Name

1. Name least four operating systems and say a little about them.

* **Open BSD: A BSD kernel based operating system that is reputed for its security and thoroughly audited and entirely open code.**
* **Gentoo: A meta distribution (usually) using the Linux kernel that allows the user to completely customize and optimize their system based on its intended use and what packages they want it to have. Gentoo systems do not update in the normal sense instead they are on a rolling update system (portage) where a snapshot of the current package source codes are compiled on the system to update it.**
* **Android: A mobile operating system spearheaded by google that runs on mobile devices around the world.**
* **Mint: A light weight linux distribution based on stripped down Ubuntu, available with a variety of Window Managers (but not unity)**

2. What is an operating system?

**An operating system is the primary layer of software on a computer that runs between the hardware and everything else on the computer. Operating systems manage essential tasks like resource management, File I/O, scheduling, providing drivers to interface with hardware and security. All user applications run on top of the operating system and request resources offered by the operating systems through a series of system calls (requests for the operating system to do something). The operating system is also the only hardware that is allowed to run in privileged mode (ring 0) on the cpu(s).**

3. What are some major functional areas of operating systems? **Major functional areas of operating systems include the kernel loader (system with basic knowledge of file-system and in the case of Linux gzip decompresser), Kernel, File I/O systems, Memory management system, CPU scheduler, and driver unloading and loading facilities.**

4. What is the purpose of profiling programs? How is this related to optimizing programs? **Profiling programs are used to gather information on how much system resources (such as cpu time) a specific process is using and to gather statistical data on those processes use resources. This data can be used to analyze what process or parts of a program use the most resources and can be used to find bottlenecks and then optimize them. Or to better share resources across a system.**

5. What are some advantages to micro-kernel designs of an operating system? Micro-kernels **have the bare minimum an operating system needs to function. All other functionality is added on in self contained modules. As a result micro kernel based operating systems are extremely small which lends them to running on embedded devices. Also as all modules are separate the failure of any one module will not take down an entire system where as if that functionality was compiled into the kernel its failure would take down the entire system.**

6. What are some differences between static and dynamic libraries? **Dynamically libraries are linked at to at run time and only exist in one place in system ram. As a result several processes can share the same dynamic library. This allows for easier updates of important pieces of software. If a security hole is found in the functionality of a dynamic library it can be patched, and only the dynamically library has to be recompiled and then replaced. Then all software that uses this library will be fixed. Static libraries are linked in at compile time and can only be run by one process at a time. Multiple process that use the same static library will each have their own instance of that same static library in ram. However static libraries allow for more portable deployment because you can include a static library with your program and guarantee that the computers you deploy to will have this library, where as if it is dynamic it may or may not be installed.**

7. Give an example of a system API call. **A program that needs to read from and write to a file will (at some point) use a system API call to request block information from that desired hardware device and likewise use a system API call to request writing of a data block to the hardware.** **The function template for the write() system call on \*nix operating systems looks like: ssize\_t write(int** *fd***, const void \****buf***, size\_t** *count***);**

8. Give some advantages and disadvantages to virtualization. **Virtualization allows the running of multiple operating systems on one host operating system. This can allow you to for instance run a game in windows, while having a virtual Linux environment for software development. Virtual operating systems are isolated on their own virtual machines so the total failure or hijacking of a virtual operating system, will not compromise the hardware it is running on or other virtual machines running on the same hardware. However virtual machines rarely run as efficiency as software on a native machine, virtual machines are complicated and difficult to write and debug. Software running an virtual machines has limited resources and the overhead of running a virtual machine takes resources away the software you to be running.**

9. Process communication is done via memory maps and message passing. Give some example scenarios where one would be a better choice than the other, and explain why. **A multi threaded program where multiple process are reading from and updating the same data (such as an instant messaging program) would be better suited for the message passing model of inter process communication as this prevents race conditions and in expected results. An digital audio workstation program that does (nearly) real time audio processing between multiple tracks running on multiple threads, or processes would be better suited for the memory mapping model of interprocess communication as the sound being produced in a reasonable time is more important then the quality of the signal that is produced.**

10. Name the main features an operating systems’ user interface should have for a desktop environment. What changes for a server environment? **A desktop environment should have a graphical user interface (GUI) that adds a layer of abstraction that allows the user to efficiently and intuitively navigate the system and execute commands/run programs. Things like click-able icons and movable tiles representing programs (assuming they are intuitive to use) would make for a good desktop user interface.**

**Servers run on a different paradigm and will often be unmanned for if not all of their operational life cycles. Graphical user interfaces (especially modern ones with 3d acceleration) take up a lot of system overhead, that could be put to better use operating the server. As a result servers are better suited to having a command line interface that takes minimal resources to run and gives the user (with a higher level of technical aptitude) more control over the operation of the system.**