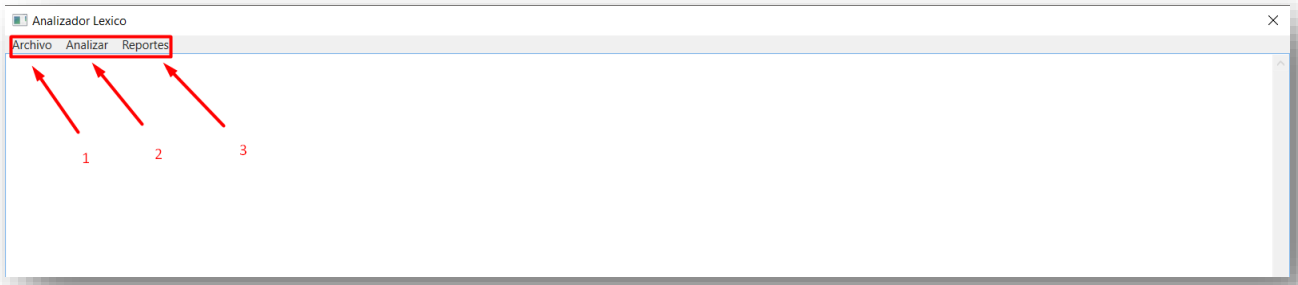
Universidad de San Carlos De Guatemala
Facultad de Ingeniería
Lenguajes Formales y de Programación A+
Ing. Otto Rodríguez
Aux. Jorge Veliz

Proyecto 1 – Manual De Usuario

Pantalla inicial

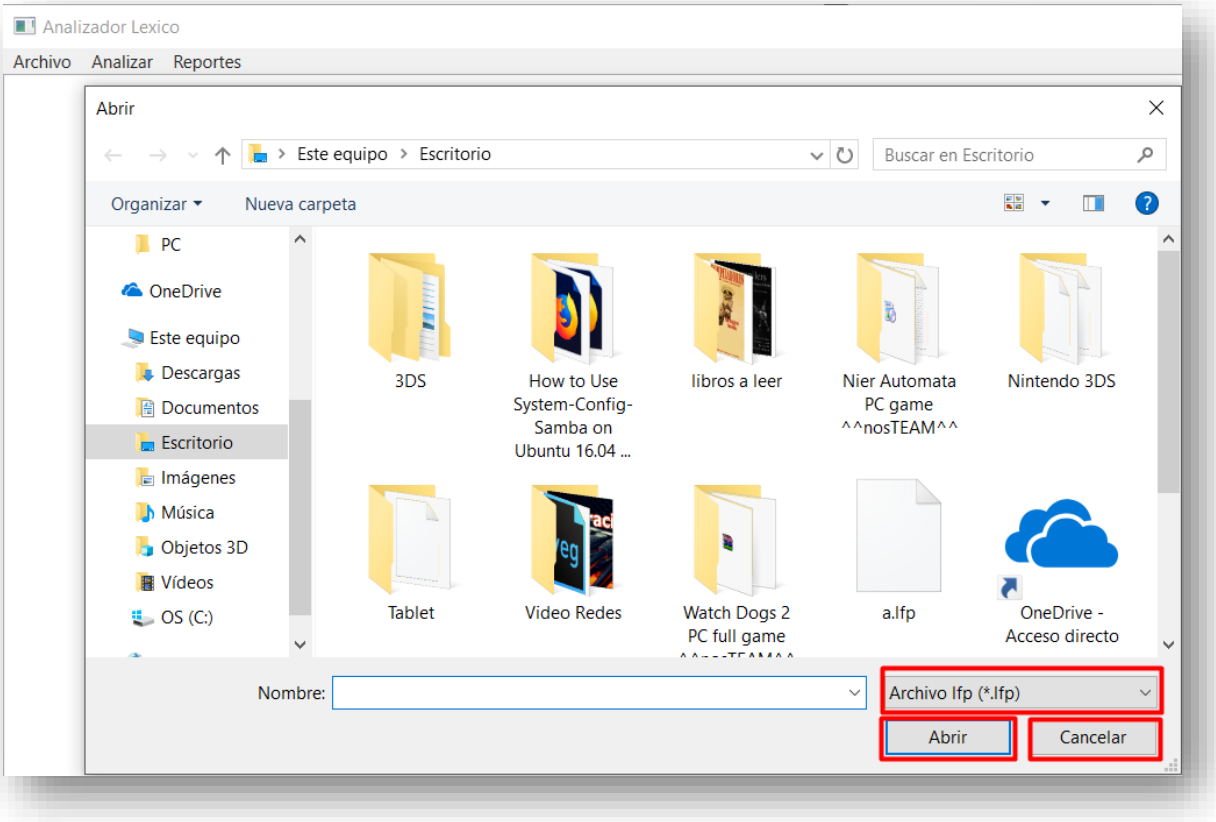
Pantalla inicial que se muestra al usuario al iniciar la aplicación, cuenta con las siguientes opciones:



