* How do you run a malware in the form of DLL?
* Ans: Run a malware in the form of a DLL by going to the command line and running it using this command: rundll32.exe DLLname, Export argurments.
* Which tool would you use to monitor Registry?
* Ans: Regshot. It is an opensource registry comparison tool which compares 2 registry snapshots and reports on the differences.
* What is the difference in purpose between Procmon & Process Explorer?
* Ans: Procmon monitors registry, file, process, and network activity. It monitors all system calls for all processes in real time by default. There is a need to filter events. Process explorer on the other hand, provides insight to running processes. It can list active processes, show DLLs loaded for each process, see active TCP connections and create/kill/validate processes. It is also able to verfiy whether a process has been digitally signed by Microsoft as many malwares have names which are the same as a legitamate Windows process. This is only applicable to processes on the disk and not in RAM.
* Which tool would you use to capture network packets?
* Ans: Wireshark. It provides visualization of network packets, packet-stream analysis, and in-depth packet analysis.
* What is mutex? Why is it important in the context of malware analysis?
* Ans: Mutex means mutually exclusive flag. It acts as a gate keeper to a section of code allowing only one thread in and blocking access to all other threads. This ensures that the code being controlled is only being hit by a single thread at a time.
* They are important because malware authors use it to prevent malware from reinfecting a system which is already infected by the malware.
* What is the purpose of Windows Registry?
* Ans: The registry helps Windows manage and operate your computer, ensuring access to critical resources and helping important programs configure settings. A hierarchical database structure of keys and values make up the registry.
* What is the difference between a process and a thread?
* Ans: **Process**  
  Each process provides the resources needed to execute a program. A process has a virtual address space, executable code, open handles to system objects, a security context, a unique process identifier, environment variables, a priority class, minimum and maximum working set sizes, and at least one thread of execution. Each process is started with a single thread, often called the primary thread, but can create additional threads from any of its threads.
* **Thread**  
  A thread is an entity within a process that can be scheduled for execution. All threads of a process share its virtual address space and system resources. In addition, each thread maintains exception handlers, a scheduling priority, thread local storage, a unique thread identifier, and a set of structures the system will use to save the thread context until it is scheduled. The thread context includes the thread's set of machine registers, the kernel stack, a thread environment block, and a user stack in the address space of the thread's process. Threads can also have their own security context, which can be used for impersonating clients.
* Microsoft Windows supports preemptive multitasking, which creates the effect of simultaneous execution of multiple threads from multiple processes. On a multiprocessor computer, the system can simultaneously execute as many threads as there are processors on the computer.
* Ans: Both processes and threads are independent sequences of execution. The typical difference is that threads (of the same process) run in a shared memory space, while processes run in separate memory spaces.
* I'm not sure what "hardware" vs "software" threads you might be referring to. Threads are an operating environment feature, rather than a CPU feature (though the CPU typically has operations that make threads efficient).
* Erlang uses the term "process" because it does not expose a shared-memory multiprogramming model. Calling them "threads" would imply that they have shared memory.
* What is the difference between an Application, Process, and Services?
* Ans: Both applications and services have processes associated with them.
* An **application** is a **program which you interact with on the desktop**. This is what you spend almost all your time using on the computer. Internet explorer, microsoft word, iTunes, skype - they are all applications.
* A **process** is an **instance of a particular executable** (.exe program file) running. A given application may have several processes running simultaneously. For example, some modern browsers such as google chrome run several processes at once, with each tab being a separate instance/process of the same executable. In some cases, complicated applications may have multiple processes; for example, Visual Studio runs a separate process when it compiles code from when it displays the IDE. However, most often, a given application is running from a single process; for example, no matter how many microsoft word windows you have open, only a single instance of winword.exe is running.
* A **service** is a **process which runs in the background** and does not interact with the desktop. In Windows, services almost always run as an instance of the svchost.exe process, the windows service host process; however, there are sometimes exceptions to this.
* Sometimes, processes may run in the background without interacting with the desktop, but without being installed as a service. Many device drivers with enhanced features do this. For example, a touchpad driver will usually have a process which runs when a user logs in and handles the special features of the touchpad but isn't a service and doesn't show any windows to the user.
* Sometimes an application may depend on a certain service. Printing from any program requires that the print spooler service be active. Installation packages (.msi installers) require that the windows installer service be running. Antivirus programs usually employ a service so they can continue running even when the user is not logged in.
* Processes usually exit when an application is closed, however this is not always the case. Some programs, particularly download and backup programs, may continue to run in the background without displaying any windows. Antivirus is also an example of this - in addition to using a service, many antivirus applications run a process silently in the background which only displays an application to the user when action is required.