Proyecto de la Asignatura de Compiladores

Primera Entrega

Realizado por:

Jairo Alejandro Castrillón Libreros

Juan Camilo Fernández

Presentado a:

Fernando Galindres

Universidad Tecnológica de Pereira

2019

# En que consiste un analizador léxico?

El analizador léxico es el encargado de recibir el código fuente de otro programa y mediante una especificación se obtiene como salida los tokens o componentes léxicos que se encuentran definidos en el código fuente y son reconocidos en el marco de un lenguaje, para el reconocimiento de los tokens en el código fuente nos apoyamos en el uso de expresiones regulares, osea que el documento debe estar tipado de acuerdo al léxico que esta definido en el compilador.

Este proceso que se realiza con el analizador léxico es muy importante para la fase de análisis sintáctico ya que con los tokens obtenidos el analizador sintáctico construye la estructura de datos, un árbol sintáctico.

# Palabras Reservadas

Ya que el compilador debe ser desarrollado para el lenguaje Mini Java tuvimos en cuenta las siguientes palabras reservadas.

* BOOLEAN | boolean
* BREAK | break
* CLASS | class
* CONTINUE | continue
* ELSE | else
* EXTENDS | extends
* FALSE | false
* IF | if
* INT | int
* LENGTH | length
* NEW | new
* NULL | null
* RETURN | return
* STRING | string
* THIS | this
* TRUE | true
* void | VOID
* WHILE | while

# Compositores Lexicos:

* AND
* ASSIGNMENT
* BINARY
* COMMA
* LINECOMMENT
* MULTICOMMENT
* RIGHTSQRBRACKET
* LEFTSQRBRACKET
* DIVISION
* NOTEQUAL
* CONCAT
* EQUAL
* NOT
* HEXADEC
* IDEN
* RIGHTBRACE
* LEFTBRACE
* GREATEREQUAL
* GREATER
* LESSEQUAL
* LESS
* SUBSTRACTION
* UMINUS
* MULTIPLICATION
* NUMBER
* CIENTIFIC
* FLOAT
* OR
* RIGHTPARENT
* LEFTPARENT
* MODULO
* DOT
* SEMICOLON
* ADDITION

# Inconvenientes y problemas al desarrollar la primera entrega

Los inconvenientes que tuvimos al momento de realizar la primera entrega son los siguientes:

1. Definir una expresión regular para el reconocimiento de números hexadecimales ya que estos números están formados por caracteres alfanuméricos.
2. La identificación de identificadores ya que estos no siguen un formato predeterminado.

**Casos de prueba: de** <https://cs.fit.edu/~ryan/cse4251/minijava/>

//

class Test {

int partition(int [] a, int low, int high) {

int pivot = a[low];

int i = low;

int j = high;

int tmp;

while (true) {

while (a[i] < pivot) i = i+1;

while (a[j] > pivot) j = j-1;

if (i >= j) break;

tmp = a[i];

a[i] = a[j];

a[j] = tmp;

i = i + 1;

j = j - 1;

}

return j;

}

void quicksort(int [] a, int low, int high) {

if (low < high) {

int mid = partition(a, low, high);

quicksort(a, low, mid);

quicksort(a, mid+1, high);

}

}

}

class Factorial{

public static void main(String[] a){

System.out.println(new Fac().ComputeFac(10));

}

}

class Fac {

public int ComputeFac(int num){

int num\_aux ;

if (num < 1)

num\_aux = 1 ;

else

num\_aux = num \* (this.ComputeFac(num-1)) ;

return num\_aux ;

}

}