# CODIGO FUENTE

# -\*- coding: cp1252 -\*-

# Calculadora ASCII

# Autor: [Grupo 3] 10110-G-2

# Versión 2.5.2

# 2016-12-01

#DEFINICIÓN DE CONSTANTES

NUM0 = [['x','x','x','x','x'],['x','.','.','.','x'],['x','.','.','.','x'],['x','.','.','.','x'],['x','.','.','.','x'],['x','.','.','.','x'],['x','x','x','x','x']]

NUM1 = [['.','.','.','.','x'],['.','.','.','.','x'],['.','.','.','.','x'],['.','.','.','.','x'],['.','.','.','.','x'],['.','.','.','.','x'],['.','.','.','.','x']]

NUM2 = [['x','x','x','x','x'],['.','.','.','.','x'],['.','.','.','.','x'],['x','x','x','x','x'],['x','.','.','.','.'],['x','.','.','.','.'],['x','x','x','x','x']]

NUM3 = [['x','x','x','x','x'],['.','.','.','.','x'],['.','.','.','.','x'],['x','x','x','x','x'],['.','.','.','.','x'],['.','.','.','.','x'],['x','x','x','x','x']]

NUM4 = [['x','.','.','.','x'],['x','.','.','.','x'],['x','.','.','.','x'],['x','x','x','x','x'],['.','.','.','.','x'],['.','.','.','.','x'],['.','.','.','.','x']]

NUM5 = [['x','x','x','x','x'],['x','.','.','.','.'],['x','.','.','.','.'],['x','x','x','x','x'],['.','.','.','.','x'],['.','.','.','.','x'],['x','x','x','x','x']]

NUM6 = [['x','x','x','x','x'],['x','.','.','.','.'],['x','.','.','.','.'],['x','x','x','x','x'],['x','.','.','.','x'],['x','.','.','.','x'],['x','x','x','x','x']]

NUM7 = [['x','x','x','x','x'],['.','.','.','.','x'],['.','.','.','.','x'],['.','.','.','.','x'],['.','.','.','.','x'],['.','.','.','.','x'],['.','.','.','.','x']]

NUM8 = [['x','x','x','x','x'],['x','.','.','.','x'],['x','.','.','.','x'],['x','x','x','x','x'],['x','.','.','.','x'],['x','.','.','.','x'],['x','x','x','x','x']]

NUM9 = [['x','x','x','x','x'],['x','.','.','.','x'],['x','.','.','.','x'],['x','x','x','x','x'],['.','.','.','.','x'],['.','.','.','.','x'],['x','x','x','x','x']]

SMAS = [['.','.','.','.','.'],['.','.','x','.','.'],['.','.','x','.','.'],['x','x','x','x','x'],['.','.','x','.','.'],['.','.','x','.','.'],['.','.','.','.','.']]

SMEN = [['.','.','.','.','.'],['.','.','.','.','.'],['.','.','.','.','.'],['x','x','x','x','x'],['.','.','.','.','.'],['.','.','.','.','.'],['.','.','.','.','.']]

SMUL = [['.','.','.','.','.'],['x','.','.','.','x'],['.','x','.','x','.'],['.','.','x','.','.'],['.','x','.','x','.'],['x','.','.','.','x'],['.','.','.','.','.']]

SDIV = [['.','.','.','.','.'],['.','.','.','.','x'],['.','.','.','x','.'],['.','.','x','.','.'],['.','x','.','.','.'],['x','.','.','.','.'],['.','.','.','.','.']]

PUNT = [['.','.','.','.','.'],['.','.','.','.','.'],['.','.','.','.','.'],['.','.','.','.','.'],['.','.','.','.','.'],['.','.','.','.','.'],['x','.','.','.','.']]

