UNIVERSIDAD DE SAN CARLOS DE GUATEMALA
FACULTAD DE INGENIERÍA
ESCUELA DE CIENCIAS Y SISTEMAS
LENGUAJES FORMALES Y DE PROGRAMACIÓN
SEGUNDO SEMESTRE 2,021

Sección	Catedrático	Tuto académico
A-	Inga. Vivian Damaris Campos González	César David Juárez González
A+	Ing. Otto Amílcar Rodríguez Acosta	Fernando Feliciano Chajón del Cid
B-	Inga. Zulma Karina Aguirre Ordoñez	Douglas Omar Arreola Martínez
B+	Ing. David Estuardo Morales Ajcot	Bryan Gustavo López Echeverría

Proyecto 2

Objetivos

- Que el estudiante implemente una solución de software implementando los conceptos vistos en clase y laboratorio.
- Que el estudiante implemente un analizador sintáctico utilizando los conceptos de gramáticas independientes de contexto y árboles de derivación.
- Introducir al estudiante a la ejecución de instrucciones en un lenguaje de programación .

Descripción

Se le solicita a usted como estudiante de ingeniería en ciencias y sistemas una solución de software que permita analizar y realizar reportes de datos para la toma de decisiones futuras y que además se pueda aplicar a cualquier tipo de negocio, generando información interesante para pequeñas empresas. Esta solución debe tener como entrada un lenguaje con cierta estructura por medio de un archivo con extensión “.lfp”.

DESCRIPCIÓN DEL LENGUAJE

Importación de Datos:

La importación de datos, los cuáles se utilizarán más adelante en los reportes viene declarada en dos partes:

Sección de Claves: En esta sección 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.

```
Claves = [
    "clave_1", "clave_2", "clave_3", "clave_4"
]
```

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

Sección de Registros: En esta sección 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.

```
Registros = [
    {valor1, valor2, valor3, valor4}
    {valor1, valor2, valor3, valor4}
    {valor1, valor2, valor3, valor4}
    {valor1, valor2, valor3, valor4}
]
```

```
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:

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

```
# Comentarios
```

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

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

Instrucciones de Reportería:

- imprimir(cadena): Imprime por consola el valor dado por la cadena.

```
imprimir("Reporte de ");
imprimir("Abarrotería");
>>> Reporte de Abarrotería
```

- imprimirln(cadena):

```
imprimirln("Reporte de ");
imprimirln("Abarrotería");
>>> Reporte de Abarrotería
>>> Abarrotería
```

- conteo(): Imprime por consola la cantidad de registros en el arreglo de registros.

```
conteo();
>>> 46
```

- promedio(“campo”): Imprime por consola el promedio del campo dado.

```
promedio("stock");
>>> 6.25
```

- contarsi(“campo”, valor): Imprime por consola la cantidad de registros en la que el campo dado sea igual al valor dado.

```
contarsi("stock", 0);
>>> 0

contarsi("stock", 1);
>>> 18

contarsi("stock", 2);
>>> 7
```

- datos(): Imprime por consola los registros leídos.

```
datos();
>>> codigo  producto  precio_compra precio_venta stock
>>> 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
```

- sumar(“campo”): Imprime en consola la suma todos los valores del campo dado.

```
sumar("stock");
>>> 25
```

- max(“campo”): Encuentra el valor máximo del campo dado.

```
max("precio_venta");
>>> 20.00
```

- min(“campo”): Encuentra el valor mínimo del campo dado.