1. **Archivo**

Al seleccionar la opción **"Archivo"** se le muestran al usuario una serie de opciones para gestionar el uso de archivos en el proyecto, entre estas opciones encontramos:

1.1 Abrir

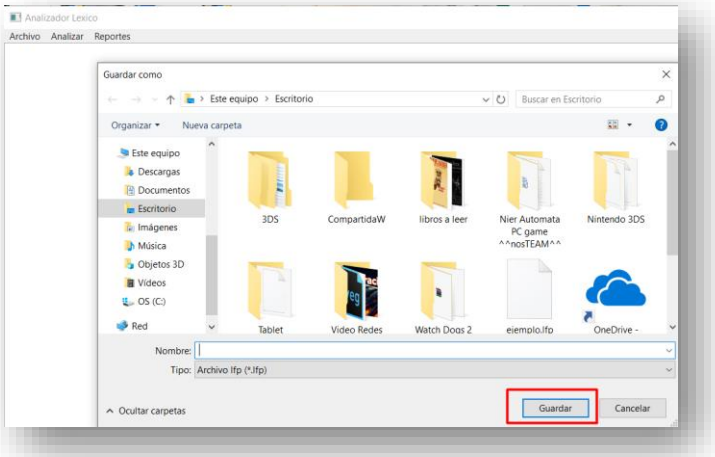


Al seleccionar la opción **"Abrir"** el sistema mostrará un explorador de archivos al usuario, el cual únicamente mostrara archivos **".lfp"**, desde el cual deberá seleccionar el archivo que desea abrir.

1.2 Guardar

Al Seleccionar la opción **"Guardar"** el sistema guarda los cambios que se hayan realizado en el archivo en la ruta especificada, si es que se utilizó la opción **"abrir"**, en caso de no existir una ruta especificada, se mostrara un explorador de ficheros para que el usuario seleccione donde quiere guardar el archivo.

1.3 Guardar Como

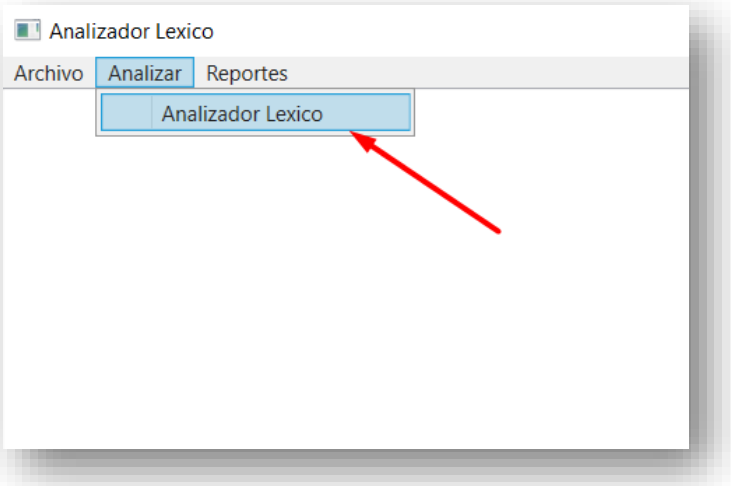


Al seleccionar esta opción el sistema mostrar un explorador de ficheros para que el usuario seleccione donde quiere guardar el archivo.

1.4 Salir

Al seleccionar esta opción el sistema se cerrará.

