

UNIVERSIDAD DE SAN CARLOS DE GUATEMALA
FACULTAD DE INGENIERÍA
ESCUELA DE CIENCIAS Y SISTEMAS
LENGUAJES FORMALES Y DE PROGRAMACIÓN
PRIMER SEMESTRE 2024



PROYECTO 2

| Sección | Catedrático | Tutor Académico |
|---------|--------------------------------------|-----------------------------------|
| A+ | Ing. Otto Amílcar Rodríguez Acosta | Gerson Rubén Quiroa del Cid |
| A- | Inga. Vivian Damaris Campos González | Enrique Alejandro Pinula Quiñonez |
| B+ | Ing. David Estuardo Morales Ajcot | Francisco Magdiel Asicono Mateo |
| B- | Inga. Zulma Karina Aguirre Ordoñez | Elder Anibal Pum Rojas |

Contenido

| | |
|--|-----------|
| 1. OBJETIVOS | 3 |
| 1.1. OBJETIVO GENERAL | 3 |
| 1.2. OBJETIVOS ESPECÍFICOS | 3 |
| 2. DESCRIPCIÓN | 3 |
| 3. CARACTERÍSTICA DE LA SOLUCIÓN | 3 |
| 4. DISEÑO DE LA INTERFAZ | 4 |
| 4.1. Área de código | 4 |
| 4.2. Menú Archivo..... | 5 |
| 4.3. Menú Análisis | 5 |
| 4.4. Menú Tokens..... | 6 |
| 4.5. Área de errores | 6 |
| 4.6. Símbolos para ignorar | 6 |
| 4.7. Sentencias para utilizar y sus descripciones | 6 |
| 4.8. Crear funciones | 7 |
| 4.9. Transformación a lenguaje Nosql para MongoDB | 8 |
| 5. ENTREGA | 9 |
| 5.1. Documentación | 9 |
| 5.2. Código Fuente | 10 |
| 6. CONSIDERACIONES IMPORTANTES | 10 |

1. OBJETIVOS

1.1. OBJETIVO GENERAL

Combinar los conocimientos adquiridos en el curso y en los otros cursos de sistemas, para crear un compilador que traduzca el lenguaje especificado y lo transforme en Sentencias de Bases de Datos No Relacionales.

1.2. OBJETIVOS ESPECÍFICOS

- Crear una herramienta que permita el diseño de sentencias de base de datos no relacionales de una forma sencilla para el usuario.
- Diseñar y construir un compilador que permita compilar archivos de entrada y visualizar el resultado en un entorno externo.
- Desarrollar la habilidad del estudiante para elaborar proyectos en base a una adecuada planificación para que aprendan la manera en la que tienen que trabajar.

2. DESCRIPCIÓN

El proyecto consiste en la elaboración de una herramienta que permita el diseño y creación de sentencias de bases de datos no relacionales de una forma sencilla. La aplicación tendrá un área de edición de código y un área de visualización de la sentencia final generada.

Cuando ya se cuente con las sentencias creadas inicialmente, se procederá a realizar la compilación respectiva lo que generará las sentencias de MongoDB que serán mostradas en el espacio de resultados que posteriormente se podrán aplicar a un entorno adecuado a MongoDB.

3. CARACTERÍSTICA DE LA SOLUCIÓN

Para poder responder a las necesidades expuestas anteriormente, se ha pensado en el desarrollo de una aplicación en lenguaje Python que permitirá la creación de las sentencias básicas para ejecutar código de MongoDB.

Con la ayuda de métodos como: Árbol y parser descendente de llamadas recursivas, se deberá implementar una solución que reconozca archivos de texto que contendrán la definición de las sentencias que se usaran en MongoDB, así como sus características particulares de cada sentencia.

La aplicación deberá mostrar los errores que puedan existir en el archivo de entrada que se esté analizando, se deben visualizar de manera agradable y con la suficiente información para saber dónde ocurre el error. Al no encontrarse errores se procede a traducir lo que el archivo de entrada requiere que se haga.

