur). If the program is CPU bound, it would probably be more efficient to swap the the program to hard disk and reload it to relatively high-speed cached low memory addresses.

Perhaps related to this problem, the Linux kernel, on bootup, will not "autosize" more than 64 MB of RAM. If the system has more memory, the kernel must be explicitly told with the "append=xxxM" argument where xxx is the megabytes of RAM (see below).

23. **Assuming the decision of how partition the hard disk has been made, describe the five steps required to boot linux from the hard disk.**

. Boot the installation media (floppy).
I. Use fdisk to create the partitions.
II. Use mke2fs and mkswap to lay down the second level disk format.
III. Unarchive the distribution software onto the hard disk.
IV. Run LILO to install LILO into the MBR.

24. **Define "LILO" and explain how to interact with it.**

The Linux loader (LILO) is read from the first sector of the Linux hard (or floppy disk) partition by the firmware BIOS. Immediately after it's loaded, LILO checks to see if any one of the <Shift>, <Control> or <Alt> keys are depressed or if the <CapsLock> or <ScrollLock> key has been set.

If none of the keys are depressed, LILO boots the default boot image (or the image specified with the DEFAULT variable). If a delay has been specified in the LILO configuration file, LILO waits until the interval has passed before continuing the default boot process.

If a key has been depressed, LILO displays the "boot:" prompt and waits for the user to type the name of a boot image (i.e. Linux kernel or other operating system). A selection of images may be obtained by pressing <?> or <Tab>. If <Enter> is pressed and no file name has been entered, the default image is booted.

For a more complete and up-to-date list of boot options consult Paul Gortmaker's BootPrompt-HOWTO.

25. Describe the three types of arguments that LILO accepts and the significance of these arguments.

In response the LILO "boot:" prompt arguments are the name of the kernel image, kernel configuration parameters, and command line arguments for the first-run process (init). Arguments are separated by spaces. For example: "boot: vmlinuz rw append=128M root=/dev/hdb2 single" will boot a compressed kernel image named "vmlinuz" with the file system volume set readable and writable, force the kernel to recognize the full 128 MB of RAM, boot the kernel image from the second partition of the second IDE hard disk, and tell init to keep the kernel at run-level 1 (single-user mode).

Once the single-user console (no login) prompt is displayed, the super-user is free to change the system as necessary, even if it means editing the password file and removing the forgotten super-user password.

26. Describe the naming conventions for Linux disks and partitions. Explain why the CD-ROM drive is missing from the "list."
27. Describe the role of fdisk and explain the type of interactions that occur with other operating system fdisks.

Fdisk displays and/or modifies the partition table for the specified hard disk first sector (aka, the Master Boot Record or MBR). Each partition has a name, boot flag, linear block address (LBA), beginning, start, and end. For MS-DOS, only one partition may be "active" or boot enabled, but LILO ignores this flag and will try to boot any specified partition. Each partition also has a size (in 512 byte sectors) and ID. The ID is in numeric and English forms and indicates which OS uses the partition and in which way the partition is employed by the OS.

Some proprietary OS versions of fdisk write undocumented (secret) data to MBR in addition to the partition table and they expect to see the same information at boot time. For example, if part of the hard disk is setup with IBM's OS/2 and Linux rewrites the partition table, then OS/2 will not boot. To overcome these types of problems, partition the hard disk and install the proprietary OS first, and **only** rename the partition types with the Linux fdisk program.

28. When the OS will not boot, explain how to trouble shoot MBR problems.

Boot from floppy with Linux, DOS, or Windows "rescue" disks. Examine and modify partitions as required. Refer to a saved or printed copy of the old partition table values.

29. Be able to describe a partition table such as the one below.

```
30.  #fdisk /dev/hda
31.  Command (m for help): p
32.  Disk /dev/hda: 255 heads, 63 sectors, 784 cylinders
33.  Units = cylinders of 16065 * 512 bytes
34.    Device Boot  Begin   Start    End   Blocks   Id  System
35.  /dev/hda1  *       1       1     26  208813+  83  Linux native
36.  /dev/hda2         27      27    784  6088635   5  Extended
```

```
37.   /dev/hda5        27      27     645  4972086   83  Linux native
38.   /dev/hda6       646     646     776  1052226   83  Linux native
39.   /dev/hda7       777     777     784   64228+  82  Linux swap
40.   Command (m for help): q
```

41. Give three "levels-of-format" and explain the rationale behind "fdformat," "mkfs," "mke2fs,"mkswap," etc.

42. Explain why one cannot just "turn off" or "reset" Linux and give four ways to stop Linux.

. Type Ctl-Alt-Del
a. shutdown now
b. halt
c. reboot

43. One way to upgrade an existing Linux system without disrupting its Web presence is to install a new hard disk and a new version of Linux on another box, copy over the configuration and user files from the original system, and then physically move the hard disk back to the original system. In this way you can test and configure the new system "off-line" and have the system down just as long as it takes to swap the hard disks. But then you discover that the new version of Linux boots only from the other box and not on the original system. What is wrong?

