

Universidad de San Carlos de Guatemala

Facultad de Ingeniería

Escuela de Ciencias y Sistemas

Lenguajes Formales y de Programación

Catedrática:

Inga. Asunción Mariana Sic Sor

Tutor académico:

Enrique Alejandro Pinula Quiñonez



PROYECTO 1

OBJETIVOS:

- Implementar una solución de software que proporcione un análisis léxico al lenguaje formal proporcionado.
- Implementación de Graphviz para la construcción de diagramas.
- Creación de una interfaz gráfica intuitiva para el usuario en la solución de software.

DESCRIPCIÓN

La empresa “Grafos Guatemala” desea implementar un software que sea capaz de analizar un archivo de entrada con una sintaxis específica y a partir de este generar grafos con la herramienta Graphviz. Para esto usted ha sido contratado para desarrollar dicho software.

Debido a que esta aplicación estará abierta al público, es necesario que la aplicación cuente con una interfaz gráfica intuitiva y con soporte para errores léxicos que puedan cometer los usuarios.

Debe desarrollarse una aplicación que contenga las siguientes opciones:

Cargar archivo

En esta opción se debe poder escoger el archivo con extensión “.code” y este será cargado a la memoria del sistema, listo para ser ejecutado.

Ejecutar Archivo

Esta opción analizará el archivo de entrada y generará las imágenes que este contenga. Es importante validar que se haya cargado un archivo con anterioridad, sino se deberá mostrar un error notificando que primero se debe cargar un archivo de entrada.

Visualización de imágenes

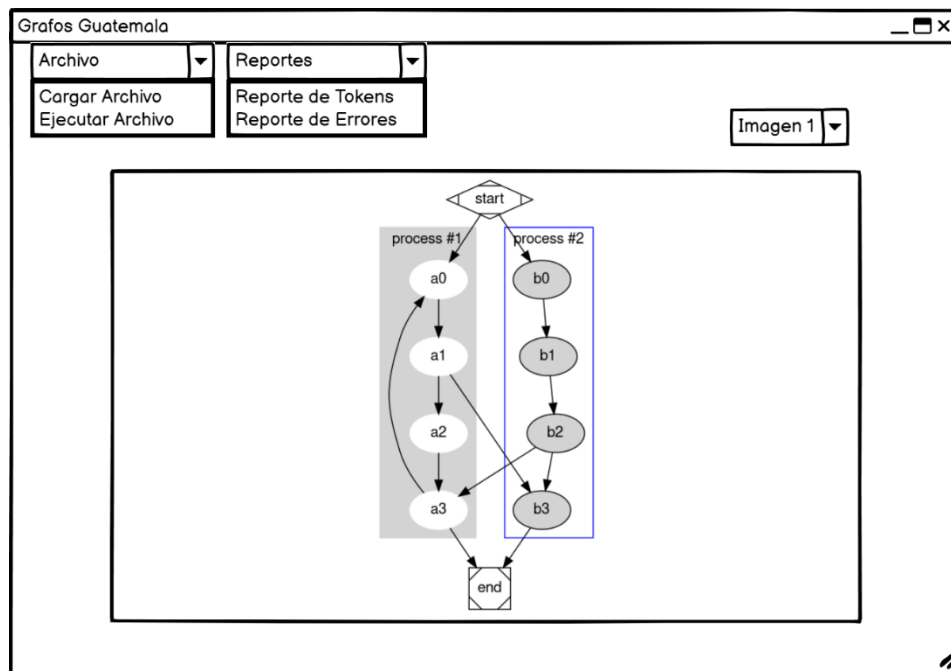
Luego de haber generado las imágenes con el archivo de entrada, debe ser posible visualizar las imágenes dentro de la aplicación.

