UNIVERSIDAD DE SANTIAGO DE CHILE
FACULTAD DE INGENIERÍA
Fundamentos de Computación y Programación

PRODUCTO N°2
Calculadora ASCII

Grupo N°3

Sección : G-2
Profesor de Laboratorio : Irene Zuccar
Profesor de Teoría : Alejandro Cisterna
Integrantes : Danny Aguilar
Guillermo Guzmán
Diego Martínez
Itzvan Pastén
Sebastián Salazar

Santiago
Diciembre de 2016

CONTENIDO

INTRODUCCIÓN

Este informe es una recopilación de todo el trabajo realizado a lo largo del semestre en el laboratorio de Fundamentos de computación y programación, el proyecto consistió en crear un programa en Python que permita realizar operaciones aritméticas con números enteros, que son entregados inicialmente como una representación de caracteres ASCII en un archivo de texto y cuyo resultado deber ser mostrado por pantalla usando el mismo formato. Enfrentar este proyecto resultó muy motivador ya que nos proporcionó la oportunidad de poner en práctica el conocimiento teórico que se presentó a lo largo del curso, y con esto resolver un problema concreto, además nos permitió descubrir la importancia de comprender correctamente un problema y tener la capacidad de realizar un proceso de abstracción, que oriente los pasos a seguir, en busca de una solución. Este proceso nos permitió crear las funciones necesarias para realizar un correcto procesamiento de la información y finalmente, en su conjunto, generar la solución a nuestro problema, es decir, nuestro programa. Este trabajo también nos ha motivado a buscar más información, que no se presenta en el curso con el objetivo de resolver nuestro problema y construir bases sólidas que nos permitan solucionar problemas más complejos en el futuro.

Este objetivo de este informe es mostrar con el mayor detalle posible el trabajo realizado a lo largo del semestre, para que, a través de este sea posible observar el desarrollo que se tuvo como grupo en el proceso de obtención del producto final. La primera parte de este documento y con el objetivo de mostrar las dinámicas de trabajo que se tuvieron durante el semestre, se detallará cada una de las sesiones en las que se trabajó como grupo. También se incluirá el material de apoyo utilizado en cada una de las presentaciones y el documento del producto N°1 original, lo que permitirá observar los cambios realizados durante el semestre generados por el *feedback* recibido principalmente por nuestros evaluadores y también compañeros. La última parte de este informe estará enfocada en mostrar el producto final, se iniciará presentando la abstracción de proceso final y corregida, seguido del código fuente de nuestra versión final del programa, que permitirá mostrar las funciones que se crearon a partir de nuestro diagrama de abstracción, además se adjuntara un manual que permitirá que un usuario pueda ocupar el programa que hemos creado “calculadora ASCII”.

FICHA DE INSCRIPCIÓN DEL PROYECTO

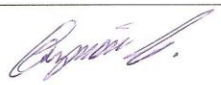
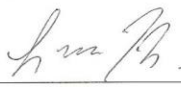
Universidad de Santiago de Chile *Fundamentos de Computación y Programación*

Reglamento de laboratorio

1. En caso de detectar alguna falta a la ética o a la honestidad por parte de alguno de los grupos de laboratorio el grupo reprueba directamente el laboratorio del curso con la calificación mínima, dicha calificación aplica al promedio final. Se consideran, como faltas a la honestidad actos tales como:
   - Facilitar parte o la totalidad de una entrega a otro grupo, de la misma u otra sección.
   - Presentar entregas con intervención de terceros.
   - Presentar entregas que no son de la autoría única de los integrantes del grupo.
   - Presentar entregas con contenidos textuales de fuentes no referenciadas.
2. En orden de aplicar dicha sanción, un comité de tres profesores, designados por la coordinación revisarán el caso y validarán la decisión.
3. En caso de que un estudiante cometa una falta, la sanción se aplicará al grupo en su completitud.
4. Cualquier situación no prevista será dirimida por la coordinación.

Firma de Conformidad y Aprobación de cada miembro del equipo

Mediante el presente documento, los firmantes declaran tomar conocimiento de las reglas del laboratorio de Fundamentos de Computación y Programación y las sanciones que se aplicarán en caso de incurrir en el incumplimiento de estas.

NOMBRE, APELLIDOS Y RUT	FIRMA	V°B° Profesor
Responsable DANNY ANDRES AGUILAR RIVAS 19.434.887-5		
Asistente del Responsable Guillermo Guzmán Almonacid 19.219.562-4		
Secretario Diego Martínez Beltrami 17.516.451-0		
Responsable del material ITZUAN PASTEN Valencia 16341364-7	ITZUAN PASTEN	
Responsable de las presentaciones SEBASTIÁN SANTANA 22.805.902-1		

Segundo semestre de 2016

MINUTAS DE CADA SESIÓN

Lunes 29 de agosto del 2016.

Se creó un grupo de WhatsApp para facilitar la comunicación y coordinación de todos los integrantes del grupo.

Se creó una cuenta compartida en la plataforma GitHub, con la intención de que todos los integrantes del grupo puedan ir aportando ideas nuevas e ir subiendo, editando, creando y comentando de forma colaborativa los avances del proyecto.

Lunes 05 de septiembre del 2016.

Se definen los roles de cada integrante del grupo y se procede a llenarlos en una hoja, en la cual Danny Aguilar se compromete a ser el responsable del grupo, Diego Martínez el secretario, Sebastián Salazar al desarrollo de las presentaciones, Iztvan Pasten, a ser responsable del material, y Guillermo Guzmán se compromete a ser el asistente del responsable, ocupando su lugar en el caso que amerite su reemplazo.

Se generan ideas de como plantear la lógica del proyecto.

Se da la tarea grupal que cada uno investigue sobre la programación orientada a procesamiento de textos en Python y a la lógica matemática de Python con el fin generar una orientación en el desarrollo del proyecto.

Se propone generar cada uno una idea de cómo abordar el proyecto para discutirlas como grupo la clase siguiente de laboratorio.

Lunes 12 de septiembre del 2016.

Se discute como grupo las diferentes propuestas entregadas por los integrantes y se comienza a avanzar en el proyecto con la idea más aceptada.

Jueves 22 de septiembre del 2016.

Se sube a la plataforma GitHub la primera versión del avance proyecto, con las funciones principales ya elaboradas, con la intención de que cada integrante del grupo pueda editarlo o agregar notas sobre posibles mejoras del avanza inicial del código.

Lunes 26 de septiembre del 2016.

Se trabaja en las funcionalidades desarrolladas hasta la fecha, y en la elaboración de un diagrama de abstracción que englobe las funcionalidades realizadas y que faltan por realizar.

Lunes 03 de octubre del 2016.

Se expone el primer avance del proyecto en la clase de laboratorio.

Lunes 10 de octubre del 2016.

Se sube un segundo avance del proyecto, actualizando los avances logrados hasta la fecha, con la implementación de mejoras en las funcionalidades creadas y en la implementación de nuevas funciones.

Lunes 17 de octubre del 2016.

Se discuten ideas sobre la implementación de un módulo que permita manipular más fácilmente los archivos de textos.

Se dan tareas individuales sobre el informe del proyecto que debe elaborarse.

Se organiza una junta grupal para el día martes 19 de octubre, para avanzar en el primer informe del proyecto.

Miércoles 19 de octubre del 2016

Se junta el grupo en la biblioteca central de la universidad de Santiago, en la cual se avanzó sobre el primer informe y presentación del proyecto.

Se desarrolla el diagrama de abstracción del proyecto.

Se genera una nueva instancia de discusión de ideas, para seguir avanzando en lo que falta del código.

Sábado 22 de octubre del 2016

Se organiza una junta en la biblioteca de la universidad de Santiago, donde se finalizan las funciones restantes, teniendo practicante el proyecto andando, con los requerimientos básicos que este requería.

Lunes 24 de octubre del 2016.

Se analiza el código creado y posibles mejoras.

Se genera una instancia de discusión de los posibles bugs o errores de entrada que pueden existir al momento de ingresar el archivo requerido.

Se crea una función que analice errores de caracteres dentro del archivo de entrada.

Miércoles 26 de octubre del 2016.

Se crean más funciones ligadas a los posibles errores de entradas y bugs que presenta el programa.

Se prepara la presentación del avance del proyecto.

Lunes 07 de noviembre del 2016.

Se presenta el avance del proyecto en laboratorio sin nota.

Viernes 11 de noviembre del 2016.

Se presenta el avance del proyecto al curso y profesor de catedra.

Lunes 14 de noviembre del 2016.

Se discute como grupo la elaboración de una pequeña interfaz que facilite la ejecución del programa.

Lunes 21 de noviembre del 2016.

