



PROCESADORES DE LENGUAJE

Ingeniería Informática
Especialidad de Computación
Tercer curso
Segundo cuatrimestre



Departamento de Informática y Análisis Numérico
Escuela Politécnica Superior de Córdoba
Universidad de Córdoba

Curso académico 2023 - 2024

Hoja de ejercicios de FLEX

Observación

- Se deberá proporcionar un fichero *makefile* y un fichero de entrada con datos para probar el analizador léxico desarrollado en cada ejercicio.
- En todos los ejercicios,
 - se debe comprobar si el número de argumentos es correcto,
 - si el fichero de entrada existe o no
 - y, de forma opcional, enviar los resultados a un fichero de salida, que no debe existir antes.
 - *programa.exe entrada.txt salida.txt*
 - *programa.exe entrada.txt*

1. Direcciones de correo electrónico

- Codifica un analizador léxico denominado **correo.l** que permita reconocer direcciones de correo electrónico que deben tener la siguiente estructura
 - *usuario@entidad.ext*
- donde
 - *usuario*:
 - debe ser una cadena de caracteres compuesta por letras, números, el punto “.” o el símbolo “_”,
 - pero deberá empezar por una letra
 - y el símbolo “_” no podrá aparecer dos veces seguidas ni al final.
 - *entidad*: representa la organización, asociación o empresa y debe estar compuesta solamente por letras.
 - *ext*: representa la extensión que debe estar compuesta por dos o tres letras.
- Por ejemplo
 - *i32gavap@uco.es*

- carmen.lara@empresa.com
 - pablo.luque_19@ayuda.org
- Ejemplos no válidos
 - 1jefe@ejemplo.es
 - error--1@ejemplo.es1
 - error2-@ejemplo.12
- El programa deberá
 - contar el número de direcciones de correo electrónico correctas que haya reconocido
 - y mostrar un mensaje de error cuando no reconozca una dirección de correo electrónico correcta.

2. Fichero con figuras geométricas

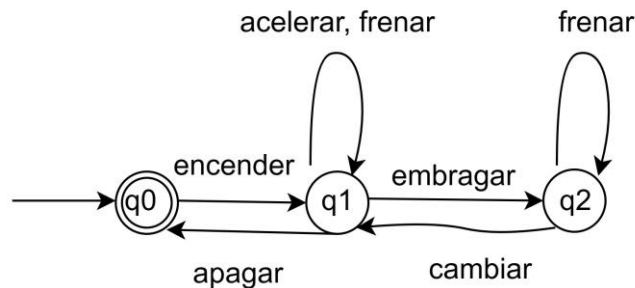
- Codifica un analizador léxico denominado **figura.l** que permita reconocer figuras geométricas, como, por ejemplo, las siguientes:
 - triángulo (0.0 0.0) (1.0 0.0) (1.0 1.0)
 - Se indican las coordenadas de los tres puntos del triángulo
 - cuadrado (0.0 0.0) (1.0 1.0)
 - Se indican las coordenadas de dos vértices del cuadrado que ocupan posiciones diagonalmente opuestas.
 - círculo (0.0 0.0) 1.0
 - Se indican las coordenadas del centro y el radio.
 - Nota:
 - El radio y las coordenadas de los puntos serán números reales, que podrán tener notación científica.
 - Las palabras “triángulo” y “círculo” se podrán escribir con y sin acento.
- El programa deberá funcionar de dos formas diferentes
 - **Modo interactivo**
 - Los datos de las figuras se introducirán desde el teclado.
 - Al reconocer una figura, deberá calcular su perímetro y su área.
 - Se mostrará un mensaje de error cuando no reconozca una figura geométrica correcta.
 - El programa finalizará con el carácter fin de fichero o al teclear “#” al principio de la línea.
 - Al finalizar, el analizador deberá mostrar el número de figuras que haya reconocido de cada tipo.
 - **Modo no interactivo:**
 - Se utilizará un fichero de entrada con los datos de las figuras geométricas
 - Por ejemplo

triángulo (0.0 0.0) (1.0 0.0) (1.0 1.0)
 cuadrado (0.0 0.0) (1.0 1.0)
 círculo (0.0 0.0) 1.0

- Al reconocer una figura, deberá calcular su perímetro y su área.
- Se mostrará un mensaje de error cuando no reconozca una figura geométrica correcta.
- Al finalizar, el analizador deberá mostrar el número de figuras que haya reconocido de cada tipo.

3. Simulación de un autómata finito determinista

- Utiliza los **estados de flex** para codificar un analizador léxico denominada **automata.l** que permita simular el funcionamiento del siguiente autómata finito determinista.



- Este autómata finito determinista simula el funcionamiento básico de un automóvil.
- El analizador se podrá usar de forma interactiva o usando ficheros de entrada o salida.