```
min("precio_compra");
>>> 10.50
```

- exportarReporte(“titulo”): Genera un archivo html con una tabla en donde se encuentren los registros leídos y con el título como parámetro.

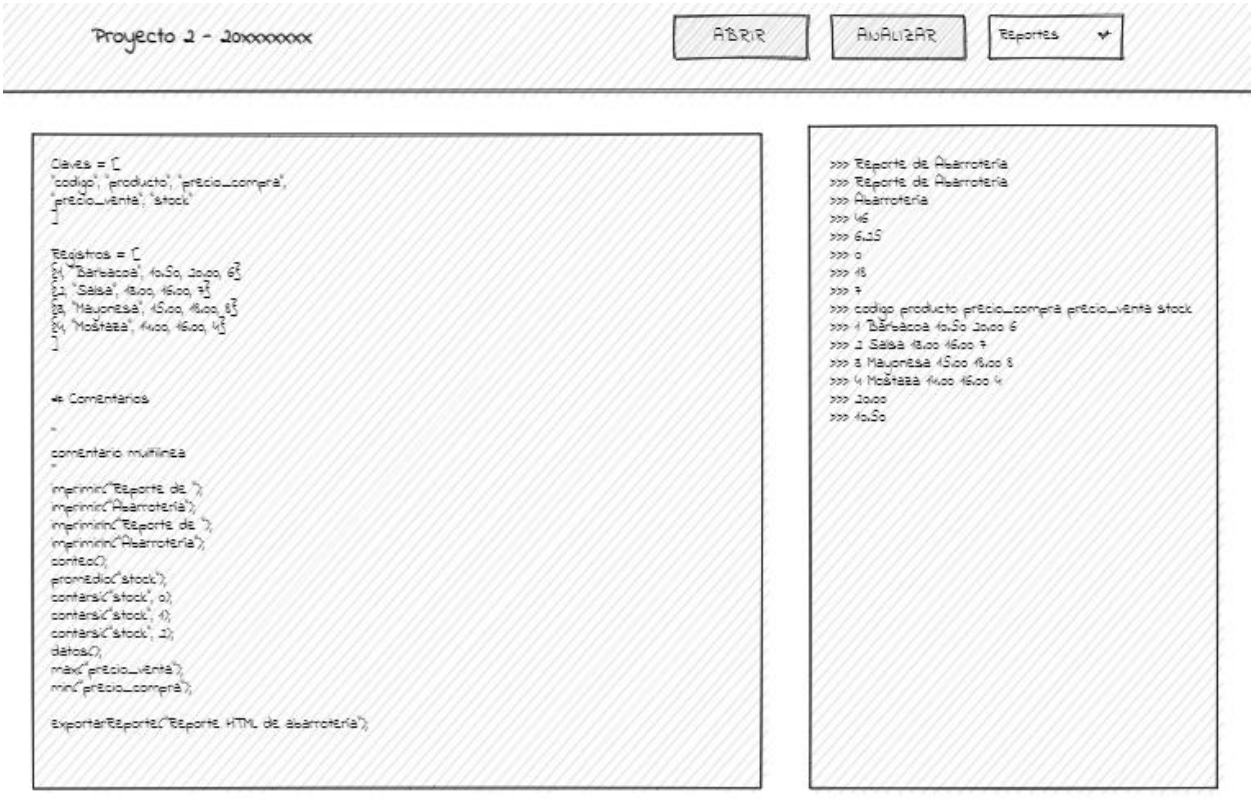
```
exportarReporte("Reporte HTML de abarrotería");
```

Reporte HTML de abarrotería				
código	producto	precio_compra	precio_venta	stock
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

Componentes de interfaz gráfica.

La aplicación cuenta con una interfaz gráfica que posee las siguientes características:

- Cargar archivo: Un botón que al presionarlo permita cargar el archivo con extensión lfp.
- Área de texto: Debe tener un área donde se pueda visualizar y modificar el código lfp.
- Analizar archivo: Un botón que analice el código lfp.
- Consola: Un área de texto que no se pueda editar, solamente visualizar texto generado por las instrucciones dadas por el lenguaje.
- Menú Reportes: Un menú que pueda generar los siguientes reportes:
    - Reporte de Tokens
    - Reporte de Errores
    - Árbol de derivación



Reportes

Se deben generar en formato html los siguientes reportes.

1. Reporte de errores: Se debe generar una tabla con todos los errores léxicos y sintácticos que se encontraron, indicando el caracter o token leído, fila y columna.
2. Reporte de tokens: Se debe generar una tabla con todos los tokens analizados indicando el tipo de token, lexema, fila y columna del token leído.
3. Árbol de derivación generado en la lectura del código fuente utilizando graphviz.

Entregables

- Manual de usuario.
- Manual técnico: Debe incluir expresiones regulares, método del árbol y AFD para cada token así como gramática independiente del contexto utilizado en el analizador sintáctico.
- Código fuente.

Consideraciones importantes

- La práctica debe de desarrollarse individualmente.
- Se debe de utilizar el lenguaje de programación Python
- La entrega se realizará en la plataforma UEDI. Todos los archivos solicitados deberán ser entregados en un archivo comprimido zip con el siguiente nombre: [LFP]Proyecto2_Carnet.zip. Tomar en cuenta que el único medio de entrega es la plataforma UEDI.
- La calificación se realizará en línea, esto para que quede constancia de la forma en que se calificó y como soporte en la toma de decisiones en reclamos por parte del alumno si se presenta el caso y se calificará desde lo entregado.
- La calificación es personal con una duración máxima de 30 minutos, en el horario posteriormente convenido.
- El estudiante es responsable del horario que elija para calificarse, en caso de no poder presentarse deberá notificar al auxiliar con suficiente anticipación (2 días antes) para ceder su lugar a otro estudiante, en caso contrario el estudiante solo obtendrá el 80% de su nota obtenida.
- COPIA PARCIAL O TOTAL DEL PROYECTO TENDRÁ UNA NOTA DE 0 PUNTOS, Y SE NOTIFICARÁ AL CATEDRÁTICO DEL CURSO Y POSTERIORMENTE SI SE REQUIERE A LA ESCUELA DE SISTEMAS PARA QUE SE APLIQUEN LAS SANCIONES CORRESPONDIENTES.
- Cualquier librería que se desee usar debe ser consultada previamente con los auxiliares.
- **Fecha de entrega del proyecto: 28 de octubre de 2021 antes de las 23:59 horas**