Se crea la interfaz con ideas propuestas por integrantes del grupo.

Se hace una prueba de todo lo que llevamos, incluyendo todos los posibles errores, tanto en la entrada del archivo, como en el contenido del archivo, y se analiza cómo funciona el programa.

Lunes 28 de noviembre del 2016.

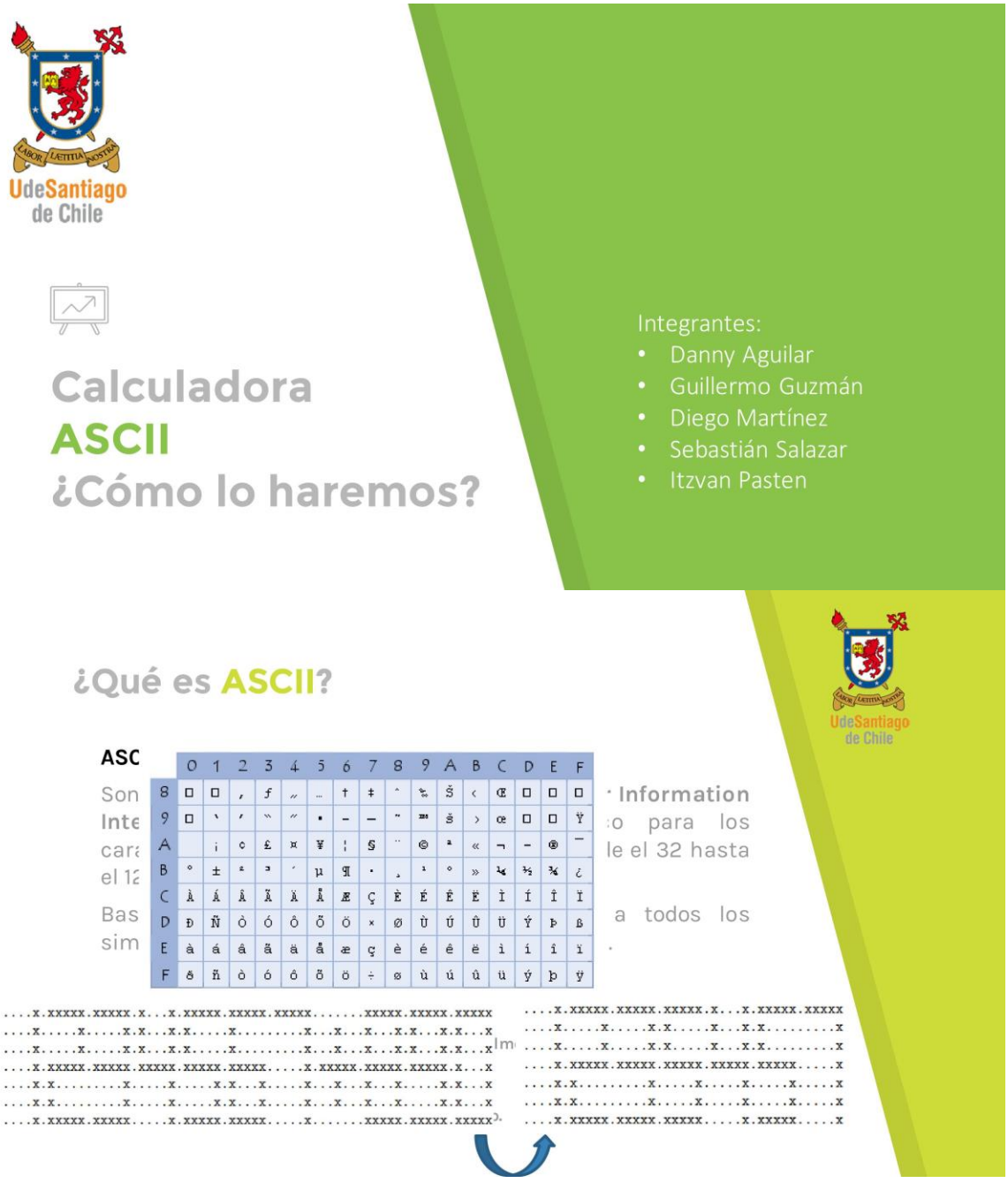
Se dan tareas individuales para el informe final y la elaboración final de la presentación del proyecto.

Miércoles 30 de noviembre del 2016.

Se organiza una junta grupal en la biblioteca de la universidad, para la finalización del informe del proyecto final.

PRESENTACIONES

Presentación de Anteproyecto



PARA COMENZAR!

¿Cómo lo haremos?

1.

ETAPA INICIAL

...

En esta etapa comenzaremos a anotar cuales herramientas necesitaremos en un futuro, ya sea un editor de código, una plataforma para guardar, etc.

LAS QUE CREEMOS SON **HERRAMIENTAS** NECESARIAS

- Github
- Editor de python (a elección)
- Chat Grupal en WhatsApp

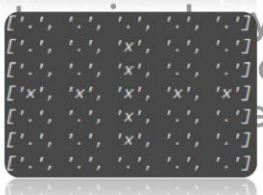
La razón por la cual las elegimos es por que creemos que nos facilitan mucho nuestro trabajo a la hora de controlar versiones y / o comunicarse con los demas.

2.

Plantearse una solución

...

Para esta etapa tuvimos que planear varias "posibles" soluciones para poner en marcha, dentro de ellas, elegimos la mas factible y en caso de no funcionar podremos continuar con las demás.



manera logra
dora e intentar que nuestro
evuelva las matrices ya

UdeSantiago de Chile

NUESTRO **PLAN** A SEGUIR

- Modelar cada uno de los caracteres pedidos
- Realizar una lógica de calculadora
- Lograr que nuestra calculadora retorne las matrices ya hechas
- Insertar las matrices en un archivo txt
- Encontrar la manera de que nuestro programa lea archivos txt y reconozca matrices
- Finalmente que transforme la matriz de entrada a algo que nuestro programa entienda.

UdeSantiago de Chile

NUESTROS **AVANCES**

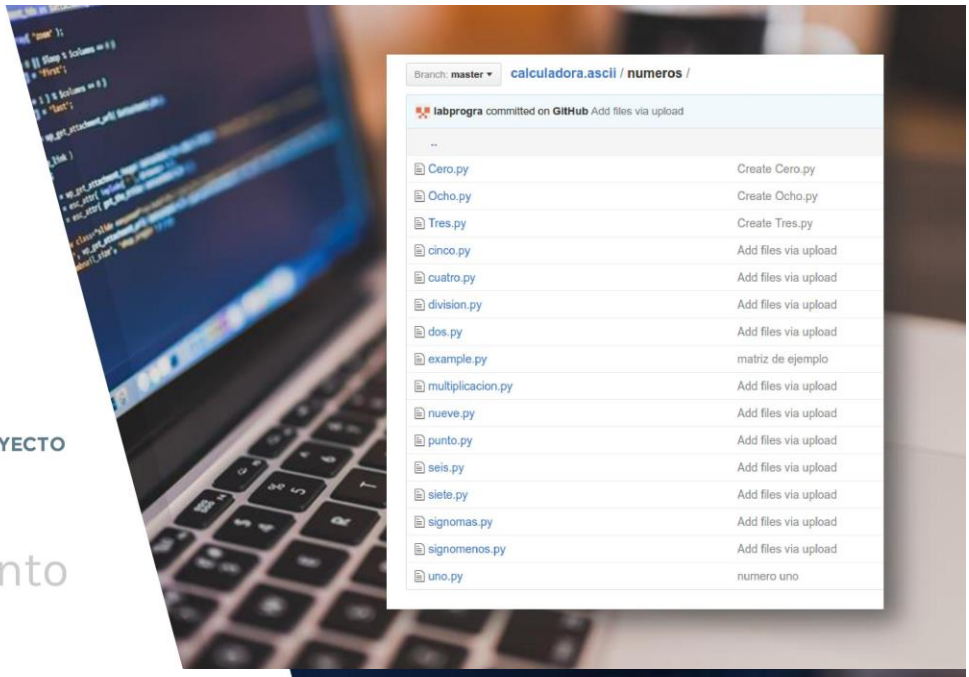
UdeSantiago de Chile

ELEGIMOS **GITHUB** DEBIDO A QUE...

Es sencillo
Es un entorno mucho mas sencillo que otras plataformas para editar archivos de codigo.

Es seguro
Gracias a su seguridad podremos tener la certeza de que siempre con una conexion a internet tendremos nuestro trabajo siempre actualizado en caso de cualquier problema.

