```python
#https://github.com/KrishAleti/PwdLeakCheck
#Gives the count how many times our entered password was hacked if it is found in the list of hacked paswword if not found, carryon message will be disaplyed
#To run from terminal: python PwdLeakCheck.py password1 password2 password3
#https://docs.python.org/3/library/hashlib.html
import requests
import hashlib
import sys

def request_api_data(query_char):
    url = 'https://api.pwnedpasswords.com/range/' + query_char
    res = requests.get(url)
    if res.status_code != 200:
        raise RuntimeError(f'Error fetching:{res.status_code}, check the API and try again')
    return res # <Response [200]>
def get_pass_leaks_count(response_hashes, our_hash_to_check):
    hashes = (line.split(':') for line in response_hashes.text.splitlines())
    for h,count in hashes:
        # print(h,count)
        if h==our_hash_to_check:
            return count
    return 0
def pwned_api_check(password):
    sha1password=hashlib.sha1(password.encode('utf-8')).hexdigest().upper()
    first5_char, tail_char=sha1password[:5], sha1password[5:]
    response=request_api_data(first5_char) #we get all the hashes that match the begining of our hashed password
    # print(first5_char,tail_char)
    # print(response)
    return get_pass_leaks_count(response,tail_char)
def main(args):
    for password in args:
        count= pwned_api_check(password)
        if count:
            print(f'{password} was found {count} times...you should probably change your password')
        else:
            print(f'{password} was not found... Carryon!!!')
    return "done!"
if __name__ =='__main__':
    # main(sys.argv[1:])
    sys.exit(main(sys.argv[1:])) #to exit the process and come back to the cmd line just if anycase the process doesn't exit.
    #we get "done!" here because we are exiting out of this file, so this entire py file is run and at the end as 'main' returns done!,
    # it is also printed


''' You could use *args like main(*args) and pass in each argument separately but here
 The expectation for this function is that a list 'main(sys.argv[1:])' will be passedin instead, so it will be assigned to just a single argument main(args)

```

```
47  Don't get caught up in the name args or *args. Those names are arbitrary. For *args the
    important part is the * which tells Python you want to collect all the arguments into one
    container. You could call it anything, like *passwords or *my_stuff.'''
48
49  """here we are giving the password from the terminal and the terminal commands will be
    stored somewhere (like in terminal if we presss up arrow we will get the
50  previous command), so this is not super secured. So, the good way is to read passwords
    from the text file instead of cmd and shred the txt file after work is done """
```