Jessie George
CS4440-02 Operating Systems Project

Subject: Luhn Algorithm for Credit Card number validation.

I made a thread for each of the credit cards in the argument list. There is no data dependency between different cards when validating a card so these threads can run concurrently. All these threads are stored in a list.
Join threads for all cards before printing the verification results for all cards. Maintain the order of input when printing output by assigning an input number to each card, and storing the verification result in an array of Strings for output. We need to join all threads to ensure that the output array will no longer be modified.

One thread checks digits in even places of a card. One thread checks digits in odd places of a card. There is no data dependency between even and odd places for a card so these threads can run concurrently.
The total depends on the sums achieved from processing even and odd places. This data dependency yields a need for synchronization. Thus, join the even and odd threads before calculating the total.

One thread calculates the total and checks if it is divisible by 10. Join this thread before printing verification output for this card, because the verification output depends on the valid boolean which may be updated to true by this thread if the total is divisible by 10.

I named the class CredVerify since it verifies credit card inputs.
I chose to store the credit card companies as a list of CompanyInfo objects because it allows for reuse of code comparisons. I used private static final for the inner class, method, and data member related to the companies. Private for security. Static because a CredVerify object (i.e an input card) is not required to access the companies, and only one copy of this information is needed across all CredVerify objects. Final because it should not be modified elsewhere.

Concise verification results are printed to console for all cards. Any Exceptions are printed to console.
Detailed processing is stored in log files. The directory name is cred<current timestamp> and within the directory there are files called input<# in arguments list>.txt for each card. These log files are helpful to see why a card is invalid, or see calculation steps.
I chose logging to separate files instead of using a semaphore for synchronized logging to console because of performance speed and because the details of different cards do not affect each other.

When testing, I tried to show some edge cases and bad input along with the reasons why those cards are invalid. For valid inputs, I referred to the table given on page 2 of the project question, and tried to show examples for each of the different prefixes and card lengths.