**Assignment 1: Context Switching**

I read the assignment sheet on GitHub. I refreshed my C language skills and read the assembly language X86 registers table and use of the registers. I referred the given swap program, it was helpful to write the context switch assembly language program. I implemented the context switching of threads and learnt how to switch the old thread to new thread. I understood how the memory is allocated in the thread.

**What would be an elegant way to create and manage an arbitrary number of threads?** Threads concurrency would be an elegant way to create an arbitrary number of threads. The threads can access the shared memory space. Context switching is used to manage multiple threads. Context switching is the process of storing and restoring the state of a thread.

**Can you think of any uses for even the very simple form of threading we've developed in this assignment? What extra features could we add to make it more useful?** This simple threading program is used to compute the mathematical logic and simple input-output. It is more useful to add extra features such as mutual exclusion (mutex) and synchronization.  
  
**Memory management can be a challenge in threading systems. When is it safe to free a thread's stack? What about its thread control block? How would this change if we wanted to return results from our threads, or implement a join procedure? (see the Birrell article for a description of join)**

When a thread completes its tasks, then the thread terminates and control block of thread is also deleted. A join procedure returns the result of the thread’s initial procedure (created by fork). If we wanted to return results from our threads or implement a join procedure, the parent thread should not terminate but it should be waiting till it gets the value or till its child terminates.