UNIVERSIDAD NACIONAL AUTÓNOMA DE MÉXICO

FACULTAD DE ESTUDIOS SUPERIORES ARAGÓN

Ingeniería en Computación

DISEÑO Y ANALISIS DE ALGORITMOS

Tarea 8 creación de diccionarios

Profesor: Marcelo Pérez Medel

Manzano Ponce Josué

Olvera Martínez Leonardo

Ortega Batún Luis Fernando

Grupo: 1507

Fecha: jueves 20 de noviembre de 2023

Haga un programa que cree un diccionario y obtenga diccionarios a partir de textos en:

- Inglés
- Francés
- Alemán
- Portugués
- Italiano

Esta tarea tiene un parecido a la tarea numero 2: Cifrado Cesar, en donde la parte B de la tarea anteriormente mencionada nos pide que creemos un diccionario en español, así que partiremos de este mismo código haciendo modificaciones para mejorarlo y cumplir los requisitos que se nos piden en esta nueva tarea.

ITALIANO

Para empezar, nos planteamos la magnitud de la creación de dichos diccionarios, tendríamos que buscar texto en cada idioma que fuera lo suficientemente largo y variado para poder crear un diccionario con una amplia gama de palabras para hacer el cifrado por sustitución.

italiano_palabras

Decidimos empezar con el diccionario en italiano, ya que, según nuestro análisis, el segundo texto estaba en italiano. Comenzamos buscando en publicaciones de periódicos de Italia, textos, libros, fue una tarea agotadora, pero la cantidad de caracteres que estábamos generando y de datos almacenados era abrumadora, meterlos directamente a un programa generaría problemas y es entonces que tuvimos la idea de meterlo a un archivo de extensión *.txt* y a partir de ese archivo tomaríamos dichas palabras para comenzar el diccionario.

Todas las palabras, texto que pudimos encontrar en italiano se fue a un archivo llamado “italiano_palabras.txt”, una vez que consideramos prudente la cantidad de información que teníamos empezamos con el código para almacenar nuestras palabras.

```
def cargaCifrado():
    with open("italiano_palabras.txt", 'r', encoding='utf-8') as archivo:
        contenido = archivo.read()
    return contenido

a = cargaCifrado()
```

El código fue el anterior, en donde definimos una función llamada “*cargaCifrado*” donde abrimos nuestro archivo, pero al ser un idioma de una lengua romance

tuvimos un problema con caracteres como: á, é, ö, ‘, etc. Así que convertimos nuestros datos extraídos a un UTF-8, después de esto ya no teníamos problemas.

Guardamos nuestro texto en una variable *a.*

mostrar_caracteres

Nuestro texto al estar en una lengua romance tenía bastantes caracteres que al momento de empezar nuestro proceso de descifrado nos generaría problemas. Creamos una función que nos mostrara cada uno de los caracteres que había en nuestro nuevo texto.

```
lista = []
def mostrar_caracteres(texto):
    texto = texto.lower()
    for c in texto:
        if c not in lista:
            lista.append(c)
    lista.sort()
    print (lista)

mostrar_caracteres(a)
```

Para mostrar los caracteres, creamos una lista y en nuestra función mostrar_caracteres le damos una cadena, esta cadena será la variable *a*, en donde almacenamos nuestro conjunto de palabras en italiano. Tomamos nuestra cadena y la hacemos minúsculas (evitando tener variables repetidas como: a, A, b, B, etc.), posteriormente recorremos cada carácter de nuestro texto y si encuentra un carácter que no se encuentre en la lista, lo añade.

Nuestro código, usando nuestro conjunto de datos en italiano nos muestra los siguientes caracteres:

```
= RESTART: C:\Users\usuario\Desktop\Uni\2024-1\Diseño y análisis de algoritmos\P
royecto_Final_Equipo_17\italiano_crear_diccionario.py
[' ', '!', '"', '%', "'", '(', ')', '+', ',', '-', '.', '/', '0', '1', '2', '3',
 '4', '5', '6', '7', '8', '9', ':', ';', '?', '[', ']', '_', 'a', 'b', 'c', 'd',
 'e', 'f', 'g', 'h', 'i', 'j', 'k', 'l', 'm', 'n', 'o', 'p', 'q', 'r', 's', 't',
 'u', 'v', 'w', 'x', 'y', 'z', '«', '»', 'à', 'á', 'â', 'è', 'é', 'ê', 'ë', 'ì',
 'í', 'î', 'ï', 'ò', 'ó', 'ô', 'ù', 'ú', 'û', 'ü', '–', '—', '’', '‘', '“', '€']
```

Como podemos ver, tiene caracteres extraños que nos afectarían al momento de hacer nuestro diccionario ([, %, ¿, ¡, etc.) y también caracteres que podemos sustituir (ö, á, ô, etc.).

