Actividad 7: Entrega 1 Huffman

Análisis de algoritmia

De Alba Velarde Christian Moises

López Arce Delgado Jorge Ernesto

Sotelo Rodríguez Moises

D01 24A

23/04/2024

Introducción

El presente reporte aborda el desarrollo de un programa en Python para la creación de un compresor de archivos utilizando el algoritmo de Huffman. El propósito de este proyecto es proporcionar una herramienta simple y efectiva para comprimir y descomprimir archivos de texto, aprovechando las ventajas de la codificación de Huffman para reducir el tamaño de los archivos.

Objetivos

El objetivo principal es implementar la funcionalidad de compresión y descompresión de archivos utilizando el algoritmo de Huffman. Así como mejorar la interfaz de usuario para proporcionar una interface intuitiva así como comprimir y descomprimir archivos grandes de manera eficiente.

Desarrollo

Funciones:

LeerArchivo(archivo): Esta función recibe la ruta de un archivo como entrada, lee su contenido y muestra el número de caracteres en el archivo. Utiliza la función open() para abrir el archivo en modo lectura y archivo.read() para leer su contenido. Luego, cuenta el número de caracteres utilizando la función len() y actualiza una etiqueta en la GUI con esta información.

BuscarArchivo(): Esta función abre un cuadro de diálogo para que el usuario pueda seleccionar un archivo. Utiliza filedialog.askopenfilename() para abrir el cuadro de diálogo y archivo_entrada.delete() y archivo_entrada.insert() para mostrar la ruta del archivo seleccionado en un campo de entrada en la GUI.

Comprimir(): Función destinada a implementar la lógica de compresión de archivos. Actualmente está definida, pero sin funcionalidad implementada.

Descomprimir(): Función destinada a implementar la lógica de descompresión de archivos. Al igual que la función Comprimir(), está definida pero sin funcionalidad implementada.

Interfaz de Usuario:

Se utiliza Tkinter para crear una interfaz simple que incluye etiquetas, campos de entrada y botones.

Se muestra la etiqueta "Archivo:" seguida de un campo de entrada donde se muestra la ruta del archivo seleccionado.

Un botón "Examinar" permite al usuario seleccionar un archivo mediante un cuadro de diálogo.

Botones adicionales para las funciones de compresión y descompresión están presentes en la GUI.

Codigo:

```
import tkinter as tk
from tkinter import *
from tkinter import ttk
from tkinter import filedialog
import os

# Funcion para leer el contenido de un archivo y mostrar el numero de caracteres
def LeerArchivo(archivo):
    # Abrir el archivo en modo lectura
    with open(archivo, "r") as archivo:
        # Leer el contenido del archivo
        datos = archivo.read()
        # Contar el numero de caracteres en el archivo
        contador_caracteres = len(datos)
        # Mostrar el numero de caracteres
        contador_label.config(text=f"Caracteres en el archivo: {contador_caracteres}")
        # Imprimir el contenido del archivo
        print(datos)

# Funcion para buscar un archivo
def BuscarArchivo():
    # Abrir el cuadro de dialogo para seleccionar un archivo
    archivo = filedialog.askopenfilename()
    # Limpiar y mostrar la ruta del archivo seleccionado en el campo de entrada
    archivo_entrada.delete(0, tk.END)
    archivo_entrada.insert(0, archivo)
    # Llamar a la funcion LeerArchivo() para mostrar el contenido del archivo
```

```
    LeerArchivo(archivo)

# Funcion para la compresion de archivos
def Comprimir():
    input_file = archivo_entrada.get()
    if not input_file:
        return

# Funcion para la descompresion de archivos
def Descomprimir():
    input_file = archivo_entrada.get()
    if not input_file:
        return

# Crear la ventana principal de la aplicacion
root = tk.Tk()
root.title("Compresor de Archivos Huffman")
root.geometry("777x555")

# Crear un marco principal dentro de la ventana
mainframe = tk.Frame(root)
mainframe.pack(padx=8, pady=8)

# Etiqueta para mostrar el directorio del Archivo
archivo_ventana = tk.Label(mainframe, text="Archivo:")
archivo_ventana.grid(row=0, column=0, padx=10, pady=10)

# Campo de entrada para mostrar la ruta del archivo seleccionado
archivo_entrada = tk.Entry(mainframe, width=50)
archivo_entrada.grid(row=0, column=1, padx=10, pady=10)

# Botón para abrir el cuadro de diálogo y buscar un archivo
buscar_boton = tk.Button(mainframe, text="Examinar", command=BuscarArchivo)
buscar_boton.grid(row=0, column=2, padx=10, pady=10)

# Botón para iniciar la compresión de archivos
comprimir_boton = tk.Button(mainframe, text="Comprimir", command=Comprimir)
comprimir_boton.grid(row=1, column=0, columnspan=3, pady=10)

# Botón para iniciar la descompresión de archivos
descomp_boton = tk.Button(mainframe, text="Descomprimir", command=Descomprimir)
descomp_boton.grid(row=2, column=0, columnspan=3, padx=10, pady=10)

# Muestra el contador de caracteres
```

```
contador_label = tk.Label(mainframe, text="")
contador_label.grid(row=3, column=0, columnspan=3)

# Inicia el bucle de la interface
root.mainloop()
```

Conclusiones

Para esta primera parte lo que se entrega es la parte de interface grafica ya en esta parte se hizo todos los botones así como también se hizo un una manera mas estética mejor para un último usuario así como se agregó un contador de caracteres, en el siguiente avance de la actividad se realizaras la parte de backend la cual se utilizara árbol de huffman para poder comprimir y descomprimir los archivos seleccionados