UdeSantiago de Chile



NUESTRO **PROYECTO** AVANZANDO

Por el momento

UdeSantiago de Chile

COMENZAMOS CON UNA MATRIZ BASE

Esta matriz se fue editando para dar con la forma a todos los símbolos que necesitamos en matrices de 7x5 para los números y 5x5 para los simbolos.

1ó2 meses

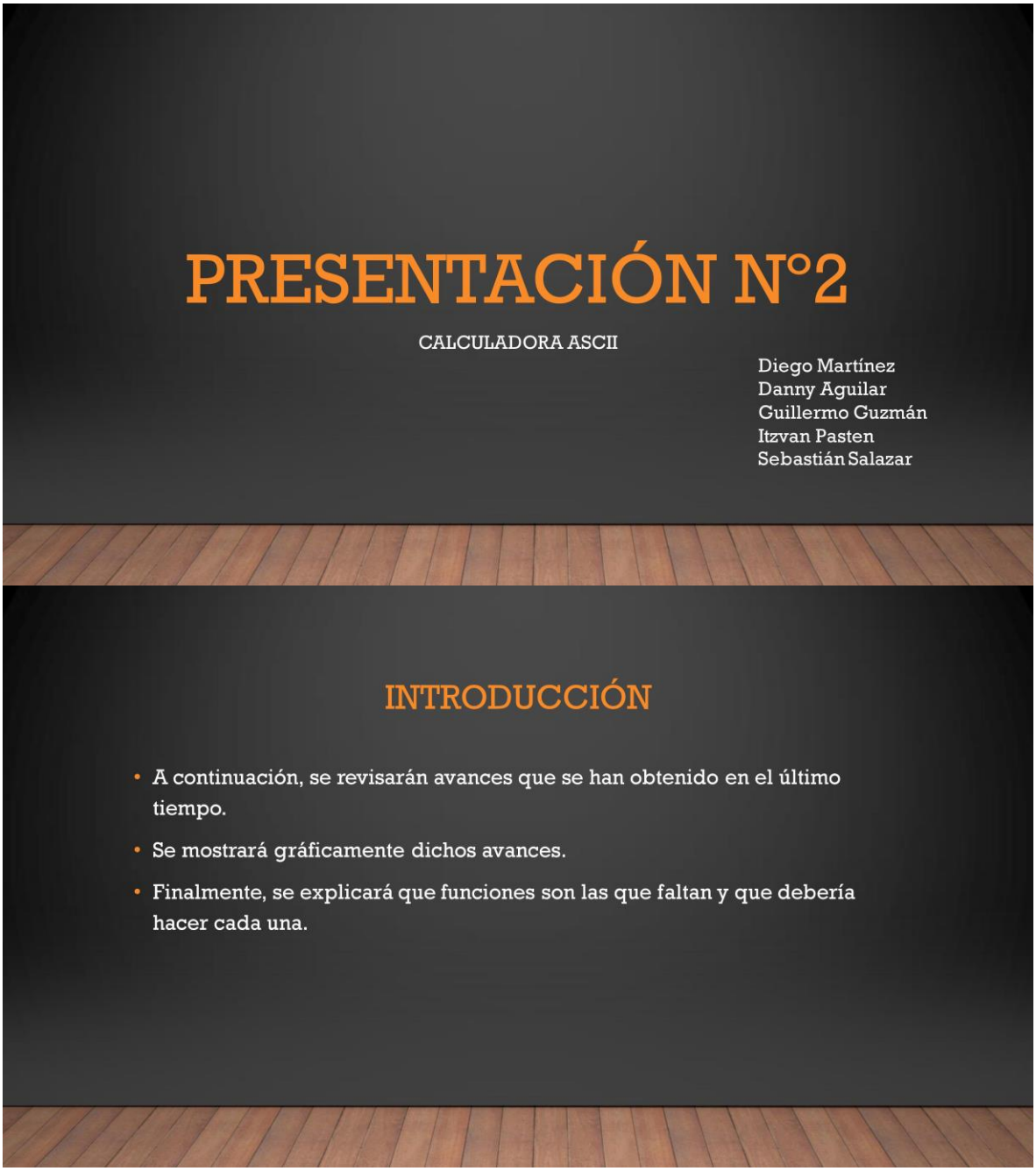
Tiempo estimado para terminar una primera versión.

GRACIAS!

Alguna pregunta?

No duden el levantar la mano

Presentación n° 1[1]



[1] Esta presentación contenía animaciones que no pudieron ser replicadas en el informe, para las diapositivas 8-12

INTEGRANTES

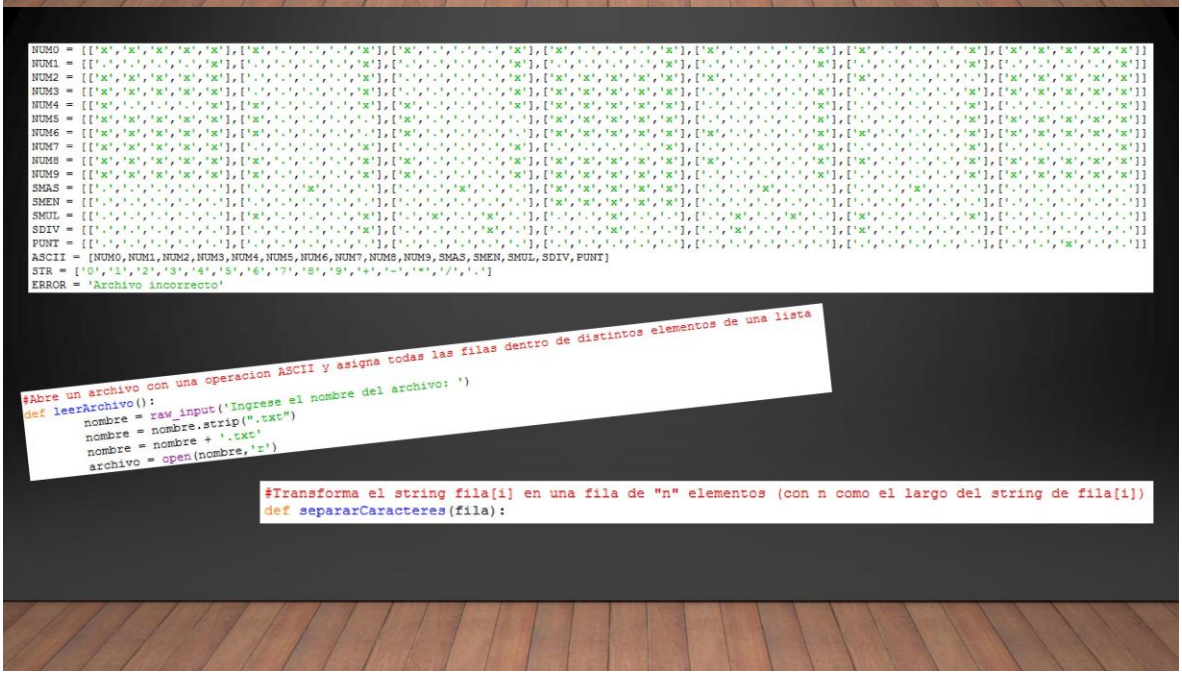
- Danny Aguilar – Responsable de Grupo
- Guillermo Guzmán – Asistente del Responsable
- Itzvan Pasten – Encargado de Materiales
- Diego Martínez – Secretario
- Sebastián Salazar – Encargado de Presentación

CALCULADORA ASCII

- Nuestro proyecto tiene como objetivo la validación y manipulación de archivos .txt, en formato de matrices 7x5 que contengan 2 números y una operación a realizar.
- La salida de nuestro programa, genera un archivo con el mismo formato, que contiene el resultado de la operación antes mencionada

FUNCIONALIDADES (1)