Limpieza

Ahora que sabemos todos los caracteres que tiene nuestro texto sigue una fase de "limpieza" en donde tomaremos los caracteres que no deberían estar en nuestro diccionario y los eliminamos o sustituimos, dependiendo del caso.

```
a = a.lower()
a = a.replace("à", "a").replace("á", "a").replace("â", "a").replace("è", "e")
a = a.replace("é", "e").replace("ê", "e").replace("ë", "e").replace("ì", "i")
a = a.replace("í", "i").replace("î", "i").replace("ï", "i").replace("ò", "o")
a = a.replace("ó", "o").replace("ô", "o").replace("ù", "u").replace("ú", "u")
a = a.replace("û", "u").replace("ü", "u")
```

Primero, lo que haremos será sustituir aquellas letras que son válidas en nuestro diccionario, pero pueden causar conflicto (ö, á, ô, etc.). No olvidemos hacer a nuestra cadena *a* minúscula, ya que en caso de tener una Á, no hará el cambio.

```
caracteresEspeciales = ['\n', '!', '"', '%', '"', '(', ')', '+', ',', '-',
                        '.', '/', '0', '1', '2', '3', '4', '5', '6', '7', '8',
                        '9', ':', ';', '?', '[', ']', '_', '«', '»', '–', '—',
                        '”', '“', '”', '€']
for caracter in caracteresEspeciales:
    a = a.replace(caracter, "")
```

Después, lo que haremos será meter nuestros caracteres raros en una lista llamada "*caracteresEspeciales*". Recorreremos nuestro conjunto *a* carácter por carácter y buscaremos si este se encuentra en nuestra lista, de ser así, lo reemplazaremos por un vacío.

Diccionario

Ahora nuestra cadena esta lista para poder realizar el diccionario.

```
palabras = a.split() #hacemos dividimos por palabras
palabras_unicas = set(palabras) #asigna palabras unicas
diccionario = (sorted(palabras_unicas)) #ordenamos alfabeticamente
diccionario = sorted(diccionario, key=lambda x: (len(x), x)) #ordenamos por longitud
diccionario = ' '.join(diccionario)
with open('italiano_diccionario.txt', 'w') as archivo:
    archivo.write(diccionario)
```

Creamos una variable "*palabras*" y le asignamos el valor de nuestra variable *a* usando la función split() para obtener un conjunto de palabras separadas por su espacio. Usando la función set() eliminamos las palabras repetidas. Después ordenamos las palabras de alfabéticamente y longitud usando la función sorted(). Finalmente las agregamos a una variable diccionario, creamos un archivo llamado "italiano_diccionario.txt" y escribimos en él el valor de nuestra última variable.

INGLÉS, FRANCÉS, ALEMÁN, PORTUGUÉS

El proceso para crear los demás diccionarios fue prácticamente el mismo. Solo se hacían modificaciones pequeñas para cada idioma ya que no todos poseían los mismos caracteres en cada texto.

Cada código se mostrará en la documentación junto con las partes de los otros idiomas, pero cada fragmento pertenece a un código distinto para cada idioma, se hará así por fines prácticos para mostrar todos los códigos.

idioma_palabras

Como con el italiano, ahora lo que seguía era crear archivos de extensión .txt, en donde almacenáramos texto para cada idioma, lo cual sería una tarea bastante compleja, pero considerando que éramos 3 integrantes del equipo, pudimos llenar el archivo de manera eficiente.

Para evitar algún problema con el salto de línea, una vez considerábamos que la extensión de nuestros textos era lo suficientemente grande, usando la herramienta de Word, “*reemplazar*”, reemplazamos los saltos de línea (^p) con espacios (“ “), de esta manera obtuvimos un texto de una sola línea.

Usamos el mismo código que el de italiano, pero haciendo las modificaciones para cada idioma:

```
def cargaCifrado():
    with open("portugues_palabras.txt", 'r', encoding='utf-8') as archivo:
        contenido = archivo.read()
    return contenido

def cargaCifrado():
    with open("frances_palabras.txt", 'r', encoding='utf-8') as archivo:
        contenido = archivo.read()
    return contenido

def cargaCifrado():
    with open("ingles_palabras.txt", 'r', encoding='utf-8') as archivo:
        contenido = archivo.read()
    return contenido

def cargaCifrado():
    with open("aleman_palabras.txt", 'r', encoding='utf-8') as archivo:
        contenido = archivo.read()
    return contenido
```

Almacenamos de manera individual cada cadena en una variable *a*

mostrar_caracteres

Nuevamente, como se ha mencionado anteriormente, el código es prácticamente el mismo que el italiano, pero con ligeras modificaciones.

