TP N°2 Algoritmos y Programación: Generador de tweets

Materia:(7540) Algoritmos y Programación I

Cátedra:ESSAYA, DIEGO NICOLAS - MARTINEZ,GASTON ALBERTO

Práctica: Barbara

Nombre y Apellido: Joel Isaac Fernandez Fox

Padrón:104424

Ayudante a Cargo: Joel Nicolas Saidman

Inconvenientes a la hora de programar:
*Buscar una manera rápida de leer el archivo tweets.csv.

*Lo mismo que el anterior solo que con el archivo favoritos.csv.
me costó encontrar una forma de ir agregando texto y que a la hora de consultar ese texto no se haga tan pesado con el comando favoritos.

Soluciones planteadas:
*Seguí su consejo y utilice un diccionario pero nose si la aplique bien.

*Lo que quería para que no se me haga tan pesado era escribir al principio del texto así a la hora de leerlo nada más leía las líneas de la cant. de favoritos que se pedía ya que pensaba agregar todo un tweet en una sola línea si se decidía agregar un tweet a favoritos. Como no se me ocurrió nada, no me quedo de otra que leer todo el archivo para llegar al tweet más reciente.

GeneradorTweets.py

```
import sys
from csv import reader
import random
def comportamiento_en_consola():
	'''Comportamiento de lo que ingresa el usuario en la consola.'''
	if len(sys.argv)==1:
		print('No ingresó ningún parámetro.')
		return
	if sys.argv[1].lower()=='generar':
		if len(sys.argv)>4:
			print('Ingresó parámetros de más.')
			return
		if len(sys.argv)==2:
			generar_main()
		if 4>=len(sys.argv)>=3:
			if len(sys.argv)==3:
				generar_main(sys.argv[2])
			else:
				generar_main(sys.argv[2],sys.argv[3])
	if sys.argv[1].lower()=='trending':
		if len(sys.argv)>3:
			print('Ingresó parámetros de más.')
			return
		if len(sys.argv)<3:
			print('No ingresó los suficientes parámetros.')
			return
		if not(sys.argv[2].isdigit()):
			print('Por favor ingrese un número entero.')
			return
		trending_main(sys.argv[2])
	if sys.argv[1].lower()=='favoritos':
```

```
		if len(sys.argv)>3:
			print('Ingresó parámetros de más.')
			return
		if len(sys.argv)==3 and not(sys.argv[2].isdigit()):
			print('Por favor ingrese un número entero.')
			return
		if len(sys.argv)==3:
			favoritos_main(sys.argv[2])
		else:
			favoritos_main()

def generar_main(usuario1='Todos',usuario2=''):
	'''Imprime un tweet generado pseudo-aleatoriamente utilizando
	como fuente únicamente los tweets de los usuarios seleccionados.
	Si se ingresa únicamente el usuario usuario1,
	el tweet generado debe generarse como si lo escribiera este usuario.
	De no ingresarse ningun usuario, se deberá generar un tweet usando como fuente
	todos los tweets disponibles.
	De ingresarse un usuario del cual no se tiene información el programa mostrará
	un mensaje de error por consola especificando dicho error.
	Luego de mostrar el tweet, se debe dar la opción de agregar este tweet a favoritos,
	guardándolo en un archivo en caso de que el usuario responda afirmativamente.'''
	print('Generando tweet pseudo-aleatorio. Por favor espere...')
	dict_tweets=dar_dict_tweets()
	list_tweets_users=dar_list_tweets_users(dict_tweets,usuario1,usuario2)
	actual_palabra=generar_primer_palabra(list_tweets_users)
	palabras_tweet_pseudoaleatorio=[actual_palabra,]
	while actual_palabra!='':
		actual_palabra=appendear_palabra_aleatoria(list_tweets_users,actual_palabra,
		palabras_tweet_pseudoaleatorio)
	mostrar_tweet_pseudoaleatorio(usuario1,usuario2,palabras_tweet_pseudoaleatorio)
	agregar_tweet_a_favoritos(palabras_tweet_pseudoaleatorio,usuario1,usuario2)

def dar_dict_tweets():
	'''Va recorriendo por el archivo tweets.csv linea por linea para ir armando un
	diccionario con los nombres
	de los twiters como llave y una lista de todos sus tweets como valor. Finalmente,
	devuelve ese diccionario.'''
	with open('/home/joel/Documentos/Ejercicios de Python/TP N°2:Generador de
	Tweets/tweets.csv') as arch_tweets:
		dict_tweets={}
		lineas_tweets=reader(arch_tweets,delimiter='\t')
		for linea in lineas_tweets:
			try:
				dict_tweets[linea[0]]+=[linea[1],]
			except:
				dict_tweets[linea[0]]=[linea[1],]
		return dict_tweets

def dar_list_tweets_users(dict_tweets,usuario1,usuario2):
	'''Recibe el diccionario de tweets y los usuarios, intentan consultar y guarda la
	lista de tweets
	de cada usuario en una variable. En caso de no encontrarse al usuario,
	lanza un error custom llamado NoExisteUsuario.'''
	if usuario1=='Todos':
		list_tweets_users=[]
		for list_tweets_por_user in dict_tweets.values():
			list_tweets_users+=list_tweets_por_user
		return list_tweets_users
	try:
		list_tweets_users=dict_tweets[usuario1]
		if usuario2!='':
```

```
                        list_tweets_users+=dict_tweets[usuario2]
        except KeyError:
                raise NoExisteUsuario('Algún usuario es inexistente en el archivo
                tweets.csv.')
        return list_tweets_users

def generar_primer_palabra(list_tweets_users):
        '''Recibe la lista de todos los tweets de los usuarios y devuelve la primer palabra
        de alguno de ellos.'''
        cant_tweets=len(list_tweets_users)-1
        tweet_primera_palabra=list_tweets_users[random.randint(0,cant_tweets)]
        list_tweet_primera_palabra=tweet_primera_palabra.split(' ')
        primera_palabra=list_tweet_primera_palabra[0]
        return primera_palabra

def
appendear_palabra_aleatoria(list_tweets_users,actual_palabra,palabras_tweet_pseudoaleatorio
):
        '''Recibiendo la lista de tweets de los usuarios y la ultima palabra agregada de la
        lista de palabras
        que van a formar parte del tweet pseudo-aleatorio. Va appendeando en una lista las
        palabras que le siguen
        a la ultima palabra agregada en los tweets. La que más se repita va a ser la que
        tenga más probabilidades
        de ser la proxima palabra del tweet pseudo-aleatorio. Finalmente, devuelve la
        palabra elegida.'''
        list_palabras=[]
        for tweet in list_tweets_users:
                list_tweet=tweet.split(' ')
                contador=0
                for palabra in list_tweet:
                        if contador==1:
                                list_palabras.append(palabra)
                                contador=0
                        if actual_palabra.lower()==palabra.lower():
                                contador+=1
        if list_palabras==[]:
                return ''
        palabra_aleatoria=random.choice(list_palabras)
        if actual_palabra[-1]=='.' or actual_palabra[-1]==';':
                palabra_aleatoria=palabra_aleatoria[0].upper()+palabra_aleatoria[1:]
        else:
                palabra_aleatoria=palabra_aleatoria.lower()
        actual_palabra=palabra_aleatoria
        palabras_tweet_pseudoaleatorio.append(actual_palabra)
        return actual_palabra

def mostrar_tweet_pseudoaleatorio(usuario1,usuario2,palabras_tweet_pseudoaleatorio):
        '''Imprime el tweet pseudo-aleatorio y por quien fue generado.'''
        print(f'Tweet generado por:{usuario1}      {usuario2}.\n')
        tweet_pseudoaleatorio=' '.join(palabras_tweet_pseudoaleatorio)
        print(f'{tweet_pseudoaleatorio}\n')

def agregar_tweet_a_favoritos(palabras_tweet_pseudoaleatorio,usuario1,usuario2):
        '''Recibe la lista de palabras del tweet pseudo-aleatorio y el nombre de los
        usuarios.
        Pregunta al usuario si quiere agregar el tweet pseudo-aleatorio a favoritos.
        En caso de que responda afirmativamente, escribe o agrega en un archivo el tweet
        pseudo-aleotorio
        en una linea.'''
        valores_positivos=['y','se','s','yes','yeah','si','positivo','+']
        valores_negativos=['n','no','nel','negativo','-']
```

```
        while True:
                si_o_no=input('¿Desea agregar este tweet a favoritos?[s/n]: ')
                si_o_no=si_o_no.lower()
                if si_o_no in valores_negativos:
                        return
                if si_o_no in valores_positivos:
                        break
                print('Por favor ingrese un valor positivo o negativo sin espacios.')
        tweet_pseudoaleatorio=' '.join(palabras_tweet_pseudoaleatorio)
        tweet_pseudoaleatorio=f'Generado
        por:{usuario1}\t{usuario2}.\t{tweet_pseudoaleatorio}.\n'
        with open('/home/joel/Documentos/Ejercicios de Python/TP N°2:Generador de
        Tweets/favoritos.csv','a') as arch_fav:
                arch_fav.write(tweet_pseudoaleatorio)

class NoExisteUsuario(Exception):
        pass

def trending_main(top_n):
        '''Imprime tos temas más comunes (palabras precedidas por un hashtag #) de los que
        se habla
        en nuestros tweets almacenados.'''
        dict_hashtags=dar_dict_hashtags()
        list_trending=sacar_trending_list(dict_hashtags,top_n)
        mostar_trending_top(list_trending)

def dar_dict_hashtags():
        '''Lee los tweets almacenados y genera un diccionario con los hashtags como llave y
        como valor su número
        popularidad para luego devolver ese diccionario.'''
        dict_hashtags={}
        valores_invalidos=[',',';','.','?','!']
        with open('/home/joel/Documentos/Ejercicios de Python/TP N°2:Generador de
        Tweets/tweets.csv') as arch_tweets:
                tweets_hashtag=reader(arch_tweets,delimiter='#')
                for hashtag_linea in tweets_hashtag:
                        if len(hashtag_linea)==2:
                                hashtag=hashtag_linea[1].split(' ')
                                hashtag=hashtag[0]
                                if hashtag[-1] in valores_invalidos:
                                        hashtag=hashtag[:-1]
                                if hashtag in dict_hashtags:
                                        dict_hashtags[hashtag]+=1
                                else:
                                        dict_hashtags[hashtag]=1
                return dict_hashtags

def sacar_trending_list(dict_hashtags,top_n):
        '''Recibe el número top y la lista con todos los hashtags, los ordena de mayor a
        menor popularidad
        y los agrega a una lista que va a devolver.'''
        list_trending=[]
        list_values=list(dict_hashtags.values())
        list_values_ordenado=sorted(list_values)
        list_values_ordenado=list_values_ordenado[::-1]
        for i in list_values_ordenado:
                if len(list_trending)==int(top_n):
                        break
                for hashtag in dict_hashtags:
                        tuple_hashtag=(hashtag,i)
                        if dict_hashtags[hashtag]==i and not(tuple_hashtag in list_trending):
                                list_trending.append(tuple_hashtag)
```

```
                        break
        return list_trending

def mostar_trending_top(list_trending):
        '''Recibiendo la lista trengding top pedido va imprimiendo uno debajo del otro cada
        hashtag en orden.'''
        indice=1
        for i in list_trending:
                print(f'{indice}. {i[0]}: {i[1]}')
                indice+=1

def favoritos_main(cant_favs=-1):
        '''Imprime todos los tweets favoritos almacenados. La cantidad de favoritos a
        mostrar es opcional,
        pero de ser ingresada, solo mostrará por pantalla dicha cantidad en orden
        cronológico descendente
        (de más recientes a menos recientes).'''
        list_fav=dar_list_fav()
        mostar_tweets_favs(list_fav,cant_favs)

def dar_list_fav():
        '''Lee el archivo favoritos.csv y va insertando en una lista las lineas con tweets
        de más recientes
        a menos recientes.Finalmente, devuelve la lista.'''
        list_fav=[]
        with open('/home/joel/Documentos/Ejercicios de Python/TP N°2:Generador de
        Tweets/favoritos.csv') as arch_fav:
                for linea in arch_fav:
                        tweet_fav=linea
                        list_fav.insert(0,tweet_fav)
        return list_fav

def mostar_tweets_favs(list_fav,cant_favs):
        '''Se encarga de imprimir cada tweet de la lista que recibe, hasta satisfacer la
        cantidad solicitada.'''
        contador=0
        for i in list_fav:
                if contador==int(cant_favs):
                        break
                print(i)
                contador+=1

comportamiento_en_consola()
```