Universidad del Desarrollo
Facultad de Ingeniería

Taller de Programación

Certamen 2

19 de Noviembre de 2019

Instrucciones:

- Lea atentamente el enunciado de cada uno de los problemas.
- Elija solo **DOS** de los TRES problemas del certamen.
- Para cada problema cree un archivo.py distinto. El nombre del archivo debe ser el número del problema (uno.py, dos.py o tres.py)
- Comprima los problemas en un solo archivo **ZIP** y subalo a la sección Evaluación en http://canvas.udd.cl. Solo tiene una oportunidad para subir sus respuestas.
- Recuerde que usaremos un software de detección de plagio para detectar copia.

(3pts) Problema 1. Cuenta Vocales

Programe la función `vocales(texto)` que recibe un texto, y que retorne un diccionario con las siguientes características:

1. las llaves (`keys`) del diccionario deben corresponder a las palabras del texto y estas deben aparecer una sóla vez en el diccionario
2. los valores `(values)` del diccionario deben corresponder al número de vocales que contiene la palabra llave `key` .

Por **ejemplo**, si su función recibe el texto:

```
ningun canon borrara el surco de tu arrozal el derecho de vivir en paz
```

, el diccionario que debe retornar es:

```
{'ningun': 2, 'canon': 2, 'borrara': 3, 'el': 1, 'surco': 2, 'de': 1, 'tu': 1,
 'arrozal': 3, 'derecho': 3, 'vivir': 2, 'en': 1, 'paz': 1}
```

Nota: asuma que el texto que se ingresará no contiene números ni signos de puntuación, y que todo el texto será ingresado en minúsculas.

Respuesta posible

```python
def vocales(texto):
    vocales = ['a', 'e', 'i', 'o', 'u']

    dic_vocales = {}
    texto_en_palabras = set(texto.split())
    palabras = list(texto_en_palabras)
    for i in palabras:
        palabra_n = i
        contador = 0
        for e in palabra_n:
            if e in vocales:
                contador = contador + 1

        dic_vocales[palabra_n] = contador
    return(dic_vocales)

def vocales(texto):
    vocales_dic = dict()
    palabras = texto.split() #Paso el texto a una lista

    for palabra in palabras: #Guardo las palabras unicas en un diccionario
        if palabra in vocales_dic:
            vocales_dic[palabra] += 1
        else:
            vocales_dic[palabra] = 1

    for llave, valor in vocales_dic.items(): #Reemplazo los valores
        contador=0
        for e in llave:
            if e in ['a','e','i','o','u']:
                contador +=1
        vocales_dic[llave]=contador


    return vocales_dic

mensaje = 'ningun canon borrara el surco de tu arrozal el derecho de vivir en paz'
conteo = vocales(mensaje)
print(conteo)
```

Puntaje

- 0.6 Separa y guarda en un set (y luego una lista) cada palabra del texto
- 0.4 Usa un ciclo para recorrer cada palabra del texto

- 0.4 Usa un ciclo para recorrer cada letra de cada palabra
- 0.8 Cuenta el número de vocales de cada palabra
- 0.8 Llena el diccionario con cada palabra y número de vocales que contiene

(3pts) Problema 2. Cámara de Diputados

Dada la contingencia nacional, usted decide realizar un estudio sobre el desempeño de los diputados y su gastos operacionales. Para esto, usted cuenta con información obtenida de la web, almacenada en el archivo `data_diputados.txt`. Este archivo contiene 5 datos: el `id` del diputado, su `nombre`, `partido_político`, `gasto_operacional`, y el `total_asistencias`, respectivamente. Escriba un programa que responda la pregunta de investigación: **¿cuál es el diputado con el mayor gasto operacional de la cámara, y a qué partido pertenece?**.

Respuesta posible

```
f = open('data_diputados.txt','r')
ids = []
nombres = []
gasto = []
partido = []
asistencias = []
for l in f:
    L=l.split(',')
    nombres.append(L[1])
    gasto.append(L[3])
    partido.append(L[2])
f.close()

gasto_max = max(gasto)

nombres_gasto_max = []
gastos_max = []

for i in range(len(gasto)):
    if gasto[i] == gasto_max:
        nombres_gasto_max.append(nombres[i])
        gastos_max.append(gasto[i])
        print(nombres[i]+ ' ' + partido[i] + ' $' + str(gasto[i]))
```

Puntaje

- 0.8 abre correctamente el archivo y lee los datos, cerrando el archivo al finalizar
- 0.6 guarda correctamente la información de nombre, gasto y partido (0.2 c/u)
- 0.6 calcula correctamente el maximo de gastos
- 0.6 obtiene el nombre y partido del diputado con maximo gasto
- 0.4 imprime el nombre y partido

(3pts) Problema 3. Similitud

Una de las principales tareas en el campo de la Inteligencia Artificial es reconocer en qué medida dos elementos (imágenes, texto, música, etc...) son similares. Esto sirve para programar algoritmos sencillos que permiten a plataformas como Spotify o Youtube recomendarte canciones o videos, basándose en su similitud.

Una de las medidas más utilizadas para calcular qué tan parecidos son dos elementos es la **Similitud de Kowalski**. Esta medida representa un elemento como un vector, donde cada dimensión corresponde a una característica de los elementos a comparar. Por ejemplo, para el caso canciones las caracteristicas de los vectores pueden ser los bpm, el estilo musical, etc.

La **Similitud de Kowalski** para los vectores X e Y de n-dimensiones en $\mathbb{R}^n$ está definida como:

$$kowalski(X, Y) = \frac{X \cdot Y}{\sqrt{X \cdot X}\sqrt{Y \cdot Y}}$$

, donde $X \cdot X$ es el **producto punto** entre los vectores X y X, y $X \cdot Y$ es el producto punto entre los vectores X e Y.

Por **ejemplo**, si $X = [1, 0, 1]$, e $Y = [0, 1, 1]$ en $\mathbb{R}^3$, la **Similitud Kowalski** corresponde a 0.5, ya que:

- $X \cdot Y = [1, 0, 1] \cdot [0, 1, 1] = 1$
- $\sqrt{X \cdot X} = [1, 0, 1] \cdot [1, 0, 1] = \sqrt{2}$
- $\sqrt{Y \cdot Y} = [0, 1, 1] \cdot [0, 1, 1] = \sqrt{2}$

, y luego $\frac{1}{\sqrt{2}\sqrt{2}} = 1/2 = 0.5$.

Implemente la función `kowalski(x,y)` que retorna la Similitud de Kowalski, asumiendo que los vectores a comparar son listas. Use el código del cuadro para probar su función.

Nota: recuerde que el producto punto entre dos vectores A y B está definido como $A \cdot B = \sum_{i=1}^{n} A_i B_i$.

Respuesta posible

```python
from math import sqrt

def kawalski(x, y):
    num = 0
    for i in range(len(x)):
        num += x[i]*y[i]

    d1 = 0
    for a in x:
        d1 += a*a
    d1 = sqrt(d1)

    d2 = 0
    for a in y:
        d2 += a*a
    d2 = sqrt(d2)
    return num / (d1*d2)

x = [1, 0, 1]
y = [0, 1, 1]
sim = kowalski(x, y)
print('similaridad:', sim)
```

Puntaje

- 1.3 calcula el producto punto del numerador para cualquier n
- 1.3 calcula el producto punto del denominador para cualquier n
- 0.4 calcula correctamente la división
- Si solo calcula para vectores de 3 dimensiones (o un número fijo de dimensiones) tiene 1 pto.