ASCII = [NUM0,NUM1,NUM2,NUM3,NUM4,NUM5,NUM6,NUM7,NUM8,NUM9,SMAS,SMEN,SMUL,SDIV,PUNT]

STR = ['0','1','2','3','4','5','6','7','8','9','+','-','\*','/','.']

notFound = "NO se ha encontrado el archivo, por favor intente nuevamente"

notMatch = "El archivo no coincide con nada almacenado \n este error probablemente se deba a un caracter mal ubicado"

linea = "\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\n"

O = 'OPERACIÓN\n'

R = 'RESULTADO\n'

fallo = 'Archivo incorrecto'

errorFila = 'El largo de alguna(s) fila(s) no corresponde(n) al requerido, modifique el archivo'

errorCaracter = "Uno o más caracteres no corresponden a lo solicitado, modifique el archivo"

#IMPORTACIÓN DE FUNCIONES

"""

---- Se importa sys para poder acceder a permisos de la computadora

----- Tkinter nos permite crear una interfaz grafica

----- tkFont nos permite cambiar las fuentes de la interfaz gráfica, es necesario ya que las matrices necesitan la fuente FixedSys

----- tkMessageBox es un módulo que nos permite lanzar diversos mensajes al usuario en forma de cajas

----- tkFileDialog nos permite crear una ventana extra para buscar archivos en el computador

"""

from sys import \*

from Tkinter import \*

import tkFont

import tkMessageBox

from tkFileDialog import \*

#DEFINICIÓN DE FUNIONES

#\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_#

#FUNCIONES QUE SE ENCARGAN DEL ARCHIVO

#Abre un archivo con una operación ASCII y asigna todas las filas dentro de distintos elementos de una lista

def leerArchivo(nombre):

nombre = nombre.strip(".txt")

nombre = nombre + '.txt'

archivo = open(nombre,'r')

fila1 = []

fila2 = []

for i in range(7):

fila1.append(archivo.readline())

if i != 6:

fila2.append(fila1[i][0:len(fila1[i])-1])

else:

fila2.append(fila1[i][0:len(fila1[i])])

if (len(fila2[i]) + 1) % 6 != 0:

fila2 = True

return fila2

for i in range(1,6):

if len(fila2[i]) != len(fila2[i-1]):

fila2 = True

return fila2

return fila2

#Lee las líneas documento y verifica que solo contenga '.' y 'x'

def validacion(archivoEnUnaLinea):

check = True

error = False

fallo = 'Archivo incorrecto'

errorFila = 'El largo de alguna(s) fila(s) no corresponde(n) al requerido, modifique el archivo'

errorCaracter = "Uno o más caracteres no corresponden a lo solicitado, modifique el archivo"

mensaje = ""

if (archivoEnUnaLinea == True):

mensaje = ""

mensaje += "\n" + fallo + "\n" + errorFila

check = False

return mensaje

else:

for conjunto in archivoEnUnaLinea:

for letra in conjunto:

if not((letra == ".")or(letra == "x")):

error = True

if error:

mensaje = ""

mensaje += "\n" + fallo + "\n" + errorCaracter

return mensaje

return check

#\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_#

#FUNCIONES QUE TRANSFORMAN DE ASCII A UN NÚMERO ENTERO

#Transforma el string fila[i] en una fila de "n" elementos (con n como el largo del string de fila[i])

def separarCaracteres(archivoEnUnaLinea):

caracter = []

for i in range(7):

caracter.append([])

caracter[i] = list(archivoEnUnaLinea[i])

return caracter

#Guarda un número ASCII diferente en cada columna de la matriz numero[i][j]

def agruparCaracteres(caracter):

numero = []

for i in range(7):

numero.append([])

for j in range(int(len(caracter[i])/6)+1):

numero[i].append([])

for k in range(5):

numero[i][j].append(caracter[i][k+6\*j])

return numero

#Cada elemento de enteroMatriz[i] representa un número distinto

def agruparNumeros(numero, caracter):

enteroMatriz = []

for i in range(int(len(caracter[0])/6)+1):

enteroMatriz.append([])

for j in range(7):

enteroMatriz[i].append(numero[j][i])

return enteroMatriz

#Inserta un número (u operación) en forma de string dentro de la matriz enteroNumero

#que equivale al número en formato ASCII en la posición [i]

def reemplazar(enteroMatriz):

enteroNumero = []

for i in range(len(enteroMatriz)):

for j in range(len(ASCII)):

if enteroMatriz[i] == ASCII[j]:

enteroNumero.append(STR[j])

return enteroNumero

#Ubica la posición del operador

def posicionOperador(enteroNumero):

indiceOperacion = None

if '+' in enteroNumero:

indiceOperacion = enteroNumero.index('+')

elif '-' in enteroNumero:

indiceOperacion = enteroNumero.index('-')

elif '\*' in enteroNumero:

indiceOperacion = enteroNumero.index('\*')

elif '/' in enteroNumero:

indiceOperacion = enteroNumero.index('/')

return indiceOperacion

#Indica que tipo de operación se va a realizar

def operacion(enteroNumero):

tipoOperacion = '+'

if '+' in enteroNumero:

tipoOperacion = '+'

elif '-' in enteroNumero:

tipoOperacion = '-'

elif '\*' in enteroNumero:

tipoOperacion = '\*'

elif '/' in enteroNumero:

tipoOperacion = '/'

return tipoOperacion

#\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_#

#FUNCIÓN QUE SE ENCARGA DE CALCULAR LOS NÚMEROS

#Transforma los números a enteros y los opera

def calculadora(enteroNumero, indiceOperacion, tipoOperacion):

numeroSt1 = ''

numeroSt2 = ''

error = False

resultadoOperacion = ""

primero = ""

segundo = ""

if indiceOperacion != None:

for i in range(indiceOperacion):

numeroSt1 = numeroSt1 + enteroNumero[i]

primero = int(numeroSt1)

for i in range(indiceOperacion+1, len(enteroNumero)):

numeroSt2 = numeroSt2 + enteroNumero[i]

segundo = int(numeroSt2)

else:

for i in range(len(enteroNumero)):

numeroSt1 = numeroSt1 + enteroNumero[i]

primero = int(numeroSt1)

segundo = 0

if tipoOperacion == '+':

resultadoOperacion = primero + segundo

elif tipoOperacion == '-':

resultadoOperacion = primero - segundo

elif tipoOperacion == '\*':

resultadoOperacion = primero \* segundo

elif tipoOperacion == '/':

if segundo != 0:

resultadoOperacion = round((float(primero) / float(segundo)),4)

else:

resultadoOperacion = "No se puede dividir por CERO"

if (resultadoOperacion != "No se puede dividir por CERO"):

if (resultadoOperacion - int(resultadoOperacion)) == 0:

resultadoOperacion = int(resultadoOperacion)

return resultadoOperacion

#\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_#

#FUNCIONES QUE SE ENCARGAN DE PASAR EL NÚMERO A ASCII

#Transforma el resultado en una lista de números ASCII

def reconvertir(resultado):

listaAscii = []

resultadoString = list(str(resultado))

for i in range(len(resultadoString)):

for j in range(len(STR)):

if resultadoString[i] == STR[j]:

listaAscii.append(ASCII[j])

return listaAscii

#La lista de números ASCII se transforma en una lista con las 7 filas del resultado final

def reordenar(resultadoAscii):

resultado1 = []

resultado2 = []

resultado3 = ''

for i in range(7):

resultado1.append([])

for j in range(len(resultadoAscii)):

resultado1[i].append(resultadoAscii[j][i])

for i in range(7):

resultado2.append('')

for j in range(len(resultado1[i])):

for k in range(5):

resultado2[i] = resultado2[i] + resultado1[i][j][k]

if j != len(resultado1[i]):

resultado2[i] = resultado2[i] + '.'

resultado2[i] = resultado2[i][0:len(resultado2[i])-1]

for i in range(7):

resultado3 = resultado3 + resultado2[i]

if i < 6:

resultado3 = resultado3 + '\n'

return resultado3

#\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_#

#BLOQUE PRINCIPAL

def main(event):

#ENTRADA

try:

nombreArchivo = box.get()

archivoEnUnaLinea = leerArchivo(nombreArchivo)

try:

matrizInicial = archivoEnUnaLinea[0]+"\n"+archivoEnUnaLinea[1]+"\n"+archivoEnUnaLinea[2]+"\n"+archivoEnUnaLinea[3]+"\n"+archivoEnUnaLinea[4]+"\n"+archivoEnUnaLinea[5]+"\n"+archivoEnUnaLinea[6]+"\n"

except Exception as e:

matrizInicial = ""

