CSC 321 Lab 4
Hank Tsai

Shadows and Hashing

What I did was parsing the users' salt and hashes.
And use the salt with words in nltk.corpus to find a match with the user's hash. And print out the output. The thing I found interesting was how long it took to crack those passwords. I was cracking passwords with salt, meaning I grouped people with the same salt. And for every salt, start a new timer, the timer's unit is in seconds.

My Source code:

```
from bcrypt import hashpw
from nltk.corpus import words
import time

class User:
    def __init__(self, name, salt, hash):
        self.name = name
        self.salt = salt
        self.hash = hash

def process_user():
    users = list()
    with open('shadow.txt', 'r') as f:
        for line in f:
            name = line.split(':')[0]
            salt = line.split(':')[-1][:29].encode('utf-8')
            hash = line.split(':')[-1].strip().encode('utf-8')
            # print(name, salt, hash)
            users.append(User(name, salt, hash))
    return users

def crackPassword(users):
    word_combination = [w.encode("utf-8") for w in words.words() if len(w) >=6 and len(w) <= 10]
    user_map = dict()
    for user in users:
        if user.salt in user_map.keys():
            user_map[user.salt].append(user)
        else:
            user_map[user.salt] = [user]

    for salt in user_map.keys():
        st = time.time()
```

```
        for word in word_combination:
            hash_pass = hashpw(word, salt)
            for user in user_map[salt]:
                if hash_pass == user.hash:
                    et = time.time()
                    print(f'{user.name} password: {str(word)}  time: {et-st}')

if __name__ == '__main__':
    users = process_user()
    crackPassword(users)
```

My output: