PRÁCTICA 10

Archivos.

## Alumno: Robles Martínez Héctor

## Número de Cuenta UNAM: 317356548

## Semestre: Tercer semestre de Ingeniería en Computación (2016)

## Semestre Actual: 2021-1

## Profesor: Jorge A. Solano G.

Objetivo:

Conocer e identificar los aspectos sobre los archivos, como las operaciones, el tipo de

acceso y organización lógica.

Instrucciones:

A partir de un archivo XML con los siguientes datos:

<record>

<user>Matsoft</user>

<password>Avery</password>

<email>achapleo0@npr.org</email>

</record>

• Crear una función que permita obtener todos los datos almacenados, generando una

lista de diccionarios: [user = Matsoft, password = Avery,email = achapleo0@npr.org]

• Una vez terminada la búsqueda genera las siguientes estadísticas en funciones dis-

tintas y guardarlas en disco (en un archivo):

– Número y lista de passwords iguales.

– Número y password con mayor incidencia.

– Número de empresas (dependiendo del correo).

– Número y lista de correos iguales.

• Crea las graficas del tiempo de ejecución de cada una de las funciones anteriores.

• Genera una gráfica de pie con los números obtenidos de las funciones.

Resultados obtenidos:

import matplotlib.pyplot as plt

cont = 0

def cleanData(data):

data = data.replace(data[-1], '')

moreThan = data.find('>')

lessThan = data.find('</')

data = data[moreThan+1:lessThan]

return data

def addToDicc(dicc, data):

global cont

cont += 1

if data in dicc:

dicc[data] += 1

else:

dicc[data] = 1

return dicc

def getMaxConcurrency(dicc):

maxConc = max(dicc, key = dicc.get)

return maxConc

def personalizeDatasToPlot(dicc):

dataDicc = list(dicc.values())

aliasDicc = list(dicc.keys())

dataToPlot = []

aliasToPlot = []

k = 0

for i in range(len(dataDicc)):

if dataDicc[i] > 3:

dataToPlot.append(dataDicc[i])

aliasToPlot.append(aliasDicc[i])

else:

k += 1

dataToPlot.append(k)

aliasToPlot.append('oters')

return dataToPlot, aliasToPlot

def plotData(dicc, windowName):

moreFrequentlyDicc = getMaxConcurrency(dicc)

fig, ax = plt.subplots()

dataDicc, aliasDicc = personalizeDatasToPlot(dicc)

g = ax.pie(dataDicc, labels = aliasDicc, labeldistance = 1, shadow = True, autopct = '%1.1f%%', pctdistance = 0.9)

fig.canvas.set\_window\_title(windowName)

plt.show()

def formatDicc(dicc):

diccStr = str(dicc)

diccStr = diccStr[1:-1]

diccStr = diccStr.replace(', ', '\n')

diccStr += '\n'

return diccStr

def writeToFile(dicc, string):

try:

fileDicc = open(string + '.txt', 'w+')

fileDicc.write(formatDicc(dicc))

fileDicc.close()

return True

except:

return False

def extractUserData():

global cont

x = range(0, 2600)

y = []

for i in x:

names = {}

emails = {}

passwords = {}

concurrencies = {}

genders = {}

shirtSizes = {}

try:

file = open('dataset.xml', 'r')

except:

return

line = 'algo'

j = 0

while line != '' and j <= i:

j += 1

if line == ' <record>':

name = file.readline()

email = file.readline()

password = file.readline()

concurrency = file.readline()

gender = file.readline()

shirtSize = file.readline()

name = cleanData(name)

names = addToDicc(names, name)

email = cleanData(email)

at = email.find('@')

dot = email.find('.')

email = email[at+1:dot]

emails = addToDicc(emails, email)

password = cleanData(password)

passwords = addToDicc(passwords, password)

concurrency = cleanData(concurrency)

concurrencies = addToDicc(concurrencies, concurrency)

gender = cleanData(gender)

genders = addToDicc(genders, gender)

shirtSize = cleanData(shirtSize)

shirtSizes = addToDicc(shirtSizes, shirtSize)

try:

line = file.readline()

line = line.replace(line[-1], '')

except:

break

if file:

file.close()

y.append(cont)

cont = 0

plotData(names, 'Most Frequent Names')

plotData(emails, 'Most Frequent Emails')

plotData(passwords, 'Most Frequent Passwords')

plotData(concurrencies, 'Most Frequent Coins type')

plotData(genders, 'Most Frequent Genders')

plotData(shirtSizes, 'Most Frequent ShirtSizes')

if writeToFile(names, 'names'):

print('Names has beenn writed succesfully')

else:

print('An unexpected error has ocurred')

if writeToFile(emails, 'emails'):

print('Emails has beenn writed succesfully')

else:

print('An unexpected error has ocurred')

if writeToFile(passwords, 'passwords'):

print('Passwords has beenn writed succesfully')

else:

print('An unexpected error has ocurred')

if writeToFile(concurrencies, 'concurrencies'):

print('Concurrencies has beenn writed succesfully')

else:

print('An unexpected error has ocurred')

if writeToFile(genders, 'genders'):

print('Genders has beenn writed succesfully')

else:

print('An unexpected error has ocurred')

if writeToFile(shirtSizes, 'shirtSizes'):

print('ShirtSizes has beenn writed succesfully')

else:

print('An unexpected error has ocurred')

print(cont)

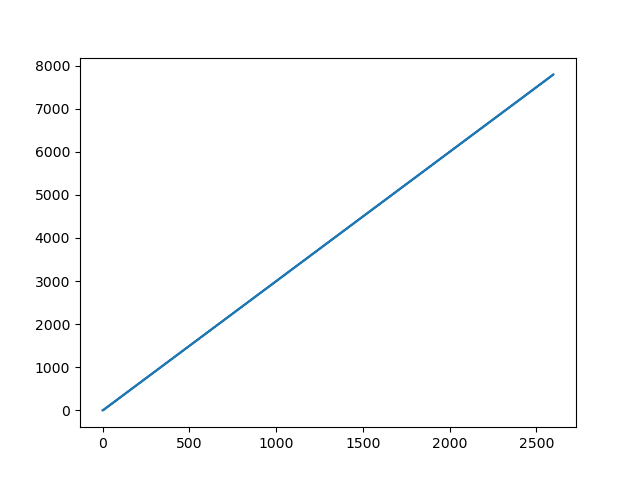
plt.plot(x, y)

plt.savefig('time.png')

plt.show()

if \_\_name\_\_ == '\_\_main\_\_':

extractUserData()



Conclusiones:

Esta práctica me sirvió para comprender cómo se manejan los archivos en python y en el sistema para poder hacer que interactúen el sistema y los archivos