#PROCESAMIENTO

if (validacion(archivoEnUnaLinea) == True):

#El archivo no contiene errores

#\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ASCII\_A\_NÚMERO\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_#

caracter = separarCaracteres(archivoEnUnaLinea)

numero = agruparCaracteres(caracter)

enteroMatriz = agruparNumeros(numero,caracter)

enteroNumero = reemplazar(enteroMatriz)

indiceOperacion = posicionOperador(enteroNumero)

tipoOperacion = operacion(enteroNumero)

resultadoOperacion = calculadora(enteroNumero,indiceOperacion,tipoOperacion)

#SALIDA

#\_\_\_\_\_\_\_\_\_\_\_\_\_\_OPERACIÓN\_EN\_TERMINAL\_\_\_\_\_\_\_\_\_\_\_\_#

print (linea+O+linea)

print matrizInicial

print (linea+R+linea)

#\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_NÚMERO\_A\_ASCII\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_#

numeroEnArreglo = reconvertir(resultadoOperacion)

matrizResultado = reordenar(numeroEnArreglo)

#\_\_\_\_\_\_\_\_\_\_\_\_OPERACIÓN\_EN\_INTERFAZ\_\_\_\_\_\_\_\_\_\_\_\_\_\_#

Operation.config(text=(linea+O+linea))

Matriz.config(text=matrizInicial)

Result.config(text=(linea+R+linea))

#\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_#

if (resultadoOperacion == "No se puede dividir por CERO"):

OUTPUT.config(text=resultadoOperacion)

print resultadoOperacion

else:

OUTPUT.config(text=matrizResultado)

print matrizResultado

else:

#Si la validación falló, existen dos tipos de errores

#1. Error de fila, hay filas de distintos largos

if (validacion(archivoEnUnaLinea) == ("\n" + fallo + "\n" + errorFila)):

#abre una ventana aparte que muestra el error y vacia todos los demas textos en pantalla

tkMessageBox.showwarning("Warning", (validacion(archivoEnUnaLinea)))

Operation.config(text="")

Result.config(text="")

OUTPUT.config(text="")

Matriz.config(text="")

#2. Error de caracter, en caso de que exista un caracter que no coincida con un punto o una equis

elif (validacion(archivoEnUnaLinea) == ("\n" + fallo + "\n" + errorCaracter)):

tkMessageBox.showwarning("Warning", (validacion(archivoEnUnaLinea)))

message.config(text="")

Operation.config(text="")

Result.config(text="")

OUTPUT.config(text="")

Matriz.config(text="")

print validacion(archivoEnUnaLinea)

print archivoEnUnaLinea

except SyntaxError:

pass

except NameError:

pass

except IOError:

print notFound

tkMessageBox.showerror("Error", notFound)

#Todas estas líneas son para que cuando arroje un error, todos los textos que están en la interfaz se vacíen y no muestre nada de un archivo anterior

message.config(text="")

Operation.config(text="")

Result.config(text="")

OUTPUT.config(text="")

Matriz.config(text="")

except IndexError:

#Igual que la anterior, pero cuando da IndexError significa que la matriz no coincidió con ninguna guardada en nuestros datos

print notMatch

#Muestra el mensaje al usuario de "notMatch" en las constantes

tkMessageBox.showwarning("Warning", fallo+"\n"+notMatch)

#Vacía todos los textos en pantalla

message.config(text="")

Operation.config(text="")

Result.config(text="")

OUTPUT.config(text="")

Matriz.config(text="")

#\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_#

#Esta función se ejecuta cuando el usuario presiona el botón de ayuda. Muestra la ventana ayuda

def instrucciones():

instrucciones1 = "Ingrese el nombre del archivo en la caja de texto. \nEl archivo debe contener reglas basicas en formato ASCII para ser leido\n"

instrucciones2 = "Se realizara el proceso matematico correspondiente. \nSe mostrara en pantalla el resultado obtenido"

tkMessageBox.showinfo("Instrucciones", instrucciones1+instrucciones2)

