Proyecto # 1: Intérprete de Jay.

Jay consiste en un lenguaje experimental que se utiliza para explicar muchas de las características y problemas en el diseño de lenguajes de programación imperativos. Este lenguaje debería ser familiar y fácil de asimilar para todos aquellos que hayan escrito programas en Pascal, C, C++ o Java. El objetivo de esta asignación consiste en implementar un intérprete de este lenguaje que permita avaluar (de forma léxica y sintáctica) y ejecutar un código fuente escrito en este particular lenguaje.

Para cumplir con este objetivo se trata de escribir una aplicación, implementando la especificación sintáctica en el lenguaje de programación Python (versión 2.7), para obtener un analizador léxico-sintáctico que permita evaluar la validez sintáctica de los mismos.

No se permite el uso de librerías o elementos externos complementarios al lenguaje de programación. Solo se podrá utilizar las librerías nativas del lenguaje que trae la instalación estándar.

Entrada y Salida.

El texto que se quiere evaluar está grabado, en un archivo de tipo texto (FUENTE.PSE). La ejecución de cada línea de código procesada será visualizada (en consola) por medio de una tabla en donde se indiquen los valores de las variables definidas, si la misma no posee errores léxicos o de sintaxis. En caso de poseer errores léxicos o de sintaxis, el formato de salida será: "LINEA "+<número>":" seguido de la expresión "Error léxico/sintaxis" y se detendrá la ejecución del intérprete.

Entrega.

La entrega se realizará por medio de un pendrive, con las siguientes consideraciones:

- Una carpeta la cual debe indicar entre corchetes la identificación del proyecto. Ejemplo: `[LP1_Alvarez_B]`.
- Se consignará el código fuente en Python (.py), **autodocumentado**, con la identificación del estudiante (nombre, apellido y cedula). Este archivo debe contener la especificación en EBNF del objeto de estudio, como parte de la documentación del mismo.
- Un archivo **.PSE** de prueba.

FECHA DE ENTREGA	**HORA:**	**07:45 am**
	FUENTE:	**06/diciembre**

El objeto de estudio.

Sintaxis Léxica de Jay.

La siguiente gramática define formalmente los elementos que pueden aparecer en un programa de Jay.

InputElement→*WhiteSpace* | *Comment* | *Token*
WhiteSpace→space | \t | \r | \n | \f
Comment→// cualquier cadena finalizada en \r o \n
Token→*Identifier* | *Keyword* | *Literal* | *Separator* | *Operartor*
Identifier→*Letter* | *IdentifierLetter* | *IdentifierDigit*
Letter→a | b | … | z | A | B | …. | Z
Digit→0 | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9
Keyword→boolean | else | if | int | main | void | while
Literal→*Boolean Integer*
Boolean→true | false
Integer→*Digit* | *IntegerDigit*
Separator→(|) | { | } | ; | ,
Operator→= | + | - | * | / | < | <= | >= | == | != | && | || | !

Sintaxis Concreta de Jay

A continuación, se presenta las reglas sintácticas del lenguaje Jay.

Program→ void main() ' { ' *DeclarationsStatements* ' } '
Declarations→{ *Declaration* }*
Declaration→*Type Identifiers;*
Type→int | boolean
Identifiers→*Identifier*{ ,*Identifier* }*
Statements→{ *Statement* }*
Statement→*Block* | *Assignment* |*IfStatement* | *WhileStatementk*
Block→' { '*Statements* ' } '
Assignment→*Identifier*= *Expression*;
IfStatement→ if (*Expression*) *Statement* [else *Statement*]
WhileStatementk→ while (*Expression*) *Statement*
Expression→*Conjunction*{ || *Conjunction* }*
Conjunction →*Relation*{ &&*Relation*}*
Relation→*Addition*{ [< |<=|==|>=|>|!=] *Addition*}*
Addition→*Term*{[+|-] Term }*
Term→*Negation*{[*|/] *Negation*}*
Negation→[!] *Factor*
Factor→*Identifier*| *Literal*| (*Expression*)