Manual de usuario

Analizador lexico con flex

Entorno de ejecución

Para poder utilizar el proyecto es necesario que utilice un sistema operativo basado en linux, para nuestro caso en especifico utilizaremos una distribución de Ubuntu.

Uso

Dependencias

El proyecto hace uso de flex, el cual no viene instalado por defecto en ubuntu y por ende debemos instalarlo antes de continuar. Para esto utilizaremos los siguientes comandos

```
sudo apt-get update
sudo apt-get install flex
```

Compilación a .c

El código fuente de nuestro programa está escrito en flex, con una extensión `.l`, por ende primeros transcribirlo a código del lenguaje `c` con una extensión `.c`. Para esto ejecutaremos el siguiente comando

```
flex LAB01_Cuesta_Salazar_Evilla.l
```

Creación del ejecutable

Finalmente debemos generar nuestro archivo ejecutable, compilando el código con el siguiente comando.

```
cc lex.yy.c -lfl -o LAB01_Cuesta_Salazar_Evilla
```

Ejecución

Una vez creado el archivo ejecutable podemos pasar usarlo mediante el siguiente comando.

```
./LAB01_Cuesta_Salazar_Evilla prueba.c
```

Donde `prueba.c` es el nombre del archivo con código C que vamos a analizar, este puede tomar cualquier otro valor como `archivo.c` `codigo.c` etc...

Nota En caso de no pasar ningun nombre de archivo al ejecutable, este por defecto anulara su ejecución con el siguiente error:

```
Error, debe pasar el nombre del archivo con codigo fuente como parametro asi:
./LAB01_Cuesta_Salazar_Evilla nombreArchivo.c
```

Resultados

El resultado de la ejecución del programa sera almacenado en el archivo `salida.txt`

Ejemplo

Haciendo uso de un archivo de entrada llamado `prueba.c` con el siguiente contenido:

```
 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);
}
```

Ejecutaremos el siguiente comando:

```
./a.out prueba.c
```

En un archivo llamado `salida.txt` obtendremos el siguiente resultado:

```
 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_asign= = Cte_real= 14.9e-8 punto_coma= ;

Id= z op_asign= = Cte_real= 12.9 punto_coma= ;

Id= cadena op_asign= = Cte_cadena= "Hola" punto_coma= ;

SCANF  parent_a= ( Cte_cadena= "%d" ,Id= i parent_c= ) punto_coma= ;

Id= i op_asign= = Id= i op_mult= * Cte_entero= 2 punto_coma= ;

PRINTF  parent_a= ( Cte_cadena= "El doble es %d" ,Id= i parent_c= ) punto_coma= ;

Fin= }




Hay 5 identificadores
Id i; Id j; Id c; Id cadena; Id z;
```