1.) There are two special signals (KILL and STOP) which are not handled by the process they are sent to. When a KILL or STOP signal is generated, the operating system itself handles this signal and kills or stops the appropriate process. Considering what you learned in today's lab, speculate as to why the system designers chose to include signals which are handled solely by the operating system.

The KILL and STOP signals are handled solely by the operating system to protect the user. If the KILL and STOP signals could be modified, then the user could write a program that couldn't be terminated.

2.)What benefit do we gain from using the pause system call as opposed to an infinite while loop?

An infinite while loop would eat up a lot of processing power. The pause command is a lot less costly of an operation. The OS can still use the core while pause() is going on. An infinite while loop would demand continuous processing.

3.)Why do we mask other signals while inside the signal handler?

We mask other signals while inside the signal handler so that out handler cannot be interrupted by other signals before it is complete.

4.)When we implement the time out, we do not mask the SIGALRM signal. Why?

We don't mask SIGALRM because we want this signal to interrupt the ctrl-c handler so it times out.