**UNIVERSIDAD PEDAGOGICA Y TECNOLOGICA DE COLOMBIA**

**INGENIERIA DE SISTEMAS Y COMPUTACION**

**LENGUAJES FORMALES – SEMESTRE I 2018**

**PROYECTO 2-50%**

PROYECTO ANALIZADOR SINTACTICO E INTERPRETADOR

Este proyecto consiste en hacer funcionar un **analizador sintáctico** de un lenguaje de programación que nos inventaremos (VARIACION DE BASIC). Una de las primeras verificaciones que un Compilador debe realizar, es validar que el código o algoritmo escrito se ajuste al lenguaje de programación correspondiente a dicho compilador, y así detener o permitir el avance de los demás pasos que un compilador debe realizar para generar finalmente un código ejecutable o programa de ejecución.

Una vez validado sintácticamente el código fuente, y de estar bien escrito, ya se podrá realizar la **interpretación** de dicho código fuente, es decir, ejecutar las ordenes de dicho programa.

Como se indicó desde los primeros días de clase del semestre, el lenguaje de programación a usar será C. Aunque algunos estudiantes quisieran trabajar en lenguaje java. Para ello es de aclarar que proyectos que trabajen según las especificaciones iniciales en Lenguaje C, serán evaluados sobre 5. Para aquellos que decidan trabajar en lenguaje java, serán evaluados sobre 4.

El proyecto tendrá dos partes:

1. Analizador Sintáctico 50% . Entrega y Sustentación. Desde Junio hasta Junio 8

2. Interpretador 50% . Entrega y Sustentación. Desde Junio hasta Junio 15

El lenguaje de programación que nos vamos a inventar hará uso de las siguientes palabras reservadas

|  |  |
| --- | --- |
| [1..0] [1..0][1..0] | Las líneas de código van enumeradas hasta con 3 digitos |
| DIM <var>  AS DOUBLE  AS STRING  DIM <listvar>  AS DOUBLE  AS STRING | Esta palabra define las variables que se van a usar en el programa.  Las líneas de código que usan esta instrucción solo se deben aceptar antes que cualquier otro tipo de línea de código.  Pueden existir multiples líneas con esta instrucción, como también en una misma línea se pueden definir mas de una variable.  Solo se manejaran dos tipos de variables, las numéricas (DOUBLE) y las alfanuméricas (STRING)  Ej:  10 DIM VAR1, VAR2 AS DOUBLE  20 DIM VARR AS STRING |
| END | Corresponde a la terminación o cierre de las líneas de código de dicho programa.  También puede ser usada dentro de algún condicional para finalizar la ejecución de las instrucciones. |
| PRINT “…”  PRINT <listvar>  PRINT “..”; <listvar> | La instrucción PRINT Realiza la impresión de información, bien se cadenas fijas delimitadas por las comillas dobles, variables o ambas a la consola del usuario.  Ej:  100 PRINT “Variable 1: “; VAR1; “ Mas Variable 2: “; VAR2; “Igual = “; VAR1 + VAR2 |
| INPUT <var> | Permite el ingreso de valores o interacción con el usuario.  Ej:  90 INPUT VAR1 |
| IF <exprlog> THEN <STATMENT>  ENDIF | Evaluador de expresiones lógicas. En caso de ser Verdadera dicha evaluación permitirá la ejecución de la siguiente línea de código. |
| IF <exprelog> THEN <STATMENTS>  ELSE <STATEMENTS>  ENDIF | Evaluador de expresiones lógicas. En caso de ser Verdadera dicha evaluación permitirá la ejecución de las siguientes líneas de código.  En caso de no ser verdadera la evaluación de una expresión lógica marcada con la anterior palabra reservada (si), se dará la ejecución de la(s) siguiente(s) líneas de código.  Ej:  200 IF VAR1 == 100 THEN  210 PRINT “VOLVER”  220 GOTO 100  230 ELSE  240 PRINT “CONTINUAR”  250 ENDIF |
| WHILE <exprlog>  <STATMENTS>  WEND | Evaluador de expresiones lógicas, (BUCLE) con repetición de ejecución de líneas de código que se encuentren escritas dentro de su respectivo cuerpo.  Ej:  200 WHILE VAR1 <= 0 OR VAR2 <=0  210 INPUT “INGRESE VAL1”; VAR1  220  230 PRINT “INGRESE VAL2”  240 INPUT VAR2  250 WEND |
| GOTO <LINENUM> | Instrucción que ordena el salto a una línea especifica para continuar la ejecución de instrucciones correspondientes.  Ej:  100 INPUT “DIGITE SU EDAD”; VAR1  200 IF VAR1 == 100 THEN  210 PRINT “VOLVER”  220 GOTO 100  230 ELSE  240 PRINT “CONTINUAR”  250 ENDIF |
| () | Caracteres de apertura y cierre para la construcción de expresiones aritméticas |
| , | Carácter de separación de VARIABLES EN SENTENCIA DIM |
| ; | Carácter de separación de elementos en instrucciones PRINT, INPUT (STATEMENT) |
| “” | Comillas abren y cierran cadenas constantes |
| \* | Operador aritmético, multiplicación, en expresiones aritméticas |
| + | Operador aritmético, suma, en expresiones aritméticas |
| - | Operador aritmético, resta, en expresiones aritméticas  También define el valor negativo de un valor o numero en expresión o asignación. |
| / | Operador aritmético, división, en expresiones aritméticas |
| ^ | Operador de potencia, seguida de un numero al cual potenciar |
| AND | Operador Lógico, Y, en expresiones lógicas |
| OR | Operador Lógico, O, en expresiones lógicas |
| < | Operador Lógico, Menor que. |
| == | Operador Lógico de igualdad |
| > | Operador Lógico, Mayor que. |
| <= | Operador Lógico, Menor o igual |
| >= | Operador Lógico, Mayor o igual |
| . | Operador aritmético, separador de unidades de décimas, para escribir valores decimales |
| 0 – 9 | Dígitos, usados en expresiones aritméticas y/o asignaciones. |