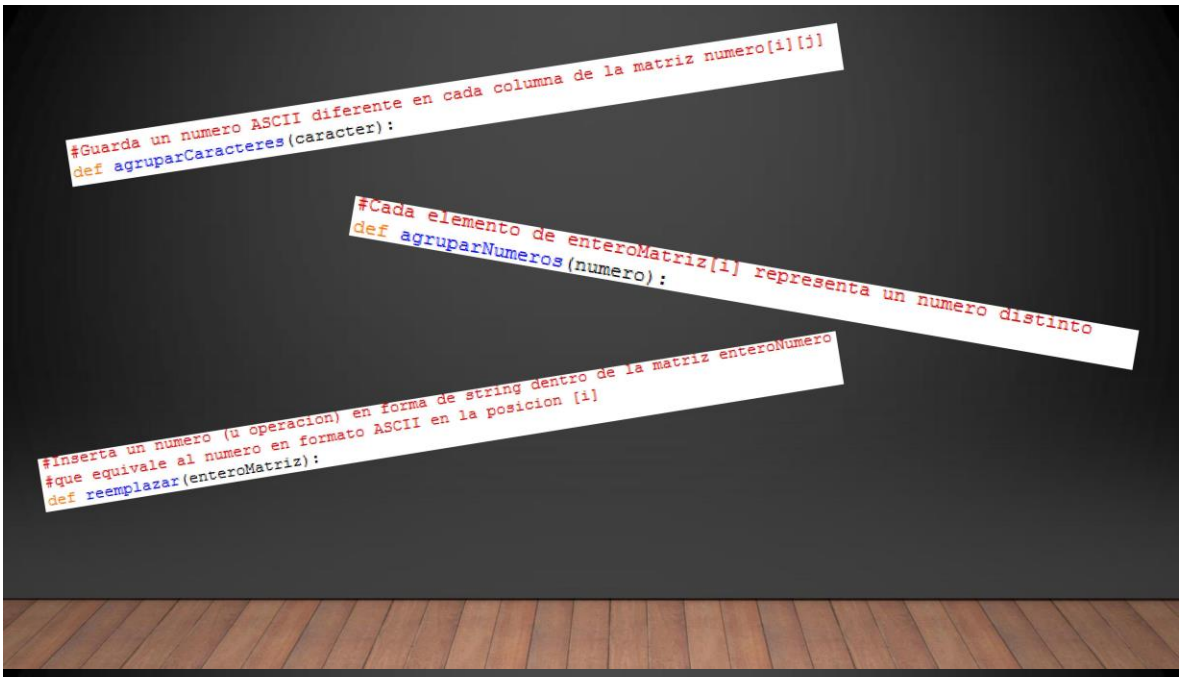
1. Creación de constantes
2. Solicitar al usuario ingresar un archivo de texto
3. Leer el archivo
4. Asignar cada línea del archivo a un elemento distinto de una lista
5. Verificar su contenido
6. Separar los caracteres del String de cada fila, resultando una lista de 7 listas
7. Agrupar caracteres, eliminando el punto
8. Reordenar caracteres

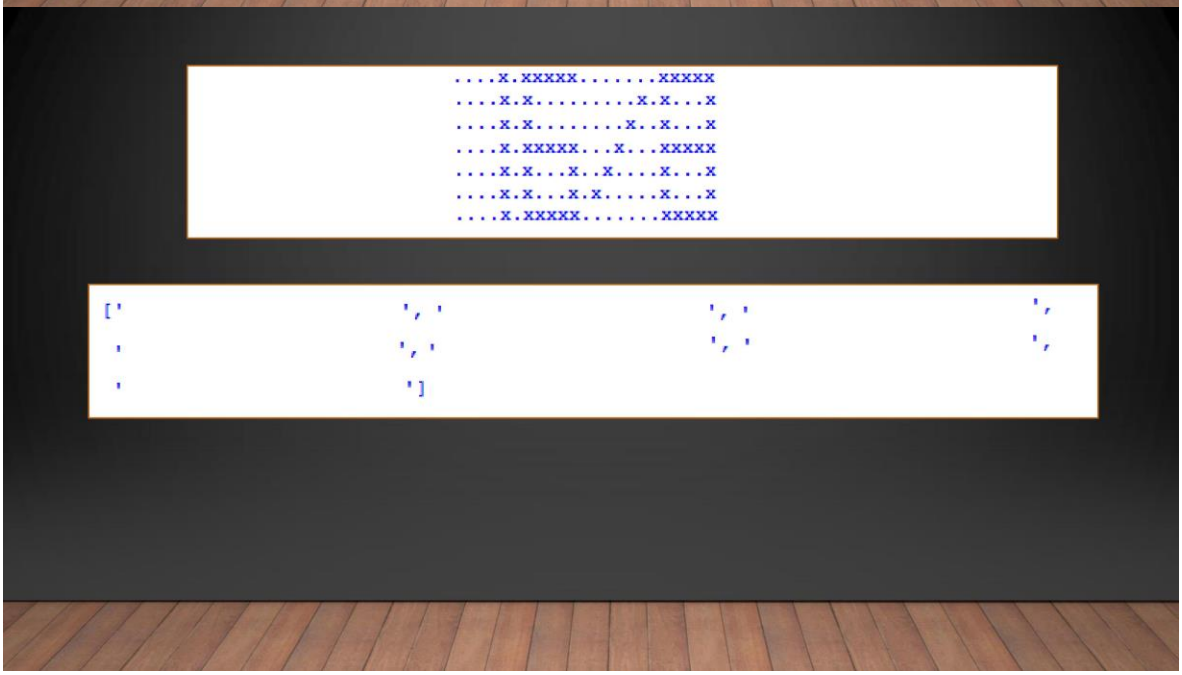


```
NUM0 = [['x','x','x','x','x'],['x',' ',' ',' ','x'],['x',' ',' ',' ','x'],['x',' ',' ',' ','x'],['x',' ',' ',' ','x'],['x','x','x','x','x']]
NUM1 = [[' ',' ',' ',' ','x'],[' ',' ',' ',' ','x'],[' ',' ',' ',' ','x'],[' ',' ',' ',' ','x'],[' ',' ',' ',' ','x'],[' ',' ',' ',' ','x']]
NUM2 = [['x','x','x','x','x'],[' ',' ',' ',' ','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']]
NUM4 = [['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']]
NUM6 = [['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']]
NUM8 = [['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']]
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',' ',' ',' ',' '],[' ',' ',' ',' ',' ']]
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','+','-','*','/','.']
ERROR = 'Archivo incorrecto'
```

```
#Abre un archivo con una operacion ASCII y asigna todas las filas dentro de distintos elementos de una lista
def leerArchivo():
    nombre = raw_input('Ingrese el nombre del archivo: ')
    nombre = nombre.strip(".txt")
    nombre = nombre + '.txt'
    archivo = open(nombre,'r')
```

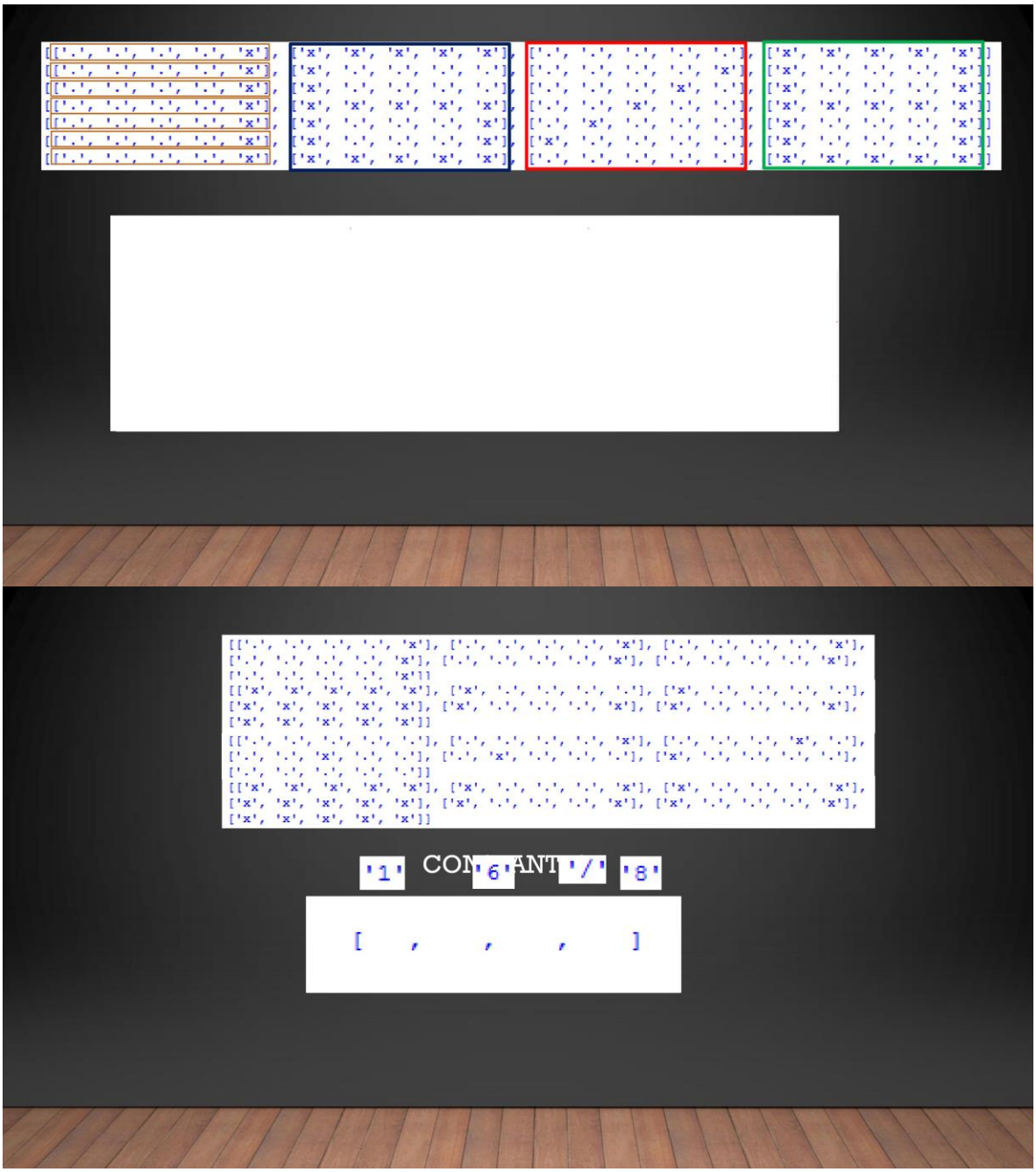
```
#Transforma el string fila[i] en una fila de "n" elementos (con n como el largo del string de fila[i])
def separarCaracteres(fila):
```