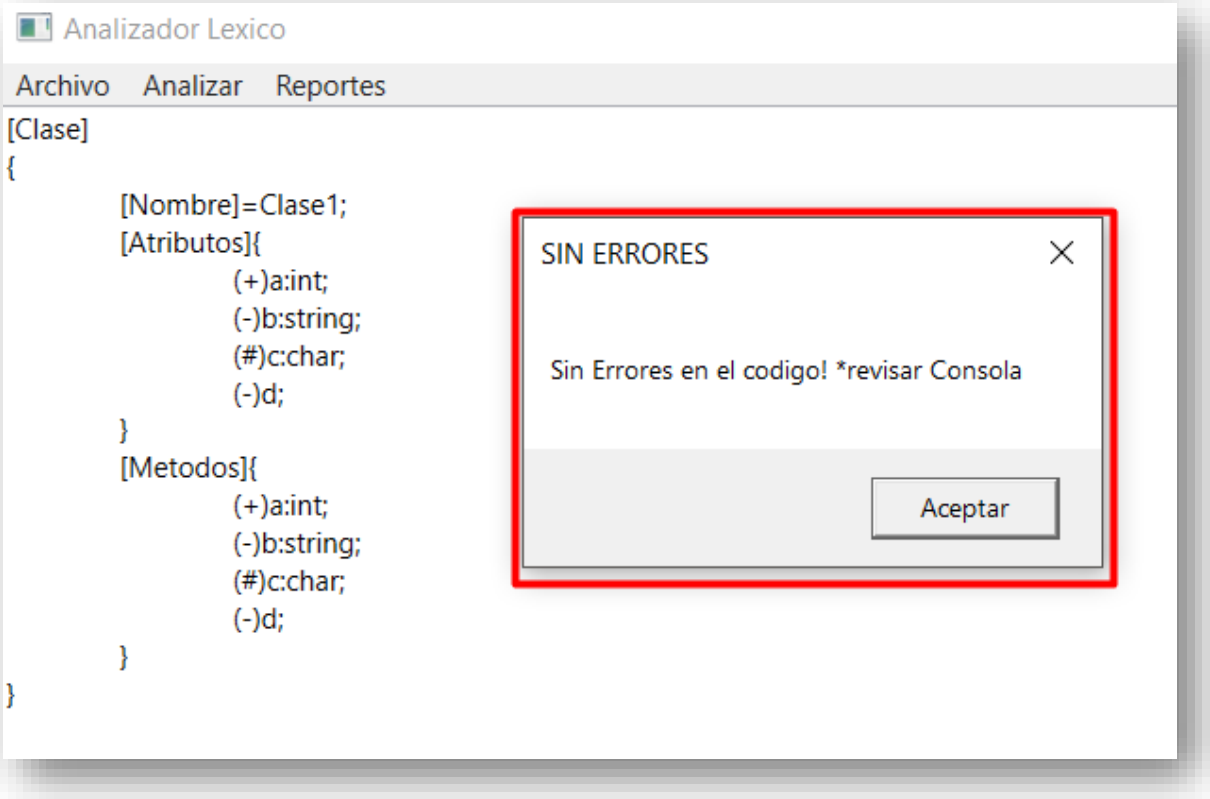
2. Analizar



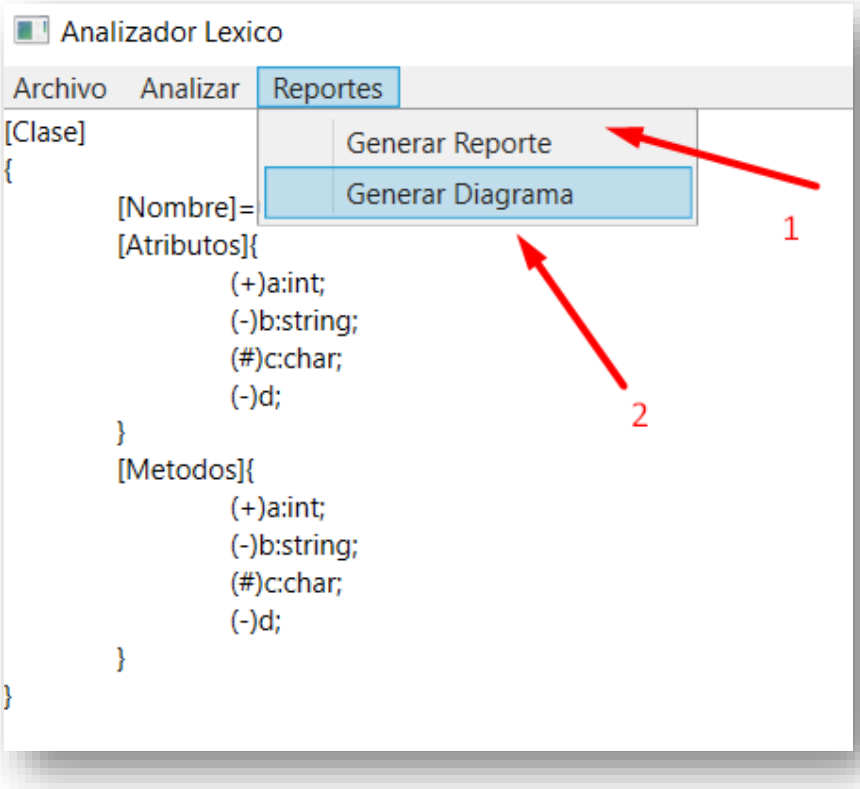
Al seleccionar el apartado "**Analizar"** el sistema mostrara la siguiente opción:

2.1 Analizador Léxico

Al seleccionar esta opción el sistema tomara el código que exista y lo analizar léxicamente, mostrando un mensaje si existe o no algún error léxico.

