Universidad De San Carlos de Guatemala
Facultad de Ingeniería
Escuela de Ciencias y Sistemas
Lenguajes formales y de programación
Sección “B-”

“MANUAL DE USUARIO”

Diego André Mazariegos Barrientos

Carné: 202003975

Objetivos

General:

Proporcionar al usuario del software una guía con la cual pueda conocer sobre el manejo adecuado de la aplicación adquirida y de esta manera lograr un uso adecuado, fácil y totalmente eficiente de la misma.

Específicos:

- Brindar al usuario, mediante una forma gráfica y sencilla de entender, todos los datos necesarios para comprender el funcionamiento lógico de la aplicación y la manera en que simula los procesos requeridos.
- Entregar al usuario las indicaciones y pasos necesarios a seguir para que la simulación de su negocio se la correcta y evitar que se generen anomalías en los resultados por un uso inadecuado.

Introducción

Este manual de usuario tiene como fin dar a conocer a todos los usuarios que hagan uso del software las funcionalidades y pasos a seguir para darle el uso más eficaz y obtener resultados satisfactorios al momento de tomar decisiones apoyados en los resultados de las simulaciones generadas por la aplicación “Proyecto 2”. Para cumplir con el objetivo propuesto se incluye la descripción de las pantallas que el usuario manejara para el ingreso de datos, manejo de la simulación y de resultados, todo esto a través de gráficos para su mayor compresión.

Descripción del Programa

La aplicación tiene como objetivo cumplir con los requerimientos solicitados para la toma de decisiones en cualquier negocio debido al crecimiento en la demanda de sus productos, por lo que la aplicación posee un analizador léxico para la lectura y el análisis de los datos de entrada con un formato previamente establecido para la generación de reportes entre otras funcionalidades dependiendo de la cadena de entrada. Con dichos datos se generan líneas de comandos y se despliegan varias funcionalidades para el manejo de la información, dicho comandos son solicitados en el archivo de entrada por medio de un cuadro de texto en la interfaz de forma agradable, y como último la aplicación cuenta con una opción de generación de reportes de los errores léxicos y tokens encontrados en el archivo de entrada

Descripción de las Funciones del Programa

Cargar archivo:

Se refiere a la búsqueda del archivo de entrada en formato LFP mediante el explorador de archivos.

Analizar archivo:

Esta opción permite analizar el archivo cargado previamente, es la encargada de hacer la lectura respectiva del archivo verificar si no existen errores léxicos y/o sintácticos en el programa, guardar todos los tokens encontrados, así como se encarga de todo el proceso y manejo de las variables en el programa y guardar todos los errores léxicos y/o sintácticos en una lista si es que existen.

Reporte tokens:

Esta opción genera un archivo en formato HTML denominados reporte de tokens en el cual por medio de tablas y de una forma agradable se presentan las listas de los tokens encontrados.

Reporte errores:

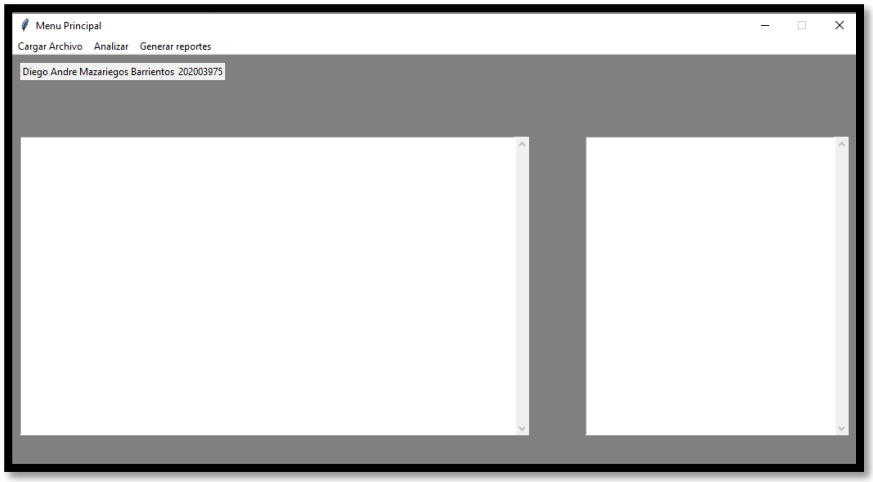
Esta opción genera un archivo en formato HTML denominados reporte de errores en el cual por medio de tablas y de una forma agradable se presentan las listas de los errores encontrados, si existiesen también de los errores léxicos y/o sintácticos.