4. Cola de impresión de una impresora

- Un fichero contiene la siguiente información sobre los documentos enviados a una impresora:
 - DD-MM-AAAA HH:MM documento
- Por ejemplo


```
09-03-2023 09:00 fichero1.pdf
10-03-2023 09:15 fichero2.docx
...
```
- Codifica un analizador léxico denominado **impresora.l** que reciba un fichero con el listado de documentos enviados a la impresora y que muestre
 - el número de documentos enviados a la impresora
 - el número medio de documentos enviados a la impresora cada día.

- El analizador se podrá usar de forma interactiva o usando ficheros de entrada o salida.
- Observaciones
 - **Nota:**
 - **Este ejercicio se debe realizar usando estados de flex.**
 - Se deberá comprobar si la fecha y las horas son correctas. En caso contrario, se mostrará un mensaje por pantalla. En particular, se deberá comprobar si el año es bisiesto.
 - También se deberá comprobar que la hora de entrada sea anterior a la hora de salida de cada fecha.

5. Analizador léxico de pseudocódigo

- Codifica un analizador léxico que permita reconocer los componentes léxicos de un programa escrito en pseudocódigo.
- **Palabras reservadas**
 - *inicio, fin, leer, escribir, si, entonces, si_no, fin_si, mientras, hacer, fin_mientras, repetir, hasta_que, para, desde, hasta, paso, fin_para*
 - No se distinguirá entre mayúsculas ni minúsculas.
 - Las palabras reservadas no se podrán utilizar como identificadores.
- **Identificador**
 - Características
 - Estarán compuestos por una serie de letras, dígitos y el subrayado;
 - Deben comenzar por una letra.
 - No podrán acabar con el símbolo de subrayado, ni tener dos subrayados consecutivos.
 - No se distinguirá entre mayúsculas ni minúsculas.
 - Ejemplos
 - Identificadores válidos:
dato, dato_1, dato_1_a
 - Identificadores **no** válidos:
dato, dato, dato__1
- **Número**
 - Se utilizarán números enteros, reales de punto fijo y reales con notación científica.
 - Todos ellos serán tratados conjuntamente como números.
- **Cadena**
 - Estará compuesta por una serie de caracteres delimitados por comillas simples:

‘Ejemplo de cadena’

- Deberá permitir la inclusión de la comilla simple utilizando la barra (\):

‘Ejemplo de cadena con \' comillas\' simples’.

- **Nota:**

- Las comillas exteriores no formarán parte de la cadena.

- **Operador de asignación**

- ASIGNACIÓN: =
- ASIGNACIÓN SUMA: +=
- ASIGNACIÓN RESTA: -=
- ASIGNACIÓN PRODUCTO: *=
- ASIGNACIÓN DIVISION: /=

- **Operadores aritméticos:**

- SUMA: +
- INCREMENTO: ++
- RESTA: -
- DECREMENTO: --
- PRODUCTO: *
- DIVISIÓN: /
- DIVISIÓN ENTERA: //
- MÓDULO: %
- POTENCIA: **

- **Operador alfanumérico:**

- CONCATENACIÓN: ||

- **Operadores relacionales de números y cadenas:**

- MENOR_QUE: <
- MENOR_IGUAL_QUE: <=
- MAYOR_QUE: >
- MAYOR_IGUAL_QUE: >=
- IGUAL: ==
- DISTINTO: <>

- **Operadores lógicos:**

- DISYUNCIÓN_LÓGICA: #o
- CONJUNCIÓN_LÓGICA: #y
- NEGACIÓN_LÓGICA: #no
- Por ejemplo:

(A >= 0) #y #no (control <> ‘stop’)

- **Comentarios**

- De varias líneas: delimitados por << y >>

<< *ejemplo*
 de comentario
 de tres líneas >>

- De una línea:
 - Todo lo que siga a los dos caracteres “!!” hasta el final de la línea.

!! *ejemplo de comentario de una línea*

- **Otros componentes léxicos**
 - FIN_SENTENCIA: ;
 - Paréntesis
 - Izquierdo: (
 - Derecho:)
- **Control de errores**
 - El intérprete deberá controlar toda clase de errores:
 - Identificador mal escrito.
 - Números mal escritos.
 - Utilización de símbolos no permitidos.
 - Etc.
- **Prueba**
 - Se deberá comprobar el funcionamiento del analizador léxico usando tres ficheros:
 - Fichero denominado Newton.txt
 - ejemplo_1.txt: fichero original **sin** errores.
 - ejemplo_2.txt: fichero original **con** errores.
 - **Importante**
 - Se valorará que los ejemplos propuestos tengan un código de un algoritmo o tarea interesante.