Alemán

```
lista = []
def mostrar_caracteres(texto):
    texto = texto.lower()
    for c in texto:
        if c not in lista:
            lista.append(c)
    lista.sort()
    print (lista)
mostrar_caracteres(a)
```

```
= RESTART: C:\Users\usuario\Desktop\Uni\2024-1\Diseño y análisis de algoritmos\P
royecto_Final_Equipo_17\aleman_crear_diccionario.py
[' ', '!', "'", ',', '-', '.', '0', '1', ':', ';', '?', 'a', 'b', 'c', 'd', 'e',
 'f', 'g', 'h', 'i', 'j', 'k', 'l', 'm', 'n', 'o', 'p', 'q', 'r', 's', 't', 'u',
 'v', 'w', 'x', 'z', '«', '»', 'ß', 'ä', 'ö', 'ü', '–']
[' ', 'a', 'b', 'c', 'd', 'e', 'f', 'g', 'h', 'i', 'j', 'k', 'l', 'm', 'n', 'o',
 'p', 'q', 'r', 's', 't', 'u', 'v', 'w', 'x', 'z']
```

Inglés

```
lista = []
def mostrar_caracteres(texto):
    texto = texto.lower()
    for c in texto:
        if c not in lista:
            lista.append(c)
    lista.sort()
    print (lista)
mostrar_caracteres(a)
```

```
= RESTART: C:\Users\usuario\Desktop\Uni\2024-1\Diseño y análisis de algoritmos\P
royecto_Final_Equipo_17\ingles_crear_diccionario.py
[' ', 'a', 'b', 'c', 'd', 'e', 'f', 'g', 'h', 'i', 'j', 'k', 'l', 'm', 'n', 'o',
 'p', 'q', 'r', 's', 't', 'u', 'v', 'w', 'x', 'y', 'z']
```

Francés

```
lista = []
def mostrar_caracteres(texto):
    texto = texto.lower()
    for c in texto:
        if c not in lista:
            lista.append(c)
    lista.sort()
    print (lista)
mostrar_caracteres(a)
```

```
= RESTART: C:\Users\usuario\Desktop\Uni\2024-1\Diseño y análisis de algoritmos\P
royecto_Final_Equipo_17\frances_crear_diccionario.py
['\n', ' ', '!', '"', '%', '&', "'", '(', ')', '+', ',', '-', '.', '/', '0', '1'
, '2', '3', '4', '5', '6', '7', '8', '9', ':', '>', '?', '[', ']', 'a', 'b', 'c'
, 'd', 'e', 'f', 'g', 'h', 'i', 'j', 'k', 'l', 'm', 'n', 'o', 'p', 'q', 'r', 's'
, 't', 'u', 'v', 'w', 'x', 'y', 'z', '«', '°', '»', 'à', 'â', 'ç', 'è', 'é', 'ê'
, 'ë', 'í', 'î', 'ï', 'ó', 'ô', 'ù', 'û', 'œ', '\u2009', '–', '’', '…', '\ufeff']
```

Portugués

```
lista = []
def mostrar_caracteres(texto):
    texto = texto.lower()
    for c in texto:
        if c not in lista:
            lista.append(c)
    lista.sort()
    print (lista)
mostrar_caracteres(a)
```

```
= RESTART: C:\Users\usuario\Desktop\Uni\2024-1\Diseño y análisis de algoritmos\P
royecto_Final_Equipo_17\portugues_crear_diccionario.py
[' ', '!', "'", '(', ')', ',', '-', '.', '/', '0', '1', '2', '3', '4', '5', '6',
 '7', '8', '9', ':', ';', '?', '[', ']', '^', '_', 'a', 'b', 'c', 'd', 'e', 'f',
 'g', 'h', 'i', 'j', 'k', 'l', 'm', 'n', 'o', 'p', 'q', 'r', 's', 't', 'u', 'v',
 'w', 'x', 'y', 'z', 'ª', '«', '»', '¿', 'à', 'á', 'â', 'ã', 'æ', 'ç', 'é', 'ê',
 'í', 'î', 'ó', 'ô', 'õ', 'ú', 'œ', '→']
```

Limpieza

De igual manera que se hizo con el italiano, hay que limpiar los caracteres que podríamos sustituir o eliminar de nuestro texto para tener un diccionario que cumpla con nuestras necesidades.