Salir:

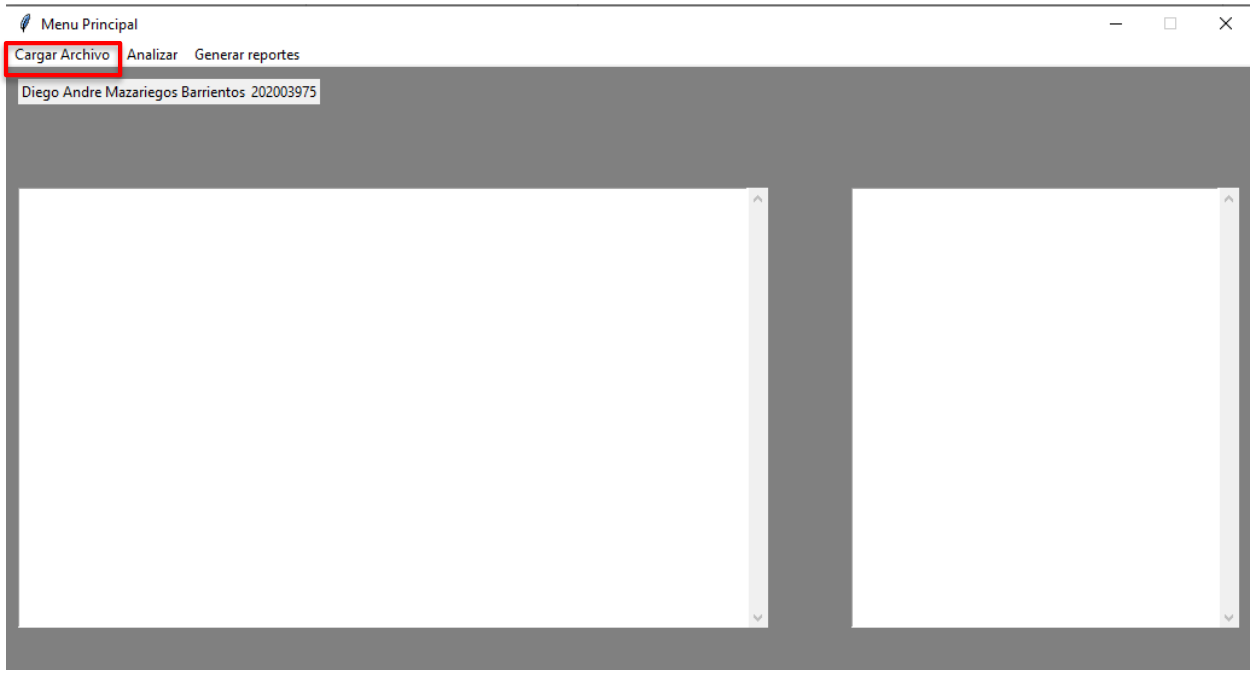
Esta opción detendrá la ejecución del programa hará aparecer un mensaje emergente en el cual preguntará si se desea salir del programa y en caso de afirmar la ejecución del programa se detendrá.

