#1 - In This Subnet?

Implement a program that determines if a given IPv4 address is in a given subnet. The IP address is passed as a string representation of a 32-bit unsigned int (e.g., 0x62D2ED4B). The subnet is passed as a string representation of a CIDR subnet (e.g., "98.210.237.192/26"). The program outputs True if the IPv4 address is in the subnet, and False otherwise.

Bonus points for a program can read address/subnet pairs from an input file and write the results, in a useful fashion, to an output file, both optionally specified on the command line.

#2 - Change of Phone Number

Implement a script to update the phone number embedded in many of the store's HTML pages.

Over the years, the content authors have embedded the number on pages using a wide variety of punctuation (i.e., 800 438-4357 or 800.438.4357) while others used letter mnemonics (e.g., 800-GET-HELP). Some inserted the US country code, with or without surrounding parentheses. The store has >50K HTML files, all under an NFS /var/www mount point shared by all servers.

Your job is to replace all the old number with one version of the new number: 202-456-1414. The job needs to be done tonight, while the web servers are running. You do NOT have to convert the new number to the existing format used on the page.

Bonus points for a solution that is implemented as a single pipeline of Linux commands.

#3 – Implement an LRU Cache

Implement a web app that uses a fixed-size LRU cache to efficiently serve up imagery

The app exposes a web API that accepts two floats, the first in the range -90 to 90 (Lat) and the second in the range of -180 to 180 (Long), which are Mars coordinates. The app returns the URL of an image file of those coordinates on the Martian surface, or one of the standard HTTP error codes.

This app implements an LRU cache of fixed size (3000), keyed on Lat/Long pairs, which operates in O(1) to return the requested URL. New URLs are obtained using a library function GetImageURL(float, float), which returnas instantaneously but at a high $ cost; implement a stub version of GetImageURL() that returns a random number as a string for purposes of this problem.

On a cache hit, the app returns the cached URL. On a cache miss, the app obtains a new URL and caches it, ejecting the oldest cached item if the cache is full. These operations must occur in O(1).

Bonus points for adding diagnostic API calls to get and clear cache hit and miss counters, and to track the execution time of each of the three main LRU cache behaviors (hit, miss when not full, miss when full).