The disk geometry entered into the other box's BIOS is not the same as the geometry for the original system's BIOS. Thus, when LILO runs on the original system, the BIOS cannot translate cylinders, sectors, or heads into the correct offset to get the kernel image.

44. When installing Linux there is usually a colorful DOS-based GUI-looking program that takes you through the installation by asking questions and configuring the system. Sometimes, however, the GUI gets "stuck" and you have no choice but to give up and begin again. How can you shorten this debugging time?

This is where virtual terminals come in. The GUI is on terminal F7. Terminal F2 shows all the kernel diagnostic messages as it boots and terminal F3 shows the batch commands being executed. Between these two console screens you can determine where the install process is hanging. For example, if the kernel cannot read the floppy, you will see time-out errors on the floppy drive. The same applies to the hard disk or CD-ROM drive as well as other peripherals.

45. Give some example error messages and subsequent fixes for the Linux boot process.

. *Out of memory* - The install kernel needs at least 2 MB; some installations may need as much as 4 MB of RAM. Smaller amounts of RAM can be used if the kernel is booted without the use of RAM disks (rinitrd).
I. *Permission denied* or *File not found* - The boot media was written on a "flaky" floppy (probably using one of those free AOL floppies). Copy the boot image to a new floppy and try again.
II. *VFS: Unable to mount root* - The root file system could not be found. Or the root file system could be on a bad RAM disk image, corrupted floppy, or unsupported CD-ROM drive.

46. Give a PMS description of the IBM PC system architecture and describe those parameters that may be in conflict during installation of a new system with new hardware.

47. Define and describe the operation of DMA. Describe its implementation on the IBM PC system architecture.

48. Explain what is meant by "autoprobing," describe where this information can be found, and explain how to use this information for hardware problem resolution.

49. Define the follow install error messages and describe how to overcome the problem.

o *Read error* or *File not found* - Poor quality media (are you sure that you have not recycled those free AOL floppies) or wrong media format (are you putting a Macintosh floppy into a PC drive). Re-format media and re-copy install images.

- *Tar: read error* or *gzip: not in gzip format* - Media OK, but the file contents are corrupted. Download a new copy (in "binary" mode) from a different site.
- *Device full* - Start over! Make new larger partitions that will hold the software you want.
- *Read_intr: 0x10* - If this occurs with general hard disk access, then it has bad blocks. Do a low-level reformat of the hard disk. If it occurs with mke2fs or mkswap, then the real partition size is smaller that the stated (program argument) size.
- *File not found* or *Permission denied* - This is a rare error message. There are missing files or the installation software has bugs that set the wrong permissions. Try a different Linux distribution.

50. Contrast the following two FTP commands: "mget *.*" *versus* "mget *" and explain how they may impact the down loading of a Linux distribution.

51. Explain why one should not do routine tasks as superuser.

First, like other Unix systems, Linux employs a fully indexed file system with direct access to and from the free block list. When a file is deleted, its contents form the first-come-first-served free blocks of the next created file. Thus, once a file is deleted in Unix, it is a final delete.

Second, running as superuser and typing in an extra space can have dire consequences, such typing `rm * .bak` or typing `rm -rf / home/fred/tmp`. In both examples the inadvertent space results in deleting all files in current directory or in the whole file system.

Third, working as superuser means the creation of files as superuser and that means a higher probability having to run as superuser in the future to access the files created earlier.

52. Once the Linux distribution is up and running there are other possible errors. List some of the errors and explain how to fix the problem.

. *Drive not bootable-Please insert system disk* - The MBR has been zapped. Use a rescue disk for the OS you are trying to boot. For example, in MS-DOS use the command FDISK /MBR.

A. *The wrong OS boots* - LILO or other primary boot system is mis-configured. Reconfigure and re-install LILO.

B. *Login incorrect* - You forgot all the passwords? Boot Linux in single user mode and run the password program as superuser.

C. *No shell* or *Shell-init: permission denied* - Somehow you erased the password file or the root permissions are too restrictive. Boot Linux in single user mode and copy the password file that you have hidden for a rainy day or `chmod` the root directory permissions.