Universidad de Guadalajara.

Centro Universitario de Ciencias Exactas e Ingenierías.

DIVISIÓN DE TECNOLOGÍAS PARA LA INTEGRACIÓN CIBER-HUMANA.

DEPARTAMENTO DE CIENCIAS COMPUTACIONALES.

Análisis de Algoritmos. / D01 / 24A

ACT 07 - Entrega 1 Huffman.

Nombre de los Profesores. JORGE ERNESTO LOPEZ ARCE DELGADO.

MOISES SOTELO RODRIGUEZ.

Alumno: Garcia Parra Jair Omar.

Código: 218534061

23 de Abril del 2024.

Introducción.

El algoritmo de Huffman es un algoritmo que nos da un método que nos permite comprimir información usando la recodificación de bytes. La técnica consiste en asignar a cada byte un código binario compuesto por una cantidad de bits lo más corta posible. Esta cantidad será variable y dependerá de la probabilidad de ocurrencia del byte.

Objetivos.

Crear un programa en Python que comprima un archivo de texto utilizando el algoritmo de compresión de Huffman.

Desarrollo.

```
1   from tkinter import *
2   from tkinter import ttk
3   from tkinter.filedialog import askopenfilename
4   from collections import Counter
5
6   root=Tk() #Creamos la ventana principal
7
```

Aquí definimos las librerías necesarias para el funcionamiento, además creamos la ventana principal root. (Se usa tkinter como ttk ya que de este modo el aspecto visual tiene un aspecto más moderno).

```
39  #Ventana inicial
40  root.title("Actividad 07 - Frontend") # Titulo de la ventana
41  root.geometry("1100x700") #Tamaño de la ventana
42  font_style=("Arial",12) #Fuente de el texto
43  root.configure(bg="black") #Color de la ventana
44
45  #Estilos de boton
46  style=ttk.Style()
47  style.configure("TButton",font=font_style, bg="black", fg="black")
48
49  principal()
50
51  root.mainloop()
```

Debajo de las funciones declaramos los estilos de la ventana como el título, tamaño, fuente, y color, de texto asi como de botón. Después llamamos a la función principal y corremos la interfaz gráfica con mainloop.

```python
def principal(): #Esta es la funcion principal donde se Mostraran los Botones para elegir el archivo y realizar los procesos
    for widget in root.winfo_children():
        widget.destroy() #Esta funcion borra el contenido del widget
    ttk.Label(root, text="Menú de seleccion de archivo", font=font_style, background="black", foreground="white").place(x=480, y=15) # Etiqueta i
    archivos = ttk.Label(root, text="Seleccione Archivo:", font=font_style, background="black", foreground="white") #Etiqueta de seleccionar arch
    archivos.place(x=505, y=50) #posicion de la etiqueta archivo
    ttk.Button(root, text="Examinar", command=lambda: explorador(archivos)).place(x=350, y=100) #Boton que nos lleva a elegir el archivo en el de
    ttk.Button(root, text="Cerrar Ventana", command=root.destroy).place(x=660, y=100) #Boton que cierra todo
```

En la función principal se tiene un for que limpia el widget en caso de tener algo antes, Después imprime la etiqueta que nos dice que se trata del menú de selección de archivos, se nos invita a abrir un archivo y tenemos los botones de abrir archivo o examinar y cerrar ventana.

```python
def explorador(archivos): #Funcion para abrir el explorador de archivos
    f_path = askopenfilename(initialdir="/",
    title="Seleccion de Archivo", filetypes=(("*.txt*","txt"),("Todos los Archivos","*.*")))
    archivos.configure(text="Archivo Abierto: \n"+f_path)#realiza la apertura del explorador de archivos y lo
    f_path.endswith('.txt')
    leer_texto(f_path)
    ttk.Button(root, text="Comprimir", command=None).place(x=500, y=200) #Este boton continuara el programa ha
    ttk.Button(root, text="Descomprimir", command=None).place(x=500, y=250) #Este boton continuara el programa
```

Después se nos envía a la función explorador la cual comienza con la parte que se encargara de abrir el archivo desde el explorador usando askopenfilename, limita los archivos a solo .txt y al elegir uno se nos muestra el mensaje de archivo abierto además de la ruta de este mismo, Seguido de esto nos envía a la función de leer texto y se imprimen los botones de Comprimir y Descoprimir.

```python
def leer_texto(f_path):
    with open(f_path, 'r', encoding='utf-8') as f_path: #Aqui se abrira el archivo seleccionado en modo de lectura
        texto=f_path.read()
        #print(texto)
        #ttk.Label(root, text=texto, font=font_style, background="black", foreground="white").place(x=510, y=400) #Mue
    frecuencia = Counter(texto)
    print("Frecuencia de caracteres:")
    for caracter, count in frecuencia.items(): #Se calcula la frecuencia de todos los caracteres usando Counter de la
        print(f"'{caracter}': {count}") #Se imprime la frecuencia de cada carácter
```

En la función de leer texto, se necesitan los datos del archivo abierto que se encuentran en f_path después usamos esto para usar la función open, y leer f_path en modo lectura. Lo guardaremos en texto y seguido de esto guardamos la frecuencia de los caracteres en “frecuencia” para contarlos usando Counter de la librería collections, por ultimo los imprimiremos en la consola con un for para comprobar que los leyó de manera correcta.

Conclusión.

Para mi esta actividad fue un poco mas sencilla que las anteriores ya que eran cosas que ya habíamos realizado anteriormente y la parte de contar los caracteres que era nueva se realizó mediante una librería lo que facilito la comprensión y ejecución.