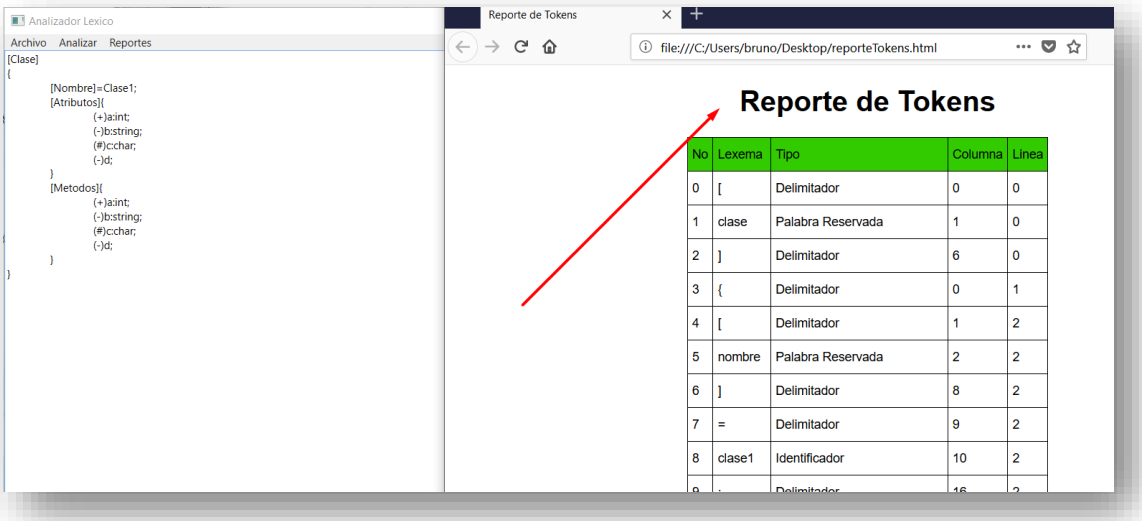


3. Reportes



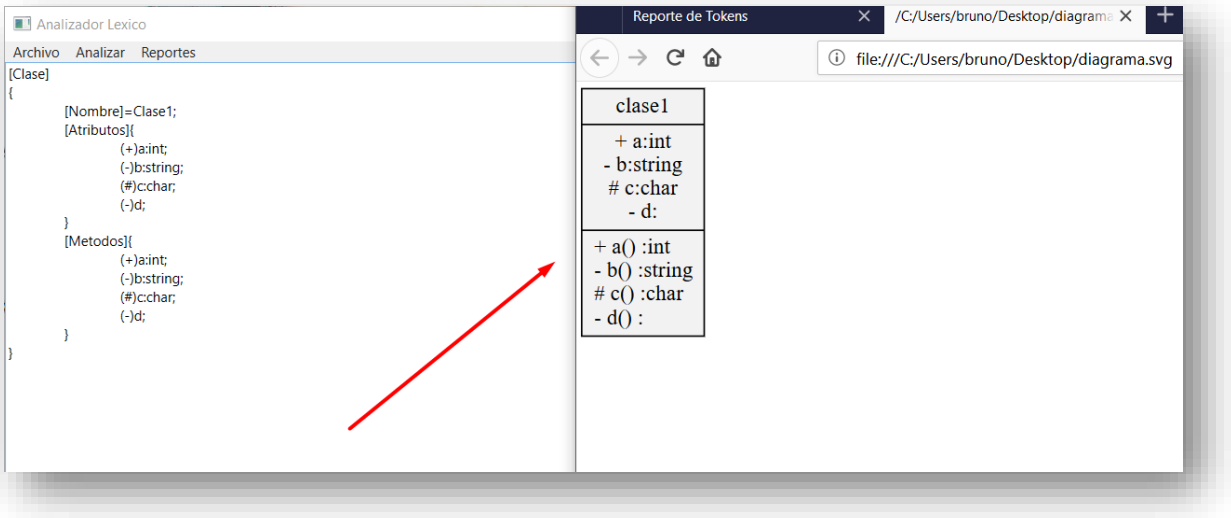
Al seleccionar el apartado "**Reportes"** el sistema mostrara las siguientes opciones:

3.1 Generar Reporte



Al seleccionar esta opción el sistema tomará el código y procederá analizarlo para generar un reporte de tokens, creando un archivo **".html"** en el escritorio y mostrándolo en el navegador predeterminado. En caso de existir algún error léxico el sistema mostrara reporte de errores en lugar de tokens.