En cualquier momento se pueden colocar comentarios dentro de cada bloque de código del programa, estos son de la manera cómo funciona en las bases de datos

Comentario de una línea:

```
--- comentario
```

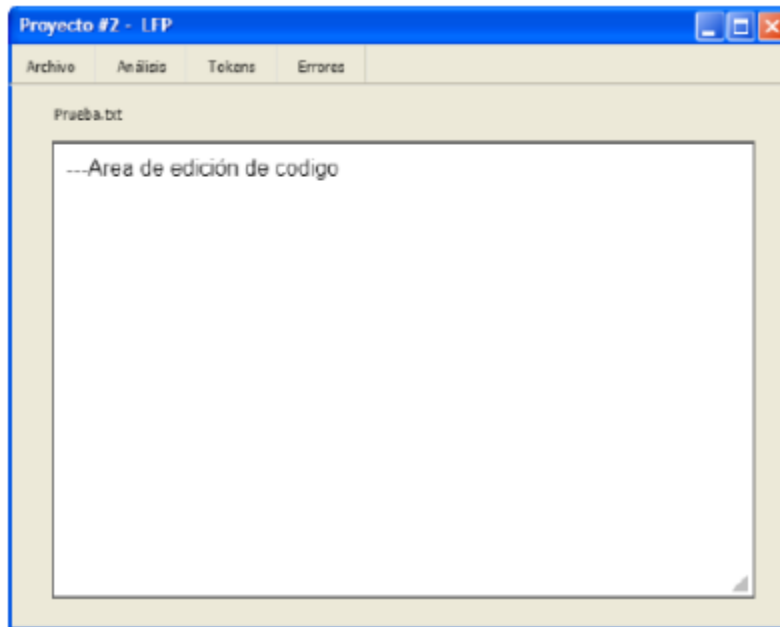
Comentario de varias líneas:

```
/*  
Comentario de  
varias líneas  
*/
```

4. DISEÑO DE LA INTERFAZ

4.1. Área de código

El área de código será un editor de texto en el que podremos cargar archivos y modificarlos. Las características del editor son las siguientes



4.2. Menú Archivo

1. Nuevo: Limpia el área de edición de código, en la cual el usuario puede editar sus sentencias. Si existe un archivo abierto deberá preguntar si desea guardar los cambios antes de limpiar el editor. Esta opción debe preguntar el nombre del archivo y el path donde se guardará.
2. Abrir: Permite abrir un archivo ya creado previamente que contiene las sentencias que generar los comandos de MongoDB. Cuando se cargue el archivo se podrá editar en el área de código.
3. Guardar: Permite guardar el código que se está escribiendo actualmente.
4. Guardar Como: Esta opción permite guardar el código de las sentencias que se está editando con otro nombre.
5. Salir: Con esta opción se cierra la aplicación.

4.3. Menú Análisis

Generar sentencias MongoDB: Esta opción analizará léxica y sintácticamente el contenido que este en el área de código, si existieran errores se mostraran TODOS los errores encontrados tanto léxicos como sintácticos que se encontraron, estos errores se mostrarán en el área de errores. De no existir errores se debe crear y mostrar las sentencias finales que estarán en sintaxis para MongoDB.

4.4. Menú Tokens

Ver Tokens: Mostrará una tabla en la cual estarán listados todos los tokens que se reconocieron en el archivo de entrada

1. Numero correlativo
2. Token
3. Numero de Token
4. Lexema

4.5. Área de errores

En el área de errores está conformada por una tabla y dentro de esta serán cargados tanto los errores léxicos como los errores sintácticos luego de compilado algún archivo. Para los errores se deben mostrar las siguientes características:

1. Tipo de error, pudiendo ser léxico o sintáctico
2. Línea en la que ocurrió el error
3. Posición del error (Columna)
4. Token o componente léxico que se esperó
5. Descripción

4.6. Símbolos para ignorar

Estos símbolos pueden ser varios consecutivos: Espacios en blanco, tabulaciones, retornos, entre otros.

4.7. Sentencias para utilizar y sus descripciones

| TipoFuncion | Descripción |
|-------------------|---------------------------|
| CrearBD | Crea una base de datos |
| EliminarBD | Elimina una Base de datos |
| CrearColeccion | Crea una colección |
| EliminarColeccion | Elimina una colección |
| InsertarUnico | Inserta registro simple |
| ActualizarUnico | Actualiza registro simple |
| EliminarUnico | Elimina registro simple |

| | |
|-------------|-----------------------------|
| BuscarTodo | Busca todos los registros |
| BuscarUnico | Busca solamente un registro |

4.8. Crear funciones

Esta es la forma que pueden crearse todos los comandos que se usaran para ejecutar en MongoDB.