#Guarda un numero ASCII diferente en cada columna de la matriz numero[i][j]
def agruparCaracteres(caracter):
#Cada elemento de enteroMatriz[i] representa un numero distinto
def agruparNumeros(numero):
#inserta un numero (u operacion) en forma de string dentro de la matriz enteroNumero
#que equivale al numero en formato ASCII en la posicion [i]
def reemplazar(enteroMatriz):


....x.xxxxx.......xxxxx
....x.x.........x.x...x
....x.x.........x..x...x
....x.xxxxx...x...xxxxx
....x.x...x..x....x...x
....x.x...x.x.....x...x
....x.xxxxx.......xxxxx
['
', '
', '
',
'
', '
', '
',
'
']


['....x.xxxxx.......xxxxx '
'....x.xxxxx.......xxxxx '
['.', '.', '.', '.', 'x', '.', 'x', 'x', 'x', 'x', 'x', '.', '.', '.', '.', '.', '.', '.', 'x', 'x', 'x', 'x', 'x']


[['.', '.', '.', '.', 'x'],


'1' CONSTANT '/' '8'
[, , ,]

FUNCIONALIDADES (2)

9. Comparar con las constantes
10. Agregar el número u operador dentro de otra lista, en la misma posición donde se encontraba el número ASCII
11. Encontrar la posición del operador matemático dentro de la lista
12. Formar los dos números (como strings)
13. Transformar los números a enteros
14. Realizar la operación matemática

```
#Ubica la posicion del operador
def posicionOperador(enteroNumero):
```

```
#Indica que tipo de operacion se va a realizar
def operacion(enteroNumero):
```

```
#Transforma los numeros a enteros y los opera
def operar(enteroNumero, indiceOperacion, tipoOperacion):
```

FUNCIONALIDADES (3)**

15. Transformar el resultado a String*
16. Separar cada caracter como un elemento de una lista*
17. Reemplazar cada elemento por su número ASCII correlativo*
18. Agrupar por filas, agregando un punto entre cada número*
19. Entregar el resultado de la operación por pantalla en formato ASCII *
20. En caso de algún error, solicitar al usuario que vuelva a ingresar el archivo de texto

MEJORAS

- El programa lee el archivo de texto y transforma los números a enteros
- El programa resuelve las 4 operaciones matemáticas requeridas
- Se creó un mecanismo de validación donde el programa revisa que el formato de las matrices sea el correcto

CONCLUSIONES

- Ya está listo lo relacionado con las entradas de la calculadora
- Se espera finalizar pronto lo que resta del programa dado que son pocas las funciones que faltan por realizar
- La salida es el proceso inverso al realizado para operar lo números

PRODUCTO N° 1

CONTENIDO

INTRODUCCIÓN

Al enfrentar este proyecto el elemento que más nos ha motivado, es el tener la oportunidad de poner en práctica el conocimiento teórico que se nos ha ido presentado a lo largo del curso, y con estas herramientas acercarnos a resolver un problema concreto. El problema elegido (calculadora ASCII) inicialmente podría parecer un problema sencillo, eso nos pareció al comienzo del curso cuando elegimos este tema, pensamos que sería fácil realizar un programa que ejecute las operaciones aritméticas básicas, que hasta la calculadora más básica (incluso la del celular) puede realizar, pero mientras avanzaba el curso e íbamos adquiriendo más conocimiento, empezamos a descubrir la cantidad de detalles que se deben considerar al realizar un programa, partiendo por la dificultad que presenta el transformar la información de una entrada (en nuestro caso un archivo de texto con código ASCII) a información que pueda ser procesada, pasando por la construcción de funciones que permitan un correcto procesamiento de la información, hasta la generación de una salida entendible para el usuario, todas están consideraciones que inicialmente no estaban presentes, nos han motivado a buscar más información que nos permitan avanzar a resolver nuestro problema y sean la base para que en el futuro seamos capaces de resolver problemas más complejos.

El objetivo de este informe es el de presentar la estructura del programa que se ha construido a lo largo del semestre se detallaran las distintas funciones que se han creado y se presentará la estructura del programa separada en entrada, procesamiento y salidas

PLANTEAMIENTO DEL PROBLEMA

Nuestro proyecto es “Calculadora ASCII” consiste en crear un programa en Python que permita realizar operaciones aritméticas con números enteros, estos números se entregaran inicialmente representados por matrices con caracteres ASCII en un archivo de texto y el resultado deberá ser mostrado por pantalla también usando matrices con caracteres ASCII.

Hemos realizado algunos cambios en el planteamiento que presentamos en la presentación anterior, uno de ellos es que decidimos no generar un archivo texto con las matrices, ya que no es de utilidad y tampoco es algo que se nos exija. Los demás cambios tienen que ver principalmente con el plan de acción, específicamente el orden y el detalle de nuestro plan de acción, esto se debe a que en la primera entrega no teníamos las herramientas necesarias para crear el programa, por lo que agrupamos demasiados procesos que son casi la totalidad del programa solo en dos, estos fueron:

1. Encontrar la manera de que nuestro programa lea archivos .txt y reconozca matrices
2. Transformar la matriz de entrada a algo que nuestro programa entienda.

Como hemos avanzado en este proyecto, gracias a nuestra gran cantidad de herramientas de programación y análisis, para ésta entrega pudimos detallar estos procesos y distinguir por lo menos 15 que serán presentados a lo largo de este informe.