3.2 Generar Diagrama



Al seleccionar esta opción es sistema tomará el código y procederá analizarlo para generar un diagrama con la información obtenida, creando un archivo **“.svg”** y un **“.txt”** en el escritorio y mostrando el **“.svg”** en el navegador predeterminado mientras que el **“.txt”** contiene el código **Graphviz** del diagrama. En caso de existir algún error léxico el sistema mostrara un mensaje de error.

Descripción del código

A continuación, se presenta el código que acepta el programa, tomado del enunciado del proyecto.

- **Bloque Clase:**

```
[Clase]
{
        [Nombre] = <identificador>;
        <BloqueAtributos>
        <BloqueMetodos>
}
```

En donde:

- Clase: Sera una palabra reservada que iniciara la descripción de un objeto clase, para luego diagramarlo. Este bloque posee lo siguiente:
    - **Nombre:** Palabra reservada seguida de un "identificador", la cual será el nombre de la clase a crear.
    - **Bloque Atributos:** Este bloque representa todos los atributos que puede contener una clase.
      NOTA: Este bloque puede o no venir (si este bloque no está incluido en una clase quiere decir que la clase representa una interfaz).
    - **Bloque Métodos:** A diferencia del Bloque Atributos este bloque **siempre** debe de venir y representará todos los métodos que tendrá una clase.

- **Bloque Atributos:**

```
[Atributos] {
                <Visibilidad> <Identificador> : <Tipo> ;
                <Visibilidad> <Identificador> : <Tipo> ;
                <Visibilidad> <Identificador>;
                ...
        }
```

En donde:

- **Visibilidad:** Representa la visibilidad que tiene un atributo dentro de la misma clase y existe la posibilidad que sea cualquiera de las siguientes 3:
    - Público (+)
    - Privado (-)
    - Protegido (#)
- **Identificador:** Sera el nombre que recibirá el atributo a definir.
- **Tipo:** Definirá el tipo que poseerá el atributo, el cual está definido por un <Identificador>. Este parámetro puedo o no venir.

- **Bloque Métodos:**

```
[Metodos] {
        <Visibilidad> <Identificador> : <TipoRetorno> ;
        <Visibilidad> <Identificador> : <TipoRetorno> ;
        <Visibilidad> <Identificador> ;
        …
    }
```

En donde:

- **Visibilidad:** Representa la visibilidad que tiene un método dentro de la misma clase y existe la posibilidad que sea cualquiera de las siguientes 3:
  - Público (+)
  - Privado (-)
  - Protegido (#)
- **Identificador:** Sera el nombre que recibirá el método a definir.
- **TipoRetorno:** Definirá el tipo de retorno que poseerá el método, el cual está definido por un <Identificador>. Este parámetro puedo o no venir.

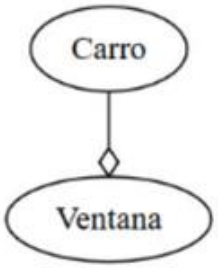
- **Bloque Asociación:**

```
[Asociacion] {
        <Identificador> : <TipoAsociación> : <Identificador>;
        <Identificador> : <TipoAsociación> : <Identificador>;
        …
}
```

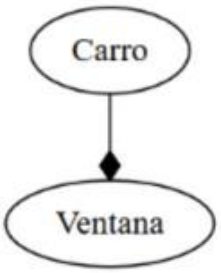
En donde:

- **Identificador:** Sera el nombre de una clase que se quiere asociar.
- **Tipo Asociación:** Definirá la asociación que se desea representar entre clases y puede ser una de las siguientes opciones:

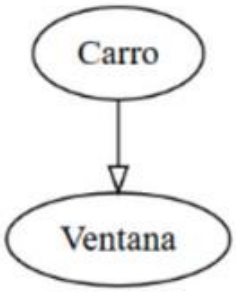
- **Agregación**



- **Composición**



- **Herencia**



- **AsociaciónSimple**