22 de abril de 2019

Juan Alberto Hernández

UNIVERSIDAD POLITÉCNICA DE SAN LUIS POTOSI

Proyecto Analizador lexico – Basic

Analizador léxico para leguaje de programación Basic

Contenido

[Introducción 2](#_Toc6850945)

[Marco conceptual 3](#_Toc6850946)

[Antecedentes 3](#_Toc6850947)

[Justificación 3](#_Toc6850948)

[Planteamiento del problema 3](#_Toc6850949)

[Objetivos 3](#_Toc6850950)

[Marco teórico 3](#_Toc6850951)

[Historia de BASIC 3](#_Toc6850952)

[Principios de BASIC 4](#_Toc6850953)

[Analizador léxico 4](#_Toc6850954)

[Propuesta 4](#_Toc6850955)

[Palabras reservadas 4](#_Toc6850956)

[Diagrama de flujo 5](#_Toc6850957)

# Introducción

El presente proyecto se enfoca al desarrollo de un analizador léxico con el conocimiento adquirido en la materia de teoría computacional, es necesario que los estudiantes tengan conocimientos sobre el bajo nivel al que se procesan los códigos, los compiladores nos ayudan a llevar acabo ese proceso.

Después de que se oprime el botón de compilar se llevan acabo muchos pasos por los que pasa el texto para poder llegar a ejecutar alguna acción.

# Marco conceptual

## Antecedentes

Existen ya desarrollados analizadores que son robustos y cuentan con todas las funciones ya determinadas, pero el propósito de este proyecto es practicar los conocimientos adquiridos en el curso de teoría computacional.

## Justificación

Es un programa que engloba muchos de los temas que se vieron en la clase y es un muy buen ejemplo de actividad integradora para validar que los conocimientos de la materia tengan buenas bases.

## Planteamiento del problema

El problema se basa en que se genere un analizador que pueda verificar la validez de algún código escrito en el leguaje de programación BASIC y que en caso de que no conemos con un código valido que los errores sean señalados

## Objetivos

* General:
  + Generar un código que sea capaz de verificar que un código sea valido
* Específicos:
  + Que el código marque el lugar en el que se generó el error
  + Que el código no se detenga si hay algún error

# Marco teórico

## Historia de BASIC

Los creadores del BASIC fueron John George Kemeny y Thomas Eugene Kurtz en el año 1964. Fue inventado para permitir a los estudiantes escribir programas usando terminales de computador de tiempo compartido. Con el BASIC se quiso hacer un lenguaje de programación mucho más sencillo que los que existían por aquel entonces. Porque los lenguajes que existían requerían tener conocimiento específicos en lenguajes más complejos que eran hechos a medida, principalmente para científicos y matemáticos.

El BASIC es un lenguaje de programación que se creó con fines pedagógicos, era el lenguaje que utilizan las microcomputadoras de los años 80. Actualmente sigue siendo muy conocido y tienen muchísimos dialectos muy diferentes al original. Veamos algo de historia del Basic. En los años 1960, las computadoras tenían un valor elevado y se usaban para tareas específicas, eran mono tarea. Pero luego, permitiendo que algunas empresas pequeñas pudieran permitirse adquirirlas. Las computadoras mejoraron mucho en velocidad, capacidad de procesamiento de datos y también lo hicieron los lenguajes. Aparecieron lenguajes como el FORTRAN, el COBOL y el Basic.

## Principios de BASIC

Los principios que originaron la creación del BASIC eran: que fuese fácil de usar por todos, crear un lenguaje de programación de propósito general, que se le pudiese incorporar características avanzadas y siguiese siendo de fácil uso para los principiantes, ser interactivo, que los mensajes de error fuesen claros, que respondiese rápidamente a los programas pequeños, que no fuese necesario tener conocimiento del hardware de la computadora, y que protegiese al usuario del sistema operativo. Porque por entonces, los sistemas operativos no eran tan interactivos como lo son hoy.

## Analizador léxico

Un analizador léxico o analizador lexicográfico (en inglés scanner) es la primera fase de un compilador consistente en un programa que recibe como entrada el código fuente de otro programa (secuencia de caracteres) y produce una salida compuesta de tokens (componentes léxicos) o símbolos.

# Propuesta

## Palabras reservadas

Básicas: ready , end, new, load, print, save, let

Saltos: goto, gosub

Condicionales: if, case

Ciclos: for, while

Conectores: &, || , ==, +, \*, -, /, <, >,

# Diagrama de flujo

FALSE

TRUE

Mostar mensaje de Fin

Palabra reservada END

FALSE

FALSE

TRUE

Mandar a estado de error la matriz y guardar la linea

FALSE

TRUE

TRUE

TRUE

FALSE

Mostrar descomposición en tokens

Mandar a estado de error la matriz y guardar la línea

Mostrar descomposición en tokens

Carácter especial que cierra

Carácter especial

Palabra reservada valida

Mandar a estado de error la matriz y guardar la linea

Mostrar descomposición en tokens

Mandar a estado de error la matriz y guardar la línea

Mostrar descomposición en tokens

Palabra reservada READY

R

Ejecutar función de analizadorLexico

Abrir archivo con el código