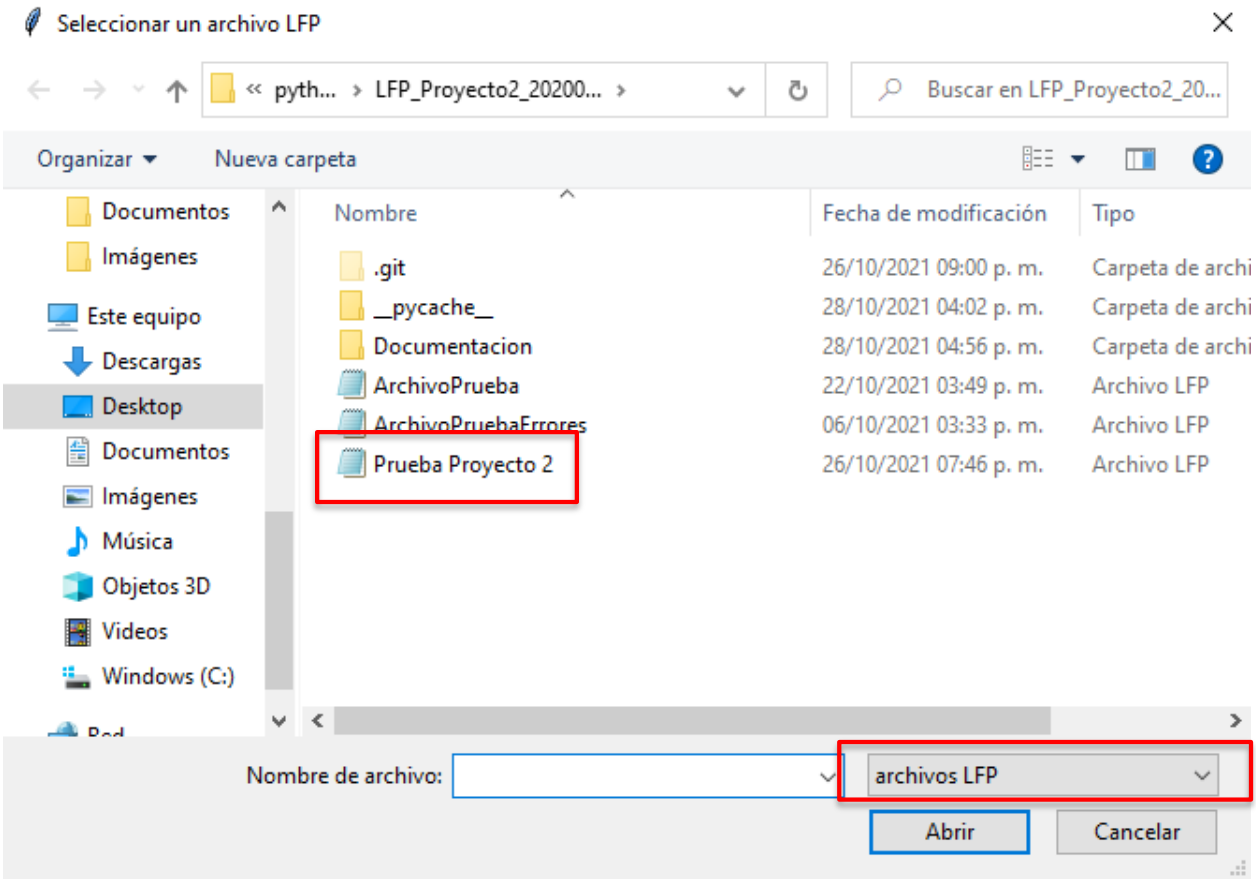
Manejo de la Interfaz Gráfica

Al momento de ejecutar el programa se desplegará esta pantalla de presentación.

