Assignment 2
EEL4930/5934 Advanced System Programming

In this assignment, you are going to implement the Location Updater program from Assignment 1 using multiple threads instead of multiple processes. To achieve this, your program will use the Pthreads library to create one thread for the Email Filter and another thread for the Calendar Filter. Note that this is an instance of the Producer Consumer problem, where Email Filter is the producer and Calendar Filter is the consumer. You will implement a dynamic data structure to represent the buffer that enables communication between the producer and the consumer. The buffer will have a certain number of entries and each entry in the buffer can hold exactly one calendar event. The size of the buffer will be determined by the command line parameter. You should assume that the Email Filter will be reading the input from the standard input. So, as an example, your multithreaded Location Updater program will be executed as follows:

```
$ ./locationUpdater 10  < inputFile
```

where 10 denotes the number of entries in the buffer and inputFile contains the emails.

Please make sure that the Email Filter thread waits if there are no available slots in the buffer and the Calendar Filter waits if there are no items in the buffer. Also, the Email Filter should let the Calendar Filter know when no more data will be arriving. To get full credit, your solution should be free of race conditions and deadlocks and produce the correct output by properly synchronizing the threads. You can use mutexes and condition variables and/or semaphores for synchronizing your threads. Please submit your files (source, README, and a Makefile) on CANVAS by the due date.