#\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_#

#ESTA FUNCIÓN ES UNA COPIA EXACTA AL MAIN PERO ESTA PROGRAMADA PARA QUE LA USE EL BUSCADOR DE ARCHIVOS GRAFICO

#La otra no se puede usar con un argumento dado, ya que contiene el argumento "event" que permite asignar teclas a diferentes funciones

def buscar(nombreArchivo):

try:

archivoEnUnaLinea = leerArchivo(nombreArchivo)

try:

matrizInicial = archivoEnUnaLinea[0]+"\n"+archivoEnUnaLinea[1]+"\n"+archivoEnUnaLinea[2]+"\n"+archivoEnUnaLinea[3]+"\n"+archivoEnUnaLinea[4]+"\n"+archivoEnUnaLinea[5]+"\n"+archivoEnUnaLinea[6]+"\n"

except Exception as e:

matrizInicial = ""

if (validacion(archivoEnUnaLinea) == True):

#El archivo no contiene errores

#\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ASCII\_A\_NÚMERO\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_#

caracter = separarCaracteres(archivoEnUnaLinea)

numero = agruparCaracteres(caracter)

enteroMatriz = agruparNumeros(numero,caracter)

enteroNumero = reemplazar(enteroMatriz)

indiceOperacion = posicionOperador(enteroNumero)

tipoOperacion = operacion(enteroNumero)

resultadoOperacion = calculadora(enteroNumero,indiceOperacion,tipoOperacion)

#\_\_\_\_\_\_\_\_\_\_\_\_\_\_OPERACIÓN\_EN\_TERMINAL\_\_\_\_\_\_\_\_\_\_\_\_#

print (linea+O+linea)

print matrizInicial

print (linea+R+linea)

#\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_NÚMERO\_A\_ASCII\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_#

numeroEnArreglo = reconvertir(resultadoOperacion)

matrizResultado = reordenar(numeroEnArreglo)

#\_\_\_\_\_\_\_\_\_\_\_\_OPERACIÓN\_EN\_INTERFAZ\_\_\_\_\_\_\_\_\_\_\_\_\_\_#

Operation.config(text=(linea+O+linea))

Matriz.config(text=matrizInicial)

Result.config(text=(linea+R+linea))

#\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_#

if (resultadoOperacion == "No se puede dividir por CERO"):

OUTPUT.config(text=resultadoOperacion)

print resultadoOperacion

else:

OUTPUT.config(text=matrizResultado)

print matrizResultado

else:

#Si la validación falló, existen dos tipos de errores

#1. Error de fila, hay filas de distintos largos

if (validacion(archivoEnUnaLinea) == ("\n" + fallo + "\n" + errorFila)):

#abre una ventana aparte que muestra el error y vacia todos los demas textos en pantalla

tkMessageBox.showwarning("Warning", (validacion(archivoEnUnaLinea)))

Operation.config(text="")

Result.config(text="")

OUTPUT.config(text="")

Matriz.config(text="")

#2. Error de caracter, en caso de que exista un caracter que no coincida con un punto o una equis

elif (validacion(archivoEnUnaLinea) == ("\n" + fallo + "\n" + errorCaracter)):

tkMessageBox.showwarning("Warning", (validacion(archivoEnUnaLinea)))

message.config(text="")

Operation.config(text="")

Result.config(text="")

OUTPUT.config(text="")

Matriz.config(text="")

print validacion(archivoEnUnaLinea)

print archivoEnUnaLinea

except SyntaxError:

pass

except NameError:

pass

except IOError:

print notFound

#Se imprime esta ventana mostrando el mensaje notFound de las constantes

tkMessageBox.showerror("Error", notFound)

#Todas estas líneas son para que cuando arroje un error, todos los textos que están en la interfaz se vacíen y no muestre nada de un archivo anterior

message.config(text="")

Operation.config(text="")

Result.config(text="")

OUTPUT.config(text="")