Reportes

Debe existir un área de reportes, estos serán exportados en formato HTML y se debe poder escoger la ubicación donde estos serán almacenados. Los reportes disponibles deben ser:

- Reporte de tokens, este debe contener:
 - Nombre del token.
 - Lexema.
 - Fila.
 - Columna.
- Reporte de errores, este debe contener:
 - Token que estaba siendo reconocido.
 - Lexema.
 - Fila.
 - Columna.
 - Caracter de error.

Ejemplo de Interfaz Gráfica



Archivo de entrada

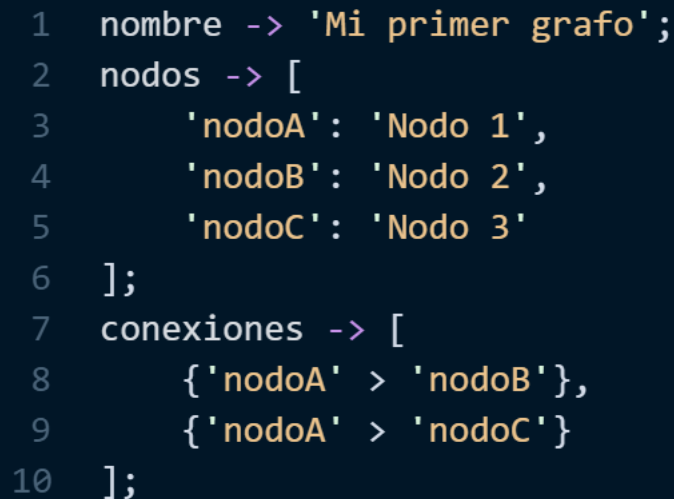
El formato del archivo de entrada para generar un grafo (imagen) será el siguiente:

```
1  nombre -> 'titulo';
2  nodos -> [
3      'nombre_nodo1': 'texto_nodo1',
4      'nombre_nodo2': 'texto_nodo2',
5      'nombre_nodo3': 'texto_nodo3'
6  ];
7  conexiones -> [
8      {'nombre_nodo1' > 'nombre_nodo2'},
9      {'nombre_nodo3' > 'nombre_nodo2'}
10 ]
11
```

Donde **nombre**, **nodos** y **conexiones** son palabras reservadas que dividen los apartados de la imagen, al final de cada una de estas instrucciones vendrá el símbolo de punto y coma (;).

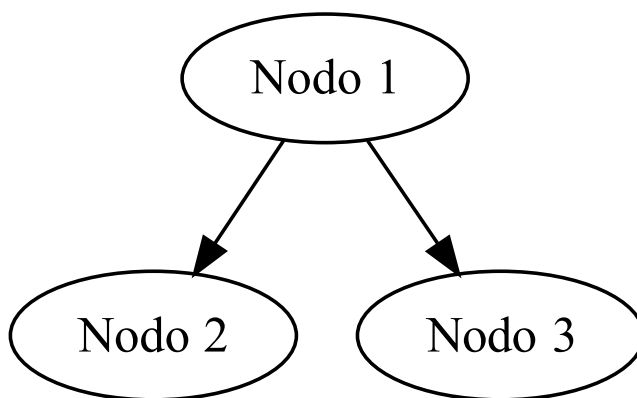
- **nombre** trae consigo el título de la imagen, el cual vendrá en lugar de 'título'. Este título deberá mostrarse en la interfaz gráfica para diferenciar las imágenes unas de otras.
- **nodos** trae una lista de nodos entre corchetes, separados por comas (,), cada nodo viene definido así: un primer string el cual será el nombre del nodo (no tendrá espacios), luego dos puntos y luego otro string que será el texto que deberá mostrar el nodo.
- **conexiones** trae consigo una lista de conexiones entre nodos, esta lista vendrá entre corchetes y separada por comas (,), cada conexión estará encerrada entre llaves y tendrá un primer string que será el nombre del primer nodo, luego el símbolo mayor que (>) y luego otro string que será el nombre del segundo nodo.

