CS 225 Extra Credit Project Proposal- Anvita Ganugapati and Aishwarya Pasham
Determining Password Strength using the KMP Algorithm

Academic Reference: https://www.cs.cmu.edu/~15451-f17/lectures/lec09-kmp.pdf

Algorithm Summary:
The KMP algorithm aims to search for a substring within a larger string. It does so by taking in a string, and pattern within it to search for, and utilizes a table to depict how many characters to skip when a pattern mismatch occurs. This is to avoid re-checking portions of the larger string. This table is updated with the length of the longest common prefix and suffix from every character in the string. In our implementation of this algorithm, we will be taking in a password, and employing KMP's pattern matching to determine the strength of the password, based on the indices in which it is spotted.

Function I/O:
The expected inputs for the KMP algorithm usually include two strings, a string in which you search for a pattern and a shorter string that is the pattern you are looking for. We plan to implement this algorithm to predict the strength of a password based on the number of repeating sequences it has.
The function findRepeats takes in a dataset containing common passwords and a string which is the pattern in passwords you are looking for. This is where the KMP algorithm is implemented. The csv or txt files will be parsed to create a vector of just the password strings from the password column. For every password in the dataset, a vector will be created with the indices the pattern occurs on (if any). The output will be a vector the size of the dataset made up of integer vectors. The second function predictStrength will take in the dataset and two strings which are the password and pattern. It will iterate through the dataset converted into a vector of strings with just the passwords to find the index the password occurs on. Then, it will call findRepeats and store the vector of pattern occurrences for the index we found earlier. The size of this vector indicates how many times the pattern occurred in that password. Based on the number of repeat occurrences, the function will output a number 1 to 3 which indicates password strength with 1 being a weak password and 3 being relatively strong.
The tests for findRepeats checks if the KMP algorithm was correctly implemented by checking what indices it found the pattern match for a given password from the dataset. The predictStrength tests verify whether the correct password strength is predicted and compared to the strength it is expected to be.

Data Description:
Our data was found on Kaggle as a csv or txt. The datasets contain common passwords, weak passwords, and leaked passwords. When we process these datasets, we are mainly parsing the data and creating string vectors that store the password column. The data is stored in /data.