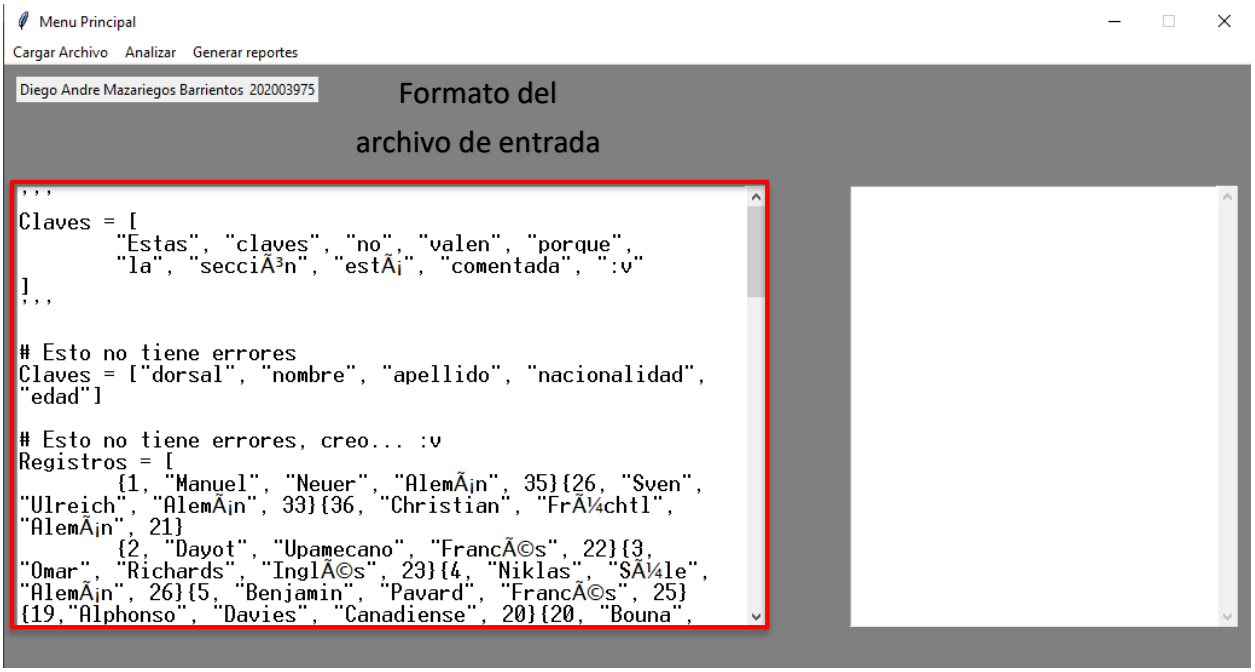


Para cargar el archivo, se desplegará una ventana para buscar el archivo y darle click.