Un ejemplo de esta sintaxis es:



```
1  nombre -> 'Mi primer grafo';
2  nodos -> [
3      'nodoA': 'Nodo 1',
4      'nodoB': 'Nodo 2',
5      'nodoC': 'Nodo 3'
6  ];
7  conexiones -> [
8      {'nodoA' > 'nodoB'},
9      {'nodoA' > 'nodoC'}
10 ];
```

Al procesar este archivo con la aplicación se esperaría el siguiente grafo:



Mi primer grafo

Dentro de un archivo de entrada puede venir n cantidad de imágenes, para separar una imagen de otra se utilizarán tres puntos (...) de la siguiente forma:

```
1 nombre -> 'Mi primer grafo';
2 nodos -> [
3     'nodoA': 'Nodo 1',
4     'nodoB': 'Nodo 2',
5     'nodoC': 'Nodo 3'
6 ];
7 conexiones -> [
8     {'nodoA' > 'nodoB'},
9     {'nodoA' > 'nodoC'}
10 ];
11
12 ...
13
14 nombre -> 'Mi segundo grafo';
15 nodos -> [
16     'nodoA': 'Nodo 1',
17     'nodoB': 'Nodo 2',
18     'nodoC': 'Nodo 3'
19 ];
20 conexiones -> [
21     {'nodoA' > 'nodoB'},
22     {'nodoA' > 'nodoC'}
23 ];
```

ENTREGABLES

En UEDI entregar únicamente el link del repositorio privado de GitHub que debe incluir:

- Código fuente de la aplicación.
- Manual técnico, incluyendo su AFD.
- Manual de usuario.

CONSIDERACIONES IMPORTANTES

- Se debe utilizar el lenguaje Python para el desarrollo.
- NO se debe tomar en cuenta errores sintácticos, únicamente se evaluarán errores léxicos.
- El proyecto debe ser desarrollado de forma individual.
- El análisis de los archivos de entrada debe ser carácter por carácter, implementando el AFD que el estudiante elabore, **NO** se permite el uso de Split.
- **NO** se permite la utilización de herramientas que faciliten el análisis léxico, como el módulo re o la herramienta PLY, el estudiante debe codificar el AFD que desarrolle.
- La interfaz gráfica puede ser desarrollada con Tkinter o PyQt5.
- Las imágenes deben ser generadas con Graphviz utilizando el lenguaje DOT.
- Se debe de crear un repositorio privado en GitHub con el siguiente nombre: LFP_Junio_2024 _<carnet>
- Se debe agregar a su auxiliar como colaborador al repositorio, con el usuario **PinD20**.
- Dentro del repositorio se debe crear una carpeta llama Proyecto1 que almacene todo lo relacionado a este proyecto (el repositorio será el mismo para el segundo proyecto, únicamente se creará una carpeta llamada Proyecto2).
- No se aceptan entregas vía correo electrónico u otro medio.
- No se puede modificar los archivos de entrada. El proyecto deberá funcionar con los archivos que se disponga para la calificación, sin modificación.
- El estudiante es responsable del horario que elija para calificarse, en caso de no poder presentarse deberá notificar al auxiliar con suficiente anticipación (1 día antes) para ceder su lugar a otro estudiante, en caso contrario el estudiante solo obtendrá el 80% de su nota obtenida.
- No se dará prórroga para la entrega del proyecto.

- **COPIA PARCIAL O TOTAL DE LA PRÁCTICA TENDRÁ UNA NOTA DE 0 PUNTOS, Y SE NOTIFICARÁ A LA ESCUELA DE SISTEMAS PARA QUE SE APLIQUEN LAS SANCIONES CORRESPONDIENTES.**
- En el caso de no cumplir con alguna de las indicaciones antes mencionadas, NO se tendrá derecho a calificación; por lo cual, se tendrá una nota de cero puntos.
- No se tomarán en cuenta commits realizados luego de la fecha y hora de entrega establecidos.
- Fecha de entrega: **13 de junio de 2024 antes de las 23:59**, no se recibirán entregas después de la fecha y hora establecida.