Overview of how the program works:

The program starrs by reading a list of words from a file words.txt and loading them into an array. These words will be used for guessing passwords. The dictionary is sorted in descending order of word length to make it more effective for the guessing process. Longer words are first, reducing iterations for shorter matches when unique words are similar. (i.e. bird and birds)

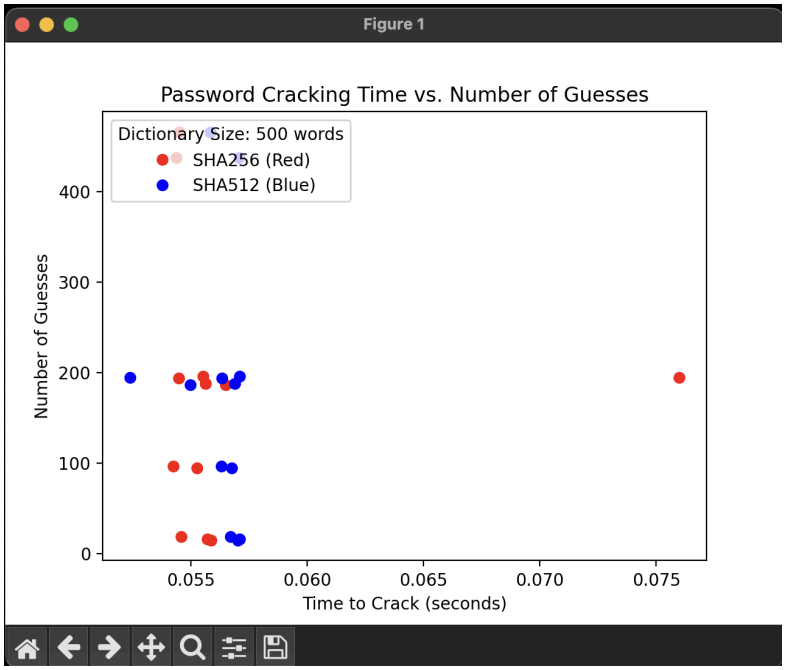
The user then enters a password, which is checked if it is in the dictionary. If the user enters 'q', the program terminates and visualizes the results through matplotlib.

Using an algorithm, the program matches the longest possible dictionary word to the beginning of the input password. This process continues until the entire password is reconstructed or if something does not much.

The password is hashed using two cryptographic algorithms: SHA-256 and SHA-512. The program uses the pbkdf2_hmac function, which enhances security with a salt and multiple iterations.

Demonstration:

To run the program you have to start off by writing “python3 solution.py” in the command prompt to start the program. Once the program is running it will prompt you with entering a password, once you enter the password it will check the dictionary of passwords in the word.txt file and see if it matches with any of the passwords. When a match is found it will start Hashing the passwords to check how fast it takes to hash them. After cracking the passwords and hashing using two different hashes, the program requests more passwords and repeats the above process again for each password. When the user is done cracking passwords, typing ‘q’ ends the program and displays the graph for all the passwords the user has entered.



Example run of the program,

>python3 solution.py

>Enter a password (or 'q' to quit): password

>SHA256 Hash: af04036ed6bd35f97f759765655a35d52a49888a8da4d68d9c1011b13699547c
SHA512Hash:13e2b5964e608223d2cc0e2c948c3998932325bd5ac92b46627963c20001e6c9486
a7799483b2ab5f163a6fd95d510e908fd63f26b65d078c83e643f5be17187

Cracked SHA256: password
Time to hash SHA256: 0.05486416816711426 seconds
Cracked SHA512: password
Time to hash SHA512: 0.05598902702331543 seconds

> Enter a password (or 'q' to quit):