LAS LINEAS DE CODIGO DEBEN ENUMERARSE AL PRINCIPIO DE CADA LINEA, LE PUEDEN SEGUIR UNO O VARIOS ESPACIOS EN BLANCO O TABULADORES.

LOS NOMBRES DE LAS VARIABLE DEBEN INICIAR CON UN CARÁCTER LETRA Y PUEDE SEGUIRLE CARACTERES ALFANUMERICOS. MAXIMO 10 CARACTERES (de lo contrario generará un error de sintaxis).

LAS EXPRESIONES ARITMENTICAS DEBEN SER RESUELTAS MEDIANTE PROGRAMACION, NO SE DEBE USAR ANALIZADORES DE EXPRESIONES HECHO POR TERCEROS O LIBRERIAS ESPECIALIZADAS.

Ejemplos:

1. **Algoritmo de Operaciones básicas**

110 DIM VAL1, VAL2, VAL3, RES AS DOUBLE

120 PRINT “Digite el PRIMER VALOR:”

130 INPUT VAL1

140 PRINT “Digite el SEGUNDO VALOR:”

150 PRINT “Digite el TERCER VALOR:”

160 RES = VAL1 + VAL2 /VAL3

170 PRINT “PRIMER RESULTADO :”; RES

180 RES = (VAL1+VAL2)/VAL3

190 PRINT “SEGUNDO RESULTADO :”; RES

200 RES = VAL1 / VAL2 \* VAL3

210 PRINT “TERCER RESULTADO :”; RES

220 END

1. **Algoritmo de potencia**

**110 DIM Base, Pow AS DOUBLE** ;

**120 DIM Cont; RES; //INUTIL PERRA**

**130 Base = 0**

**135 Pow = 0**

**140 Cont = 0**

**150 PRINT “Digite el valor que desea elevar :”**

**160 INPUT Base**

**170 RES = Base**

**180 PRINT “Digite el valor de la potencia :”**

190 WHILE ( Cont < Pow )

200 Cont = Cont + 1

210 RES = Base \* RES

220 WEND

230 PRINT “Resultado” , Base , “Elevado a “, Pow, “ igual “, RES ;

240 PRINT “FIN” ;

250 END