**Assignment 1: Threads**

In this assignment we used the pthread library to demonstrate a simple use of threads. In particular, we had to create a child thread from a parent thread and kill the child thread when the ‘Enter’ key is read in the parent thread. The source code for accomplishing this is written in C and ran on an Ubuntu. It can be accessed [here](https://github.com/Vardominator/CSULBProjects/tree/master/CECS326_OperatingSystems/Assignment1_Threads).

**Question 1:**

The operating system represents each process using a **process control block (PCB).** The PCB includes all of the information about the process, namely its **I/O status information.** This portion includes the list of devices that are allocated to the corresponding process. When a process is multithreaded, its PCB is expanded to allow multiple threads to access the same information and resources specified within the PCB. In our case, a separate thread is continuously printing messages while the main thread is waiting for a keyboard input. This is possible because of the reasons mentioned above: multiple threads within processes share resources specified within the PCB of the process. A new thread does not steal access to a particular resource of I/O device when it is created, and therefore one thread could hang out while another waits for input.

**Question 2:**

Based on the previous answer this is definitely possible. Since the same I/O resources are being shared by all of the threads within a process and since the child thread is branched off from the parent thread, it is possible halt a sleeping child thread.

**Executed code:**