Pedro Schmidt

CSCI 451

HW6 PAPER

Basically, in assignment 5 we had to write a program that uses the pthread library to create three threads. We were given specific instructions such as including a hw5.out which was for writing.

Each thread would open up a file. we had to create three threads therefore we had three files not including the hw5.out file. These files consisted of hw5-1.in and so on until we reached to hw5-3.in. All of these files were meant for reading. Initially we were supposed to create a global variable in the main that would be used to pass characters through threads. When reading one-character value from the file we put that value in a global variable. They all work in a sequential order such as thread #1 will wait for thread #2 and #3 before reading the next value and so on for each other threads. While modifying homework five to accommodate to homework six we had some changes to make and add- on to the program. Given two cases or scenarios that our program would have to follow. One of them was that a case unexpectedly exists by doing certain functions and the other case where it unexpectedly dies. Both of these we were not allowed to unlock any mutexs. In my code for case one it acknowledged that there was an issue or a page fault for my instance in a certain thread. In the output the fault was in thread three. Once this fault was acknowledged it was then replaced and then the code continued on. Eventually receiving another page fault, but this time in thread two and also being replaced. Finally, it continues through all threads until it is finished. Basically, my code kept running, while having these cases. Overall, the design of the of program sort of set it up to where if there was an issue it wouldn’t just end the program entirely, but however just replace it with another value and continue on. The thought process behind this was once we knew the process of each thread , how could we disrupt that in a way that it would not make the program end.