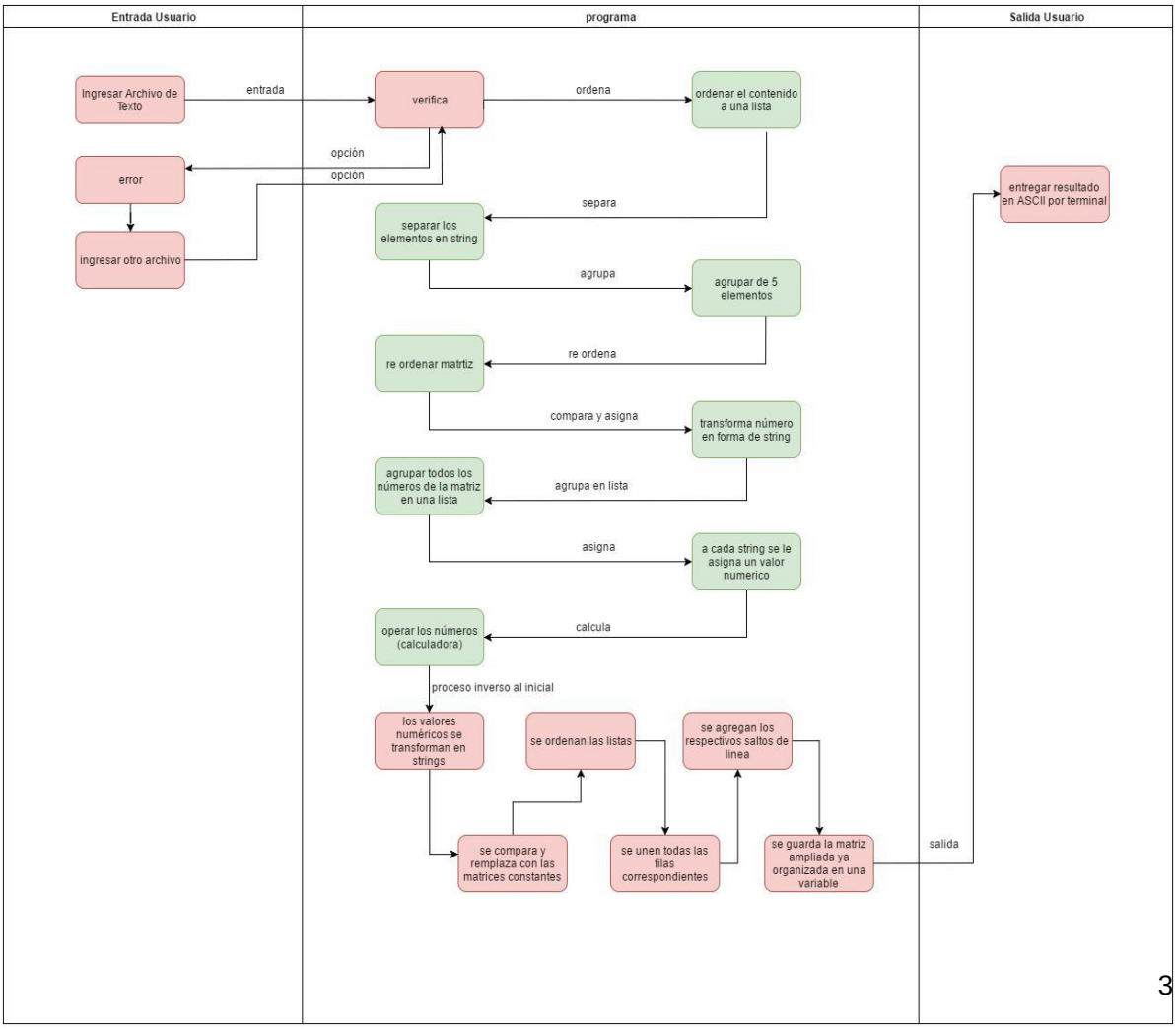
FUNCIONALIDADES

Este programa tiene por objetivo principal procesar un archivo de texto el cual nos entregara una operación matemática la que básicamente se debe resolver y entregar al usuario en el formato requerido. Para ello se realiza una serie de funciones que nos ayude a comprender y elaborar los factores presentes en el programa. Estas funciones son las siguientes:

1. Funcionalidad 1: Ingresar archivo de texto
2. Funcionalidad 2: Verificar archivo de texto
3. Funcionalidad 3: Si existe algún error informar al usuario
4. Funcionalidad 4: Pedir al usuario ingresar otro archivo de texto en caso de error.
5. Funcionalidad 5: Se procede a ordenar el contenido del archivo en una lista.
6. Funcionalidad 6: Se separa la lista obtenida, en varios elementos en formato string.
7. Funcionalidad 7: Separa los elementos obtenidos, y los agrupa de 5.
8. Funcionalidad 8: Los grupos obtenidos se ordenan para forma la matriz 7x5.
9. Funcionalidad 9: Compara y asigna numero en forma de string a la matriz obtenida.
10. Funcionalidad 10: Repetir el proceso con todas las matrices presentes.
11. Funcionalidad 11: Agrupar los números en string.
12. Funcionalidad 12: Asigna un valor número al string obtenido anteriormente.
13. Funcionalidad 13: Realiza la operación numérica requerida.
14. Funcionalidad 14: La operación obtenida se trasforma a un string.
15. Funcionalidad 15: Compara y reemplaza el carácter con las matrices constantes.
16. Funcionalidad 16: Ordena las listas de cada carácter obtenido.
17. Funcionalidad 17: Une cada fila según corresponda.
18. Funcionalidad 18: Agrega saltos de línea al terminar de ordenar cada “numero”
19. Funcionalidad 19: Guarda la matriz final y la organiza en una variable.
20. Funcionalidad 20: Entrega al usuario el resultado de la operación.

ABSTRACCIÓN DE PROCESOS

El programa comienza con la entrada, la cual tiene que insertar el usuario, a este, se le pide que adjunte la ruta de un archivo .txt que contenga una matriz con requisitos específicos, luego de eso es verificada para revisar si cumple estos requisitos, posteriormente el contenido se lee y pasa a ser ordenado a una lista que separar los elementos en strings para facilitar su uso, luego los agrupa en conjuntos de 5, ya que de esta manera podremos ser capaces de separar los dígitos uno de otro, luego coloca todo en orden y se compara con matrices hechas anteriormente, que están definidas como constantes, esto para poder determinar cuál es el número que se está trabajando. Al obtener todos los números se juntan y se transforman en valores numéricos para luego ser operados en un programa de calculadora que entrega la operación ya realizada. Aquí es donde realizamos el proceso inverso para poder transformar un número a una matriz, primero se transforman en strings todos los valores numéricos para facilitar su uso, luego se compara el valor a las matrices constantes, cuando logre identificar de cual matriz se trata esta se descompone y tras ordenarse, se agrupan todas las filas de las matrices de cada digito, se continúa haciendo eso agregando saltos de línea respectivamente. Para finalizar se guarda el arreglo en una sola variable que se le muestra al usuario imprimiéndola en pantalla como salida.



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
# Archivo ingresado por el usuario

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
# Número ASCII entregado por terminal

for linea in resultadoFinal:
print linea
```

CONCLUSIONES

Como grupo, estamos conformes con el avance obtenido, ya que, en términos de programación solo nos faltarían 3 funciones para tener el programa completamente operativo. El proceso para crear nuestro proyecto no fue fácil, era una idea muy profunda que aún no entendemos a la perfección, pero, en el largo periodo de trabajo que aún nos queda, esperamos concretizar muchos aspectos y mejorar otros.

Si bien, el progreso ha sido bastante, la coordinación como grupo ha fallado en algunos aspectos, esto, si bien es propio de trabajar por primera vez juntos, es algo que se ha ido mejorando, pero aún quedan muchos temas por resolver.

REFERENCIAS

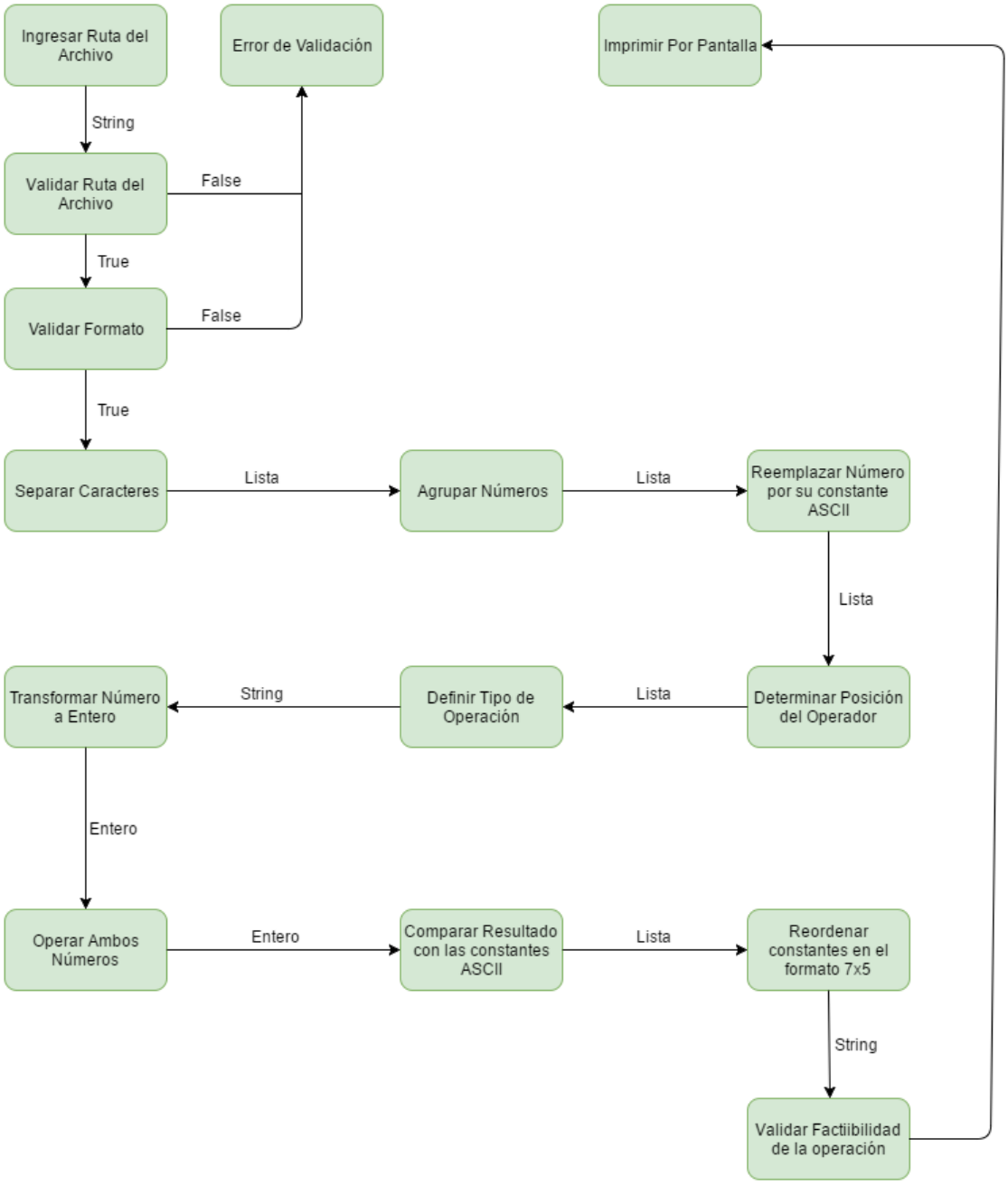
- Andrés Marzal Isabel Gracia. (2010). Introducción a Python. Francia: Universitat Jaumé.

DIAGRAMA DE ABSTRACCIÓN DE PROCESOS

Primero el programa va a solicitar que se ingrese la ruta del archivo, la que será entregada por el usuario en forma de *string*, luego esta ruta será verificada para comprobar que exista el archivo y luego comprobar que el formato ingresado en el archivo sea el requerido. Al cumplirse de manera correcta estas dos validaciones el programa asignará las líneas en listas y va a separar los caracteres de cada línea.

Después se agruparán los caracteres de las filas para que en vez de que cada elemento de la lista sea una fila del documento, sea un número distinto de la operación. Esta nueva lista se comparará con las constantes que posee el programa para que en vez de un número ASCII se transforme en un número (en forma de *string*), logrando una lista de *strings*. Luego va a buscar que operación matemática va a realizar el programa, y a la operación que encuentre le va a ubicar un índice dentro de la lista, este índice será utilizado para crear los números enteros que van a participar de la operación matemática. Luego que se conformen los dos números van a ser operados, y el resultado puede resultar ser un entero o *float*, dependiendo si el resultado es con números decimales.

El resultado anterior pasará en forma de *string* a una lista, la cual cada elemento será un número en orden correlativo al resultado. Cada número será reemplazado por su número ASCII, y finalmente se reordenarán los números para que quede en el formato estándar de salida (7 filas y números de 5 columnas, separados por un punto). Luego, este resultado será asignado a un *string* que será mostrado por pantalla.


Ingresar Ruta del Archivo
Error de Validación
Imprimir Por Pantalla
String
Validar Ruta del Archivo
False
True
Validar Formato
False
True
Separar Caracteres
Lista
Agrupar Números
Lista
Reemplazar Número por su constante ASCII
Lista
Transformar Número a Entero
String
Definir Tipo de Operación
Lista
Determinar Posición del Operador
Entero
Operar Ambos Números
Entero
Comparar Resultado con las constantes ASCII
Lista
Reordenar constantes en el formato 7x5
String
Validar Factiibilidad de la operación

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,PU
NT]
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"+archivoEnUnaLine
a[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"+archivoEnUnaLine
a[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()
#_____________________________________________________________________________
_____________________________________________________________________________
____________#
```

