Universidad Nacional Autónoma de México

Facultad de Ingeniería

Ingeniería en Computación

Lenguajes Formales y Autómatas

Proyecto Final

Analizador léxico en flex y analizador sintáctico en yacc de las sentencias de control del lenguaje C

Grupo: 05

Hernández Rubio Dana Valeria

Soto Huerta Gustavo Isaac

Ugalde Santos Atzin

Profesora: Ing. Josefina Rosales García

Índice

Objetivo: Realizar el analizador léxico en flex y el analizador sintáctico en yacc de las sentencias de control del lenguaje C.

Desarrollo: Cada equipo analizará el lenguaje de las sentencias de control de forma generalizada. De ello realizará lo siguiente:

➢ ***Lista de Sentencias de Control en C***

Una sentencia de control se refiere a estructuras que controlan el flujo de ejecución del programa. Estas estructuras permiten tomar decisiones o repetir un bloque de código según ciertas condiciones.

- if

```
if (condición) {
        // Código a ejecutar si la condición es verdadera
}
```

- if-else

```
if (condición) {
        // Código a ejecutar si la condición es verdadera
} else {
        // Código a ejecutar si la condición es falsa
}
```

- switch

```
switch (expresión) {
case valor1:
```

```
        // Código a ejecutar si la expresión coincide con valor1
        break;
    case valor2:
        // Código a ejecutar si la expresión coincide con valor2
        break;
        // Otros casos...
    default:
        // Código a ejecutar si no hay coincidencia con ningún caso
}
```

- Bucles (loops)
  - while

```
while (condición) {
    // Código a ejecutar mientras la condición sea verdadera
}
```

  - do-while

```
do {
    // Código a ejecutar al menos una vez, y luego mientras la condición
    sea verdadera
} while (condición);
```

  - for

```
for (inicialización; condición; incremento/decremento) {
```

// Código a ejecutar mientras la condición sea verdadera

}

- break

  break;

- continue

  Salta a la siguiente iteración en un bucle.

  continue;

- goto

  Permite saltar de un lugar a otro en el código de manera no lineal.

  goto etiqueta;

➢ ***Expresiones regulares de cada una de las cadenas***

variable	*{A-Za-z}+(_|{A-Za-z}|{0-9}*)*
número	*{0-9}+\.{0-9}+*
cadena	
if	*if*
if-else	if - else
switch	switch
while	while
do-while	do - while
for	*for*
break	break
continue	continue
goto	goto

➢ ***Gramática de cada una de las sentencias***

- if

G_if(P, T, NT, S)

P:{ INICIO → if (CONDICION) BLOQUE
CONDICION→ EXPRESION COMPARACION EXPRESION
COMPARACION→ operador de asignacion
EXPRESION→ variable | numero
BLOQUE→ { SENTENCIAS }
SENTENCIAS → SENTENCIA |SENTENCIA SENTENCIAS
SENTENCIA → INICIO |
}

NT={INICIO, CONDICION, COMPARACION , EXPRESION , BLOQUE, SENTENCIAS SENTENCIA }
T={if, (,), {, }, variable, asignación, numero}
S={INICIO}

- if-else
- switch
- Bucles (loops)
  - while
  - do-while
  - for

G_for(P, T, NT, S)

P:{ INICIO → for (INTERIOR) { }
INTERIOR → PRIMERO ; SEGUNDO ; TERCERO
PRIMERO → variable asignación número
SEGUNDO → variable comparación número
TERCERO → variable OPERACION
OPERACION → --|++
}

NT={INICIO, INTERIOR, PRIMERO, SEGUNDO, TERCERO, OPERACION INTERIOR}
T={for, (,), {, }, ; variable, asignación, numero, comparación, --, ++}
S={INICIO}

- break
- continue
- goto

➢ ***Obtener la tabla de clases :***

Clases	
0	Palabras Reservadas
1	Operadores de asignación
2	Operadores Lógicos
3	Símbolos Especiales
4	Números
5	Variables
6	Cadenas

➢ ***Obtener la tabla para cada clase:***

Palabras Reservadas

Palabras Reservadas	
0	asm
1	auto
2	break

3	case
4	char
5	const
6	continue
7	default
8	do
9	double
10	else
11	enum
12	extern
13	float
14	for
15	goto
16	if
17	int
18	long
19	register
20	return
21	short
22	signed
23	sizeof
24	static

25	struct
26	switch
27	typedef
28	union
29	unsigned
30	void
31	volatile
32	while

Operadores de asignación

1. == Igual
2. >
   Mayor que
3. <
   Menor que
4. >=
   Mayor que o igual a
5. <=
   Menor que o igual a
6. !=
   Diferente de
7. +=
   Suma Compuesta
8. -=
   Resta Compuesta
9. *=
   Multiplicación Compuesta
10. /=
    División Compuesta
11. %=
    Módulo Compuesto

Operadores de Asignación	
0	==
1	>
2	<
3	>=
4	<=
5	!=
6	+=
7	-=
8	*=
9	/=
10	%=

Operadores Lógicos

1. && AND lógico
2. || OR lógico
3. ! NOT lógico

Operadores Lógicos	
0	&&
1	||
2	!

Operadores Matemáticos

1. +
2. -
3. /
4. %
5. ++
6. --

Símbolos Especiales

1. ,
2. . Acceso a un miembro de una estructura
3. -> Acceso a un miembro de una estructura a través de un puntero
4. * Operador de indirección (puntero)
5. & Operador de dirección
6. ;
7. : Dos puntos (utilizado en etiquetas y casos en switch)
8. (
9.)
10. {
11. }
12. [
13.]

Símbolos Especiales	
0	**,**
1	**.**
2	**->**
3	*****
4	**&**
5	**;**
6	**:**

7	**(**
8	**)**
9	**{**
10	**}**
11	**[**
12	**]**

Números

Números	
0	0
1	1
2	2
3	3
4	4
5	5
6	6
7	7
8	8
9	9

Variables

Variables	
0	m

1	i
2	j
3	var1
4	var2

Cadenas

Cadenas	
0	
1	
2	
3	
4	

Conclusiones:

Hernández Rubio Dana Valeria:

Soto Huerta Gustavo Isaac:

Ugalde Santos Atzin:

Bibliografía:

1. Aitken, P. G., & Jones, B. (1994). *Aprendiendo C en 21 días*. Sams.

2. Deitel, H. M., & Deitel, P. J. (2004). *Cómo programar en C/C ++ y Java*. Pearson Educación.
3. Ammeraal, L. (1991). *C for programmers: A Complete Tutorial Based on the ANSI Standard*.
4. *Mi primer proyecto utilizando Yacc y Lex*. (2020, 1 octubre). Erick Navarro. https://ericknavarro.io/2020/10/01/27-Mi-primer-proyecto-utilizando-Yacc-y-Lex/
5.