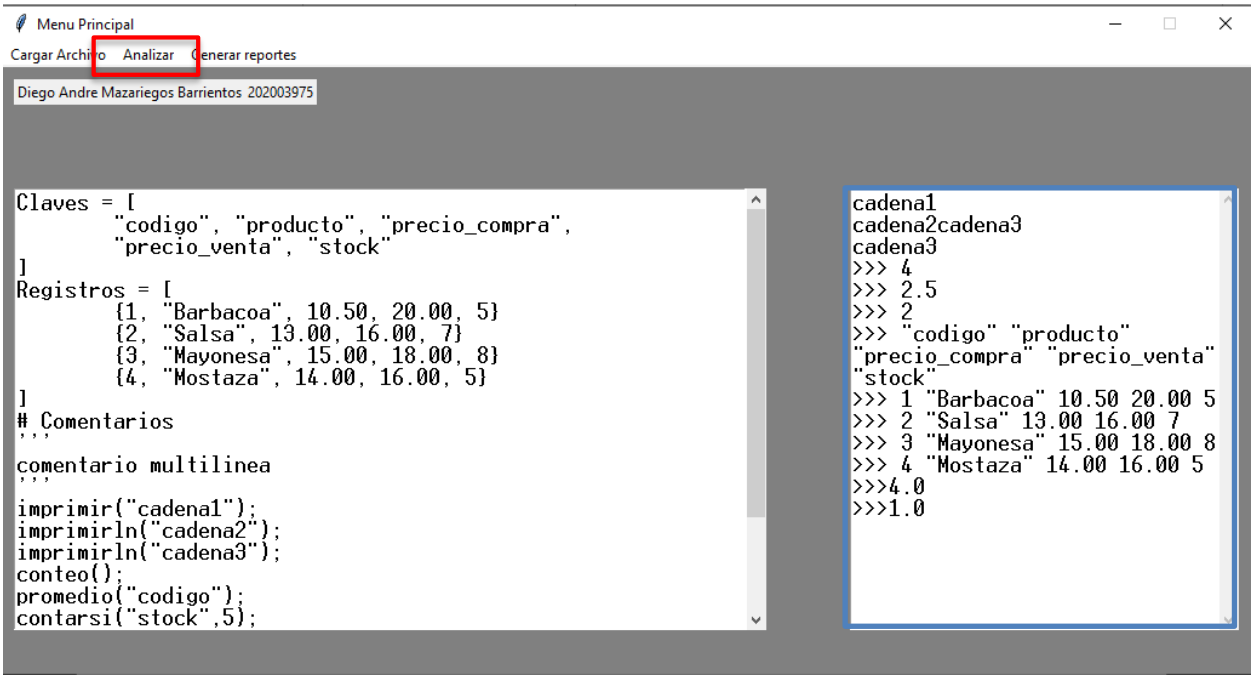




Asegúrese de elegir un archivo con el formato correcto y al darle click se desplegará su contenido en un campo de texto en el menú de la interfaz.



Para analizar el archivo dar click sobre el cuadro en rojo en la siguiente ilustración y esto empezará el analisis léxico y sintáctico del cuadro de texto verificando que la entrada este escrita de forma correcta.



En caso la entrada no posea errores se imprimira en consola el texto lo requerido por el usuario en el área de comandos.

Sección comandos

Existen diversas combinaciones de comandos con las que se pueden trabajar dentro del programa, a continuación, se mostrara de forma descriptiva y gráfica la funcionalidad de cada uno, así como su correcta estructura al momento de escribirlos.

Comandos Claves: se declaran los claves o campos por los que están construidos los registros, su estructura está formada por la palabra reservada Claves, seguido de signo igual, corchete de apertura, lista de claves y corchete de cierre.

Lista de claves está formada por cadenas de caracteres encerradas entre comillas y separadas por coma.

Ejemplo:

```
Claves = [
        "codigo", "producto", "precio_compra",
        "precio_venta", "stock"
]
```

Comando Registros: se detallan los registros que se quieren analizar y sigue la estructura dada por palabra reservada Registros, signo igual, corchete de apertura, lista de registros y corchete de cierre.

Lista de registros: Cada registro está encerrado entre llave de apertura y llave de cierre y sus valores están separados por comas, estos valores pueden ser cadenas de texto, enteros o decimales

Ejemplo:

```
Registros = [
        {1, "Barbacoa", 10.50, 20.00, 6}
        {2, "Salsa", 13.00, 16.00, 7}
        {3, "Mayonesa", 15.00, 18.00, 8}
        {4, "Mostaza", 14.00, 16.00, 4}
]
```

Comentarios de una sola línea: Se representan con un numeral y finalizan con un salto de línea.

Ejemplo:

```
# Comentarios
```

Comentarios multilínea: Inicia con tres comillas simples y finaliza con tres comillas simples.

Ejemplo:

```
'''
comentario multilinea
'''
```

Comando imprimir: Imprime por consola el valor dado por la cadena.

Ejemplo:

```
imprimir("cadena");
```

Comando imprimirln: Imprime por consola el valor dado por la cadena, si determina que existe un comando después de él y es el mismo comando se adhiere la cadena anterior y se imprime dos veces el valor del segundo comando una vez adherido el valor a la cadena del comando 1 y la segunda vez con un salto de línea.

Ejemplo:

```
imprimirln("cadena");
```

Comando conteo: Imprime por consola la cantidad de registros en el arreglo de registros.

Ejemplo:

```
conteo();
```

Comando promedio: Imprime por consola el promedio del campo dado.

Ejemplo:

```
promedio("cadena");
```

Comando contarsi: Imprime por consola la cantidad de registros en la que el campo dado sea igual al valor dado.