Para definirlo lo hacemos de la siguiente forma:

Funcion Nombre = nueva Funcion("identificador");

Funcion Nombre = nueva Funcion();

Funcion Nombre = nueva Funcion("", "JSON");

Donde:

Funcion = Es el tipo de función

Nombre = Es la identificación del comando

JSON = es una estructura JSON formada dependiendo el comando

Ejemplo:

```
CrearBD ejemplo = nueva CrearBD();
```

```
EliminarBD elimina = nueva EliminarBD();
```

```
CrearColeccion colec = nueva CrearColeccion("NombreColeccion");
```

```
EliminarColeccion eliminacolec = nueva  
EliminarColeccion("NombreColeccion");
```

```
InsertarUnico insertadoc = nueva InsertarUnico("NombreColeccion", "  
{
```

```
  "nombre": "Obra Literaria",
```

```

        "autor": "Jorge Luis"
    }
    ");

ActualizarUnico actualizadoc = nueva ActualizarUnico("NombreColeccion", "
{
    "nombre": "Obra Literaria"
},
{
    $set: {"autor": "Mario Vargas"}
}
");

EliminarUnico eliminadoc = nueva EliminarUnico("NombreColeccion", "
{
    "nombre": "Obra Literaria"
}
");

BuscarTodo todo = nueva BuscarTodo ("NombreColeccion");
BuscarUnico todo = nueva BuscarUnico ("NombreColeccion");

```

4.9. Transformación a lenguaje Nosql para MongoDB

Al tener el archivo de entrada compilados, se deben obtener los comandos que se ejecutaran en MongoDB, estos en la carpeta indicada, el nombre de los archivos de salida es el mismo que el del archivo de entrada.

La especificación para la generación de estos archivos es la siguiente:

| TipoFuncion | Funcion MongoDB | Salida Final |
|-------------|-----------------|-------------------------|
| CrearBD | use | use('nombreBaseDatos'); |
| EliminarBD | dropDataBase | db.dropDatabase(); |

| | | |
|-------------------|------------------|--|
| CrearColeccion | createCollection | db.createCollection('nombreColeccion'); |
| EliminarColeccion | dropCollection | db.nombreColeccion.drop(); |
| InsertarUnico | insertOne | db.nombreColeccion.insertOne(ARCHIVOJSON); |
| ActualizarUnico | updateOne | db.nombreColeccion.updateOne(ARCHIVOJSON); |
| EliminarUnico | deleteOne | db.nombreColeccion.deleteOne(ARCHIVOJSON); |
| BuscarTodo | find | db.nombreColeccion.find(); |
| BuscarUnico | findOne | db.nombreColeccion.findOne() |

5. ENTREGA

5.1. Documentación

1. Manual de usuario: Descripción general del uso de la aplicación (formato MD).
2. Manual técnico (formato MD): además del código bien documentado, se debe de entregar lo siguiente:
 - a. Tabla de tokens, en la cual se debe indicar el patrón (Expresión regular) que define a cada token.
 - b. Método del árbol que utilizó para encontrar el autómata finito determinista, de forma detallada.
 - c. Gráfico de su autómata finito determinista utilizado en el análisis léxico.
 - d. Gramática libre de contexto utilizada para el análisis sintáctico.
 - e. Los incisos anteriores deben coincidir con el código fuente programado de su sistema.

5.2. Código Fuente

1. Enlace hacia su repositorio de Github, en él se debe de ver cómo fueron avanzando en el tiempo por medio de los commits. Favor de realizar commits claros y ordenados de lo que hicieron en cada commit.
2. Nombre del repositorio: LFP-Proyecto2-<Carné>
3. Agregar al repositorio al auxiliar de la sección correspondiente.

6. CONSIDERACIONES IMPORTANTES

- El proyecto se debe desarrollar de forma individual.
- Esta práctica se deberá desarrollar utilizando Python.
- El escáner y el parser deberán programarse en Python, no se pueden usar librerías existentes, debe coincidir con el análisis y diseño del AFD del estudiante.
- No se pueden utilizar clases propias de Python para buscar cadenas mediante expresiones regulares, lectura de xml, ply, o librerías que realicen el parser o el análisis.
- La entrega se realizará en la plataforma UEDI. Todos los archivos solicitados deberán ser entregados en un repositorio de GitHub identificado como se mencionó anteriormente, el estudiante es responsable de verificar que dentro del repositorio se encuentren todos los archivos necesarios para su calificación.
- La calificación deberá ser en línea y se estará grabando la reunión para tener un respaldo de la forma en que se procedió.
- No se debe cambiar el código fuente o los archivos de entrada en la calificación.
- No se permite en el momento de la calificación la participación de otras personas, solamente el estudiante que se le califica y el tutor.
- No se dará prórroga para la entrega del proyecto.
- Copia parcial o total del proyecto 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 calificará el proyecto; por lo cual, se tendrá una nota de cero puntos.

Fecha de entrega: 30 de abril de 2024 antes de las 23:59