Write a COGENT explanation, along the lines of the "Gowtham and Noelle" story in class, of how the threads take turns in this application. Explain why it may be useful to have more threads than CPU cores for this program but not for the prime finder above.

In the program, a we needed to be create a web probe that would read through a list of URLs in a file using a number of threads that the user entered. The file, urls.txt contains a list of URLs to use as examples. During the execution of the program, the user needed to enter ./webprobe urls.txt followed by two numbers to let the program know how many threads to produce in order to run the program. The first number was used to find the number of time the thread needed to access the web and the second number was needed to create the probe threads and the reporting thread.

At the begnning of the program, the load_recent function was made in order to read in wwidth's mose recent times. A "for" loop was made in order to continusouly read in the most recent time as the program ran. the next for loops was used if i was equal to wwidth. if this scenario occured, then whatever was written in wwidth before would be overwritten by the new value. This is how the program was able to read the most recent access time.

In the main function, all the variables and arrays are set up initially. Then the probe threads, which each work to probe a URL, are created. Then, the reporter thread is created. In the probe() function, a section of the code is locked, so that other threads cannot execute it. Then, a line is read, and it is unlocked. The locking occurs because many threads may be modifying the same thing, so it is a safeguard. Locking and unlocking was one of the most crucial components of the program. The time is checked per access, and the time is added to the global array with a max capacity of wwidth. The access count is then incremented. After the URLs are probed by the probe threads, the reporter threadi s used to present the mean and standard deviation of the last wwidth probe results.

Because the instructions asked to "repeatedly cycle through the list," the only way webprobe.cpp would be terminated was through the ctrl+c option, since the program needed to run continuously. Similar to the example on page 186 in the text, the threads in this program gave the illusion of a multi-tasking CPU, while in reality, everything was merely pushed onto the same stack.