Alemán

```
a = a.lower()
a = a.replace("ß", "s").replace("ä", "a").replace("ö", "o").replace("ü", "u")
caracteresEspeciales = ['!', '"', ',', '-', '.', '0', '1', ':', ';', '?',
                        '«', '»', '–']
for caracter in caracteresEspeciales:
    a = a.replace(caracter, "")
lista = []
mostrar_caracteres(a)
```

Inglés

```
a = a.lower()
a = a.replace("à", "a").replace("á", "a").replace("â", "a").replace("ã", "a")
a = a.replace("æ", "a").replace("ç", "c").replace("é", "e").replace("ê", "e")
a = a.replace("í", "i").replace("î", "i").replace("ó", "o").replace("ô", "o")
a = a.replace("õ", "o").replace("ú", "u").replace("œ", "o")
caracteresEspeciales = ['!', '"', '(', ')', ',', '-', '.', '/', '0', '1',
                        '2', '3', '4', '5', '6', '7', '8', '9', ':', ';', '?',
                        '[', ']', '^', '_', 'ª', '«', '»', '¿', '→']
for caracter in caracteresEspeciales:
    a = a.replace(caracter, "")
```

Francés

```
a = a.lower()
a = a.replace("à", "a").replace("â", "a").replace("ç", "c").replace("è", "e")
a = a.replace("é", "e").replace("ê", "e").replace("ë", "e").replace("í", "i")
a = a.replace("î", "i").replace("ï", "i").replace("ó", "o").replace("ô", "o")
a = a.replace("ù", "u").replace("û", "u").replace("œ", "o")

caracteresEspeciales = ['\n', '!', '"', '%', '&', "'", '(', ')', '+', ',',
                        '-', '.', '/', '0', '1', '2', '3', '4', '5', '6', '7',
                        '8', '9', ':', '>', '?', '[', ']', '«', '°', '»','ᵉ',
                        '–', '”', '…', '\ufeff']
for caracter in caracteresEspeciales:
    a = a.replace(caracter, "")
```

Portugués

```
a = a.lower()
a = a.replace("à", "a").replace("á", "a").replace("â", "a").replace("ã", "a")
a = a.replace("æ", "a").replace("ç", "c").replace("é", "e").replace("ê", "e")
a = a.replace("í", "i").replace("î", "i").replace("ó", "o").replace("ô", "o")
a = a.replace("õ", "o").replace("ú", "u").replace("œ", "o")

caracteresEspeciales = ['!', '"', '(', ')', ',', '-', '.', '/', '0', '1',
                        '2', '3', '4', '5', '6', '7', '8', '9', ':', ';', '?',
                        '[', ']', '^', '_', 'ª', '«', '»', '¿', '→']

for caracter in caracteresEspeciales:
    a = a.replace(caracter, "")
```

Como se puede observar, en algunos casos si cambian algunas cosas como los replace o las listas de caracteresEspeciales, sin embargo, es el mismo principio para hacer una limpieza de caracteres indeseados.

Diccionario

Finalmente, la creación del diccionario para estos idiomas. El código es prácticamente el mismo que el del diccionario italiano, solo con la diferencia que cambia el nombre de como llamamos al diccionario, así que no hace falta explicar los 4 códigos de manera extensiva.

Alemán

```
palabras = a.split() #hacemos dividimos por palabras
palabras_unicas = set(palabras) #asigna palabras unicas
diccionario = (sorted(palabras_unicas)) #ordenamos alfabeticamente
diccionario = sorted(diccionario, key=lambda x: (len(x), x)) #ordenamos por longitud
diccionario = ' '.join(diccionario)
with open('aleman_diccionario.txt', 'w', encoding = 'utf-8') as archivo:
    archivo.write(diccionario)
```

Inglés

```
palabras = a.split() #hacemos dividimos por palabras
palabras_unicas = set(palabras) #asigna palabras unicas
diccionario = (sorted(palabras_unicas)) #ordenamos alfabeticamente
diccionario = sorted(diccionario, key=lambda x: (len(x), x)) #ordenamos por longitud
diccionario = ' '.join(diccionario)
with open('ingles_diccionario.txt', 'w') as archivo:
    archivo.write(diccionario)
```

Francés

```
palabras = a.split() #hacemos dividimos por palabras
palabras_unicas = set(palabras) #asigna palabras unicas
diccionario = (sorted(palabras_unicas)) #ordenamos alfabeticamente
diccionario = sorted(diccionario, key=lambda x: (len(x), x)) #ordenamos por longitud
diccionario = ' '.join(diccionario)
with open('frances_diccionario.txt', 'w') as archivo:
    archivo.write(diccionario)
```

Portugués

```
palabras = a.split() #hacemos dividimos por palabras
palabras_unicas = set(palabras) #asigna palabras unicas
diccionario = (sorted(palabras_unicas)) #ordenamos alfabeticamente
diccionario = sorted(diccionario, key=lambda x: (len(x), x)) #ordenamos por longitud
diccionario = ' '.join(diccionario)
with open('portugues_diccionario.txt', 'w') as archivo:
    archivo.write(diccionario)
```

Primero separamos las palabras, eliminamos las repetidas, ordenamos alfabeticamente y después por longitud y finalmente la agregamos a una variable y esa variable es lo que vamos a escribir en nuestro diccionario.txt.