COMPILADORES
LABORATORIO 1

TITULO: ANALIZADOR LEXICO DE UN SUBCONJUNTO DE LENGUAJE DE PROGRAMACIÓN C.
LENGUAJE: Lex (Flex) y C Bajo Linux.
FECHA DE INICIO: Agosto 29 de 2022
FECHA DE ENTREGA: Septiembre 29 de 2022
NOMBRE PROGRAMA: LAB01_Apellido1_Apellido2 (Fuente Lex).
LAB01_Apellido1_Apellido2 (Fuente C).
LAB01_Apellido1_Apellido2 (Ejecutable)
ENTREGABLE: Archivo comprimido con extensión .zip (únicamente) con nombre LAB01_Apellido1_Apellido2_Apellido2.zip conteniendo todos los archivos solicitados.
INTEGRANTES: 3 ESTUDIANTES POR GRUPO, y pueden ser de cursos distintos.

Gramática: Los siguientes son los tokens que hacen parte del conjunto a tener en cuenta:
Inicio({),si-sino-fsi,para-fpara,mq-fmq,hh-fhh,dd-fdd (dependiendo de),leer,escribir,fin
Los operadores aritméticos: op-mult (*), op-sum(+), op-sust(-), op-div(/), op-mod(%)
Asignación: op-asign(=)
Tener en cuenta la sintaxis de una instrucción interna en cuanto su terminación, es decir, si termina con punto, coma o punto y coma o con nada.
Los siguientes símbolos: parent-a (( ), parent-c ( )), coma (,), donde sean necesarios.
Los siguientes tokens: Constantes enteras, reales, cadenas (las cadenas están entre comillas dobles) y caracteres (los caracteres se encierran entre comillas simples).
Los componentes léxicos de las variables: Identificadores.
Los componentes léxicos de comparación: Igual (==), Diferente(!=) Menor-igual (<=), Mayor-igual (>=), Mayor (>) y Menor (<).
Los operadores booleanos: And (&&), Or (||), No (!).
Se deben considerar los comentarios dentro de cada uno de los programas.
Se consideran las siguientes declaraciones de variables: int, float, char. Esta son palabras claves del lenguaje de programación C.
No se debe considerar:
Manejo de arreglos de cualquier dimensión.
Manejo de cadenas
Manejo de funciones o subrutinas
Manejo de funciones predefinidas.

Entrada: Un archivo llamado **Prueba** con cualquier extensión, el cual contendrá un programa con las indicaciones anteriores y basado en el lenguaje de programación C.

Salida: Generación de un archivo de salida llamado **Salida.txt**, el cual contendrá todos los componentes léxicos que se pueden obtener del archivo programa de entrada. Los componentes léxicos correspondientes a variables deben presentar en forma de tabla al final. También en este archivo se debe generar un listado de errores léxicos, si los hay..

Consideraciones adicionales:

1. Generar dentro de la tabla de identificadores, un nombre único para las variables, aunque aparezcan más de una.
2. Las palabras claves no son identificadores, pero se resaltan como componentes léxicos.
3. Cualquier símbolo diferente de los establecidos es un error y se debe colocar también en el archivo de salida.
4. Si hay error léxico en algún punto del archivo, se debe continuar el análisis hasta el final.
5. La entrada para el programa que se genere debe ser un archivo con un programa en C y se debe generar un archivo de salida como se describió anteriormente.
6. Debe entregar un **manual de uso del programa**, de no más de dos (2) páginas donde describa cómo se describa cómo se compila y cómo se ejecuta el programa.
7. Todo el trabajo deben desarrollarse para el sistema operativo Linux Ubuntu.
8. El lenguaje C donde deben trabajar deben tomarlo del nativo de Linux.
9. Al ejecutar su programa final debe ser así: **programa-ejecutable archivo-entrada. No utilice ningún símbolo entre ambos.**

Ejemplo:

Dado el siguiente programa en el archivo de entrada **prueba.c**

```
main (void)
{ int i;
  int j;
  char c;
  char cadena
  float z;
  z=14.9e-8;
  z=12.9;
  cadena="Hola";
  scanf ("%d",i);
  i=i*2;
  printf ("El doble es %d",i);
}
```

Este es el archivo de salida en **salida.txt:**

MAIN
Parent-a= (**VOID Parent-c=**)
Inicio={
  INT Id= i **Punto-coma=**;
  INT Id= j **Punto-coma=**;
  CHAR Id= c **Punto-coma=**;
  CHAR Id= cadena
  FLOAT Id= z **Punto-coma=**;
  Id=z **Op-asig=** = **Cte real=** 14.9e-8 **Punto-coma=**;
  Id=z **Op-asig=** = **Cte real=** 12.9 **Punto-coma=**;
  Id=cadena **Op-asig=** = **Cte cadena=** "Hola" **Punto-coma=**;

SCANF Parent-a= (**Cte cadena=** "%d", **Id=** i **Parent-c=**) **Punto-coma=**;
Id= i **Op-asig=** = **Id=** i **Op-mult=** * **Cte entera=** 2 **Punto-coma=**;
PRINTF Parent-a= (**Cte cadena=** "El doble es %d",**Id=** i **Parent-c=**) **Punto-coma=**;
Fin=}

TABLA DE IDENTIFICADORES
Hay 5 identificadores
Id= i; Id= j; Id= c; Id= cadena; Id= z;

Observación: Lo que aparece en negrilla es el componente léxico y lo que sigue es el valor o lexema. Las palabras claves no tienen otro valor de lexema, sino la misma palabra, pero en mayúscula.