Ejemplo:

```
contarsi("cadena",0);
```

Comando datos: Imprime por consola los registros leídos.

Ejemplo:

```
datos();
```

Comando max: Encuentra el valor máximo del campo dado.

Ejemplo:

```
max("cadena");
```

Comando min: Encuentra el valor mínimo del campo dado.

Ejemplo:

```
min("cadena");
```

Comando exportarReporte: Genera un archivo html con una tabla en donde se encuentren los registros leídos y con el título como parámetro.

Ejemplo:

```
exportarReporte("cadena");
```

Imagen de demostración: la siguiente imagen es una demostración de los comandos y sus acciones.

```
Claves = [
        "codigo", "producto", "precio_compra",
        "precio_venta", "stock"
]
Registros = [
        {1, "Barbacoa", 10.50, 20.00, 5}
        {2, "Salsa", 13.00, 16.00, 7}
        {3, "Mayonesa", 15.00, 18.00, 8}
        {4, "Mostaza", 14.00, 16.00, 5}
]
# Comentarios
'''
comentario multilinea
'''
imprimir("cadena1");
imprimirln("cadena2");
imprimirln("cadena3");
conteo();
promedio("codigo");
contarsi("stock",5);
```

```
        {2, "Salsa", 13.00, 16.00, 7}
        {3, "Mayonesa", 15.00, 18.00, 8}
        {4, "Mostaza", 14.00, 16.00, 5}
]
# Comentarios
'''
comentario multilinea
'''
imprimir("cadena1");
imprimirln("cadena2");
imprimirln("cadena3");
conteo();
promedio("codigo");
contarsi("stock",5);
datos();
max("codigo");
min("codigo");
exportarReporte("Reporte registros");
```

Imagen de resultado:

```
cadena1
cadena2cadena3
cadena3
>>> 4
>>> 2.5
>>> 2
>>> "codigo" "producto"
"precio_compra" "precio_venta"
"stock"
>>> 1 "Barbacoa" 10.50 20.00 5
>>> 2 "Salsa" 13.00 16.00 7
>>> 3 "Mayonesa" 15.00 18.00 8
>>> 4 "Mostaza" 14.00 16.00 5
>>>4.0
>>>1.0
```

Resultado comando exportarReporte:

REPORTE DE DATOS

"Reporte registros"

"codigo"	"producto"	"precio_compra"	"precio_venta"	"stock"
1	"Barbacoa"	10.50	20.00	5
2	"Salsa"	13.00	16.00	7
3	"Mayonesa"	15.00	18.00	8
4	"Mostaza"	14.00	16.00	5

Luego de haber analizado la entrada se habilitará la opción de generar reporte de tokens y errores en donde de forma agradable se presenta un listado de los tokens encontrados en la entrada, así como de los errores si en dado caso existiesen.

REPORTE DE TOKENS

Tabla de tokens

Token	Lexema	Patrón	Fila	Columna
Palabra reservada Claves	Claves	Le = [A_Z, a_z] -> Palabra = Le+	1	0
Simbolo =	=	Sim = (=, [,],',', {,}, (,),';')	1	8
Simbolo [	[	Sim = (=, [,],',', {,}, (,),';')	1	11
Cadena	"codigo"	Le = [A_Z, a_z] -> Palabra = Le+	2	6
Simbolo ,	,	Sim = (=, [,],',', {,}, (,),';')	2	14
Cadena	"producto"	Le = [A_Z, a_z] -> Palabra = Le+	2	17

En caso de no existir errores el reporte de errores no mostrar ningún error.

REPORTE DE ERRORES

Tabla de errores léxicos

Caracter	Fila	Columna

En caso de existir errores el reporte de errores se mostrará de la siguiente forma, cabe aclarar que la columna marca el punto de comienzo del error.

En caso de seleccionar la opción salir de la barra menú saltara el siguiente mensaje solicitando una confirmación. En caso de aceptar el programa terminara su ejecución. Y en caso de cancelar el programa volverá a la ventana principal.