MANUAL DE USUARIO

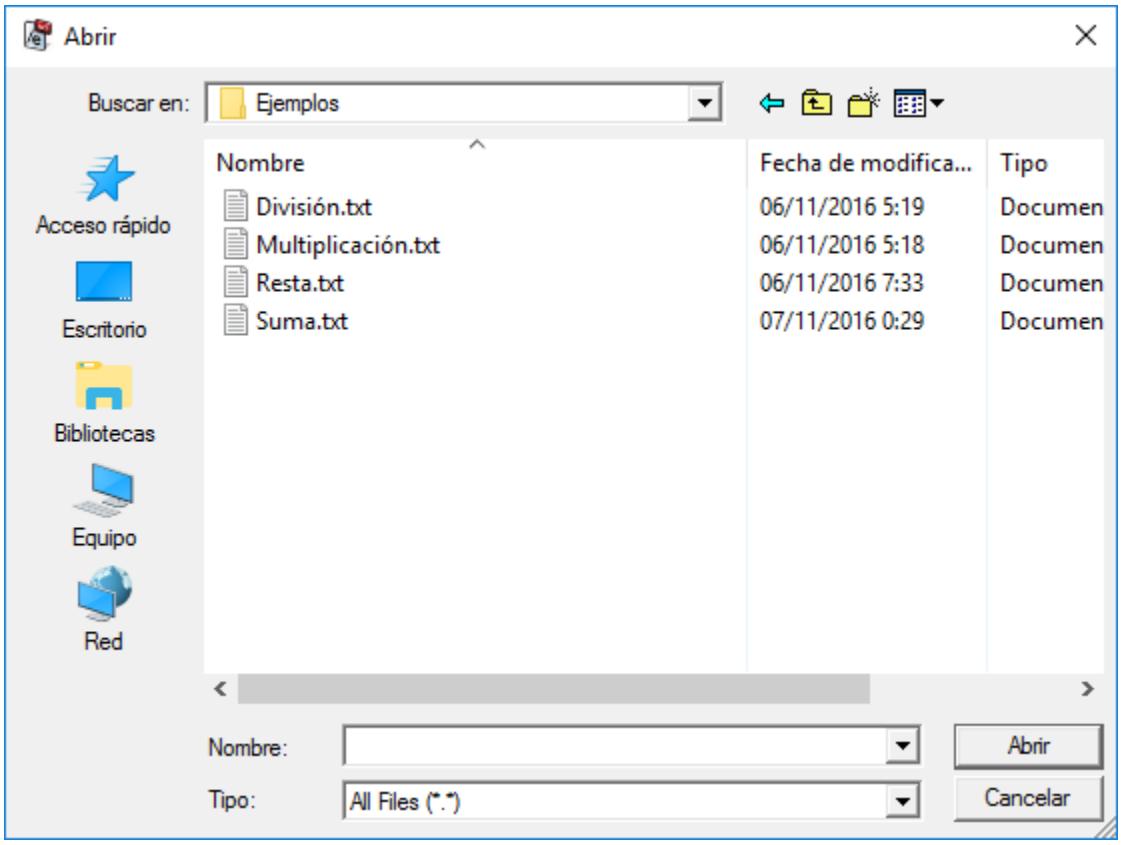
En primer lugar, para poder ejecutar la Calculadora ASCII es necesario contar con la versión 2.7.12 de Python, la cual se encuentra disponible en la página web oficial de Python[2]. Esta versión contiene las librerías necesarias para que el programa funcione correctamente. Si llegase a fallar, se incluye la librería *TKinter* en la carpeta "Archivos" que debe ser copiada en la ruta "C:\Python27\libs" en caso que ya posea alguna versión de Python instalada en su computador. Esta librería se llama "_tkinter.lib".

Luego de haber instalado Python 2.7.12, o la librería que le hacía falta para poder correrlo sin problemas, usted podrá ejecutar el programa, y se encontrará inmediatamente con la siguiente ventana:

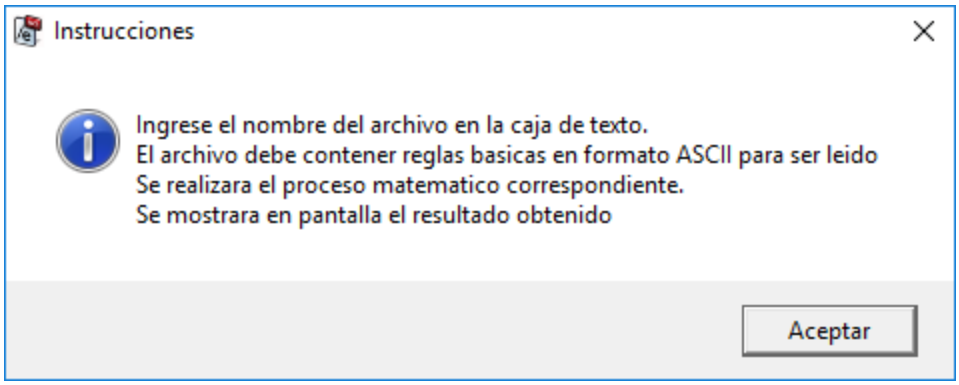
[2] https://www.python.org/downloads/

Usted tiene dos opciones:

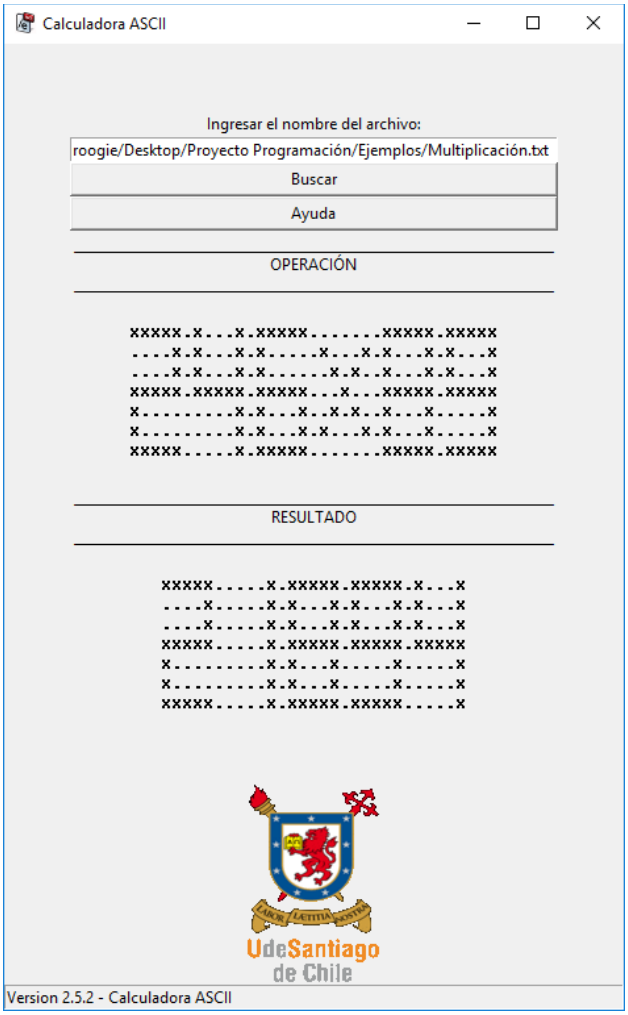
1. Si el archivo se encuentran dentro de la misma carpeta que la calculadora ASCII, usted puede escribir el nombre del archivo en la barra de búsqueda y se ejecutará el programa. Para esto se debe escribir sólo el nombre del archivo, sin su extensión. Por ejemplo si usted desea abrir el archivo “Multiplicación.txt” que se encuentra en la misma carpeta que el programa, debe escribir en la barra de búsqueda “Multiplicación”, presionar la tecla *Enter* y se realizará la operación que se encuentra dentro del archivo.
2. Al hacer click en el botón “Buscar” se abrirá la ventana “Abrir”, con la que podrá buscar directamente el directorio del archivo y abrirlo, ya sea haciendo doble click, o presionando el botón “Abrir”.



También tiene la opción de seleccionar el botón “Ayuda” que mostrará la ventana:



Luego de seleccionar el archivo, ya sea por el método (1) o (2), la Calculadora ASCII realizará la operación matemática y mostrara por pantalla la operación que se realizó (para comprobar que es la deseada), y el resultado al que se llegó. Por ejemplo, al seleccionar el archivo “Multiplicación.txt” ocurrirá lo siguiente:



El formato para que se realice la operación matemática es:

- El archivo a ejecutar debe ser un archivo de texto (ejemplo.txt).
- Los números (u operadores) deben estar hechos de 7 filas y 5 columnas y compuesto solo por puntos y equis (las cuales deben ir en minúsculas).
- Los números deben ir separados entre sí por un punto.
- Las operaciones son entre números enteros y positivos.
- El primer dígito **no** debe ser un operador matemático.

Números

```
....x.xxxxx.xxxxx.x...x.xxxxx.xxxxx.xxxxx.xxxxx.xxxxx.xxxxx
....x.....x.....x.x...x.x.....x.........x.x...x.x...x.x...x
....x.....x.....x.x...x.x.....x.........x.x...x.x...x.x...x
....x.xxxxx.xxxxx.xxxxx.xxxxx.xxxxx.....x.xxxxx.xxxxx.x...x
....x.x.........x.....x.....x.x...x.....x.x...x.....x.x...x
....x.x.........x.....x.....x.x...x.....x.x...x.....x.x...x
....x.xxxxx.xxxxx.....x.xxxxx.xxxxx.....x.xxxxx.xxxxx.xxxxx
```

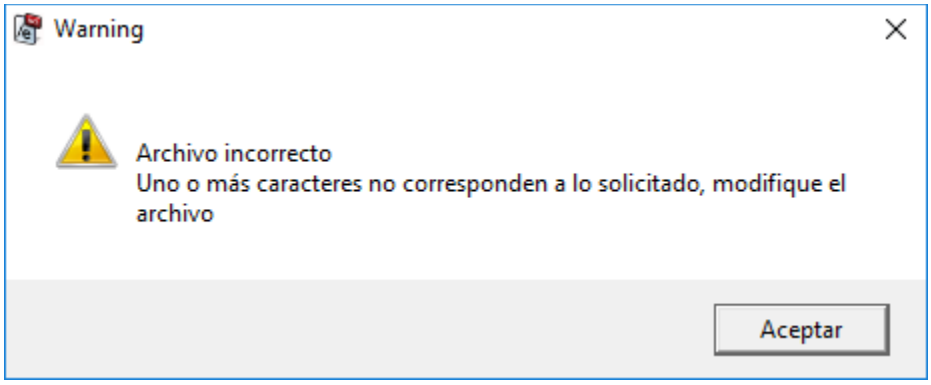
Operadores

```
.....................
..x.........x...x.....x
..x..........x.x.....x.
xxxxx.xxxxx...x.....x..
..x..........x.x...x...
..x.........x...x.x....
.....................
```

Si desea finalizar sus operaciones matemáticas cierre la ventana.

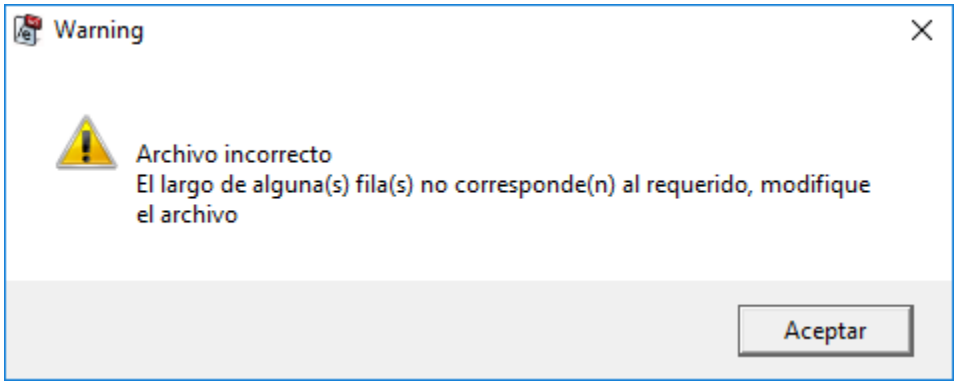
Posibles errores

Si utiliza algún caracter distinto de puntos o equis en minúsculas el programa arrojará el siguiente error:



Revise que el archivo ingresado solo contenga estos caracteres.

En el caso que existan columnas extras, o que existan filas más largas que otras el error será el siguiente:



Para resolverlo debe ajustar el formato del archivo para que sea igual al requerido por la calculadora.

Si desea dejar de ocupar la Calculadora ASCII cierre la ventana.

CONCLUSIONES

Como grupo estamos conformes con el producto generado, según nuestras pruebas el programa funciona correctamente, este fue finalizado con anticipación lo que nos permitió arreglar posibles fallos, considerar errores del usuario, y crear una interfaz estética y amigable.

Nuestra mayor falencia a lo largo del curso fue la de generar informes y presentaciones que transmitieran el esfuerzo realizado en la programación del código, esto se manifestó de forma evidente en la dificultad que tuvimos de generar un diagrama de abstracción que representara realmente nuestro código y el proceso que hicimos para crearlo.

Con respecto al trabajo en grupo, fue un proceso difícil ya que partió desde el desconocimiento casi total de cada uno de los integrantes, y aunque nos asignamos roles al inicio del curso, estos no delimitaban claramente lo que debía hacer cada uno, solo con el tiempo que trabajamos juntos fue que pudimos conocer mejor nuestros intereses, motivaciones y habilidades. Esto finalmente, permitió asignarnos roles más óptimos, delimitar mejor nuestras funciones y trabajar de forma más eficiente.

REFERENCIAS

Bodnar, J. (01 de Diciembre de 2015). *ZetCode*. Obtenido de ZetCode: http://zetcode.com/gui/tkinter/

Marzal, A., & Gracia, I. (2003). *Introducción a la Programación con Python.* Castellón de la Plana: Publicacions de la Universitat Jaume.