Matriz.config(text="")

except IndexError:

#Al igual que con la anterior, pero cuando da IndexError significa que la matriz no coincidió con ninguna guardada en nuestros datos

print notMatch

#Muestra el mensaje al usuario de notMatch en las constantes

tkMessageBox.showwarning("warning", fallo+"\n"+notMatch)

#Vacía todos los textos en pantalla

message.config(text="")

Operation.config(text="")

Result.config(text="")

OUTPUT.config(text="")

Matriz.config(text="")

def searchFile():

archivo=askopenfile()

buscar(archivo.name)

box.insert(0, archivo.name)

#\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_#

#INTERFAZ GRÁFICA

#Primero se crea la variable de mi ventana principal llamada root

root = Tk()

#Se le agrega un título a la ventana

root.title("Calculadora ASCII")

#Se le asigna la tecla ENTER para que ejecute la función main y haga el papel del botón buscar

root.bind("<Return>", main)

#En Sistemas Operativos de Linux no permite cambiar el ícono de la ventana, por lo que podemos saber de qué S.O. se trata

#Al intentar abrir el programa en linux se le mostraráun mensaje explicándole el fallo y recomendando abrirlo en Windows

try:

ubuntu = False

root.iconbitmap('icon.ico')

except Exception as e:

ubuntu = True

tkMessageBox.showinfo("Linux detectado", "Para una mejor experiencia ejecutar el programa en Windows")

#Fuentes

#Para Ubuntu u otra distribución de Linux debes descargar una fuente monoespaciada, en este caso se usa Anonymous Pro para Ubuntu

#Se descarga desde: http://www.marksimonson.com/fonts/view/anonymous-pro

#Se copia con permisos root en la carpeta /usr/share/fonts

if ubuntu:

font = tkFont.Font(family="AnonymousPro", size=12)

else:

font = tkFont.Font(family="FixedSys", size=12)

#Calibri light para los textos

calibri = tkFont.Font(family="Calibri Light", size=12)

leftFrame = Frame(root)

leftFrame.pack(pady = 50, padx = 50)

#Aquí se guarda la imagen del logo de la universidad en la variable logo

logo = PhotoImage(file="logo.gif")

usach = Label(root, image=logo)

usach.pack(side=TOP)

#Texto que le dice al usuario que ingrese el nombre del archivo

text = Label(leftFrame, text="Ingresar el nombre del archivo:")

text.grid(row=0,sticky=W+E+N+S)

#Caja de entrada de texto para que el usuario inserte el nombre del archivo

box = Entry(leftFrame, textvariable="")

box.grid(column=0,row=1,sticky=W+E+N+S)

#Bóton para buscar que esta mapeado para que cuando haga se le haga click ejecute la función main

submitFile = Button(leftFrame, text="Buscar", fg="black", command=searchFile)

submitFile.grid(column=0,row=2,sticky=W+E+N+S)

#Botón de ayuda para que cuando se le haga click ejecute la función instrucciones

HELP = Button(leftFrame, text="Ayuda", fg="black", command = instrucciones)

HELP.grid(column=0,row=3,sticky=W+E+N+S)

#Mensaje aleatorio que se usa como resguardo en caso de que alguno falle, o no se muestre

message = Label(leftFrame, text="", font=calibri)

message.grid(row=5,sticky=W+E+N+S)

#Operación título

Operation = Label(leftFrame, text="")

Operation.grid(row=4,sticky=W+E+N+S)

#Matriz de la operación

Matriz = Label(leftFrame, text="", font=font)

Matriz.grid(row=5,sticky=W+E+N+S)

#Resultado título

Result = Label(leftFrame, text="")

Result.grid(row=6,sticky=W+E+N+S)

#Matriz final en la salida

OUTPUT = Label(leftFrame, text="", font=font)

OUTPUT.grid(row=7,sticky=W+E+N+S)

#Status bar con la version del proyecto

status = Label(root, text="Version 2.5.2 - Calculadora ASCII", bd=1, relief=SUNKEN, anchor=W)

status.pack(side=BOTTOM, fill=X)

root.mainloop()

#\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_#