# ESTRUCTURA DE PROGRAMA

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

#

# Objetivo: Operar dos números en formato ASCII y entregar su resultado

# en el mismo formato

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

# Fecha: 24-10-2016

# Versión: 0.8

# BLOQUE DE DEFINICIÓN

# DEFINICIÓN DE CONSTANTES

# NUM0 = [[‘x’,’x’,’x’,’x’,’x’],…,[[‘x’,’x’,’x’,’x’,’x’]] representa NUM0

# Todos los números y signos siguen este mismo formato, y se decidió no

# incluirlo en el código por razones de orden

NUM0 = []

NUM1 = []

NUM2 = []

NUM3 = []

NUM4 = []

NUM5 = []

NUM6 = []

NUM7 = []

NUM8 = []

NUM9 = []

SMAS = []

SMEN = []

SMUL = []

SDIV = []

PUNT = []

# IMPORTACIÓN DE FUNCIONES

# no se utilizan

# DEFINICIÓN DE FUNCIONES

# Función que crea una lista con cada línea del archivo de lectura

# Entrada: No hay

# Salida: Una lista de 7 elementos, cada elemento es una fila distinta

# del archivo de lectura

**def** leerArchivo**():**

# Abre un archivo

# Lee una línea y la asigna a un elemento de una lista (se repite

# siete veces este proceso)

# Se debe terminar esta función

archivo = open("operacion.txt", "r")

**for** i **in** range(7):

fila.append(archivo.readline())

# Función que separa todos los caracteres de cada elemento de una lista

# Entrada: No hay

# Salida: Una lista de 7 elementos, cada elemento es una lista compuesta

# por cada carácter de la lista de entrada

**def** validar**():**

# Cuando se cumplen varias condiciones del archivo entonces se

# el resto del programa

# Las condiciones deberían ser que no existan caracteres distintos a

# ‘x’ y ‘.’, y que los números estén completos, si no se cumplen

# entonces se vuelve a pedir que se abra el archivo

# Se debe terminar esta función

**while** True:

**if** <condicion1> **and** <condicion2>:

break

**else**:

fila = leerArchivo()

# Función que separa todos los caracteres de cada elemento de una lista

# Entrada: Una lista de 7 elementos

# Salida: Una lista de 7 elementos, cada elemento es una lista compuesta

# por cada carácter de la lista de entrada

**def** separarCaracter**(**fila**):**

caracter = []

**for** i **in** range(7):

caracter.append([])

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

**return** character

# Función que agrupa en listas de a cinco elementos dentro de una lista # de 7 elementos y elimina el punto que separa a los números ASCII

# Entrada: Una lista compuesta por 7 listas

# Salida: Una lista que dentro de cada elemento hay varias listas, y que

# cada una de estas contiene 5 caracteres que están dentro de una misma

# fila en el archivo

**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

# Función que deja dentro de un mismo elemento todas las listas que

# representan a un número (o al operador) dentro de la operación

# matemática que se va a realizar

# Entrada: Una lista

# Salida: Una lista que sus elementos son números (u operadores) ASCII en

# el mismo formato que las constantes del código

**def** agruparNumeros**(**numero**):**

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

# Función que reemplaza los números ASCII por números String

# Entrada: Una lista con números ASCII

# Salida: Una lista con números String

**def** reemplazar**(**enteroMatriz**):**

enteroNumero = []

**for** i **in** range(len(enteroMatriz)):

**if** enteroMatriz[i] == NUM1:

enteroNumero.append('1')

**elif** enteroMatriz[i] == NUM2:

enteroNumero.append('2')

**elif** enteroMatriz[i] == NUM3:

enteroNumero.append('3')

**elif** enteroMatriz[i] == NUM4:

enteroNumero.append('4')

**elif** enteroMatriz[i] == NUM5:

enteroNumero.append('5')

**elif** enteroMatriz[i] == NUM6:

enteroNumero.append('6')

**elif** enteroMatriz[i] == NUM7:

enteroNumero.append('7')

**elif** enteroMatriz[i] == NUM8:

enteroNumero.append('8')

**elif** enteroMatriz[i] == NUM9:

enteroNumero.append('9')

**elif** enteroMatriz[i] == SMAS:

enteroNumero.append('+')

**elif** enteroMatriz[i] == SMEN:

enteroNumero.append('-')

**elif** enteroMatriz[i] == SMUL:

enteroNumero.append('\*')

**elif** enteroMatriz[i] == SDIV:

enteroNumero.append('/')

**else**:

enteroNumero.append('0')

**return** enteroNumero

# Función que encuentra la posición del operador matemático

# Entrada: Una lista de String

# Salida: Un valor que representa la posición del operador en la lista de

# números String

**def** posicionOperador**(**enteroNumero**):**

indiceOperacion = 0

**if** '+' **in** enteroNumero:

indiceOperacion = enteroNumero.index('+')

**elif** '-' **in** enteroNumero:

indiceOperacion = enteroNumero.index('-')

**elif** '\*' **in** enteroNumero:

indiceOperacion = enteroNumero.index('\*')

**else**:

indiceOperacion = enteroNumero.index('/')

**return** indiceOperacion

# Función que determina que operación se va a realizar

# Entrada: Una lista de String

# Salida: Un string que puede ser ‘+’, ‘-‘, ‘\*’, o ‘/’

**def** operacion**(**enteroNumero**):**

tipoOperacion = ''

**if** '+' **in** enteroNumero:

tipoOperacion = '+'

**elif** '-' **in** enteroNumero:

tipoOperacion = '-'

**elif** '\*' **in** enteroNumero:

tipoOperacion = '\*'

**else**:

tipoOperacion = '/'

**return** tipoOperacion

# Función que crea convierte los números a enteros y luego los opera

# Entrada: Una lista de String, la posición del operador dentro de la

# lista, y el tipo de operación a realizar

# Salida: Un valor entero o decimal

**def** operar**(**enteroNumero, indiceOperacion, tipoOperacion**):**

numeroSt1 = ''

numeroSt2 = ''

**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)

**if** tipoOperacion == '+':

resultado = primero + segundo

**elif** tipoOperacion == '-':

resultado = primero - segundo

**elif** tipoOperacion == '\*':

resultado = primero \* segundo

**else**:

resultado = float(primero) / float(segundo)

**return** resultado

# Función que transforma el resultado a número ASCII

# Entrada: Una valor entero o decimal

# Salida: Una lista donde cada elemento es una fila diferente del número # ASCII resultante

**def** resultadoASCII**(**resultado**):**

# El resultado lo transforma a String y lo separa en sus caracteres

# Cada carácter va a ser un elemento de una lista

# Se crea una lista nueva donde se reemplaza cada carácter por un

# número ASCII

# Se reorganiza con tal de que que cada elemento sea una línea del

# resultado en formato ASCII

# Aún no se realiza esta función

# BLOQUE PRINCIPAL

# ENTRADA

fila = leerArchivo()

# PROCESO

validar()

caracter = separarCaracteres(fila)

numero = agruparCaracteres(caracter)

enteroMatriz = agruparNumeros(numero)

enteroNumero = reemplazar(enteroMatriz)

indiceOperacion = posicionOperador(enteroNumero)

tipoOperacion = operacion(enteroNumero)

resultado = operar(enteroNumero, indiceOperacion, tipoOperacion)

resultadoFinal = resultadoASCII(resultado)

# SALIDA

for linea in resultadoFinal:

print linea