**Un dibujo de un personaje animado

Descripción generada automáticamente con confianza baja**

**Tecnológico Nacional de México**

Instituto Tecnológico de Veracruz

EJ23 6J1A - Lenguajes y Autómatas I

Docente: Ofelia Gutiérrez Giraldi

Proyecto: Fase Sintáctica del Compilador

Equipo 2 - Integrantes:

Honorio Acosta Ruiz

Laura Espejo Alvarado

Kitzia Guadalupe Munive Cabal

H. Veracruz, Ver., 18 de mayo 2023

**Tabla de contenido**

[ELL (EasyLearningLenguage) 2](#_Toc135258651)

[Propósito 2](#_Toc135258652)

[Origen del nombre ELL: EasyLearningLenguage 2](#_Toc135258653)

[Manual de Usuario 3](#_Toc135258654)

[Conceptos Básicos 3](#_Toc135258655)

[Tipo de Datos 3](#_Toc135258656)

[Palabras Clave 4](#_Toc135258657)

[Operadores 5](#_Toc135258658)

[Sintaxis 5](#_Toc135258659)

[Variables en ELL 5](#_Toc135258660)

[Imprimir en ELL 6](#_Toc135258661)

[Sentencias decisión en ELL 7](#_Toc135258662)

[Sentencias Bucle en ELL 9](#_Toc135258663)

[Tabla de errores 11](#_Toc135258664)

[Manual del Sistema 13](#_Toc135258665)

[Requisitos del sistema 13](#_Toc135258666)

[Descripción del sistema 13](#_Toc135258667)

[Tabla de valores 13](#_Toc135258668)

[Conceptos Básicos 15](#_Toc135258669)

[Tipo de Datos 15](#_Toc135258670)

[Operadores 16](#_Toc135258671)

[Sintaxis 16](#_Toc135258672)

[Variables en ELL 16](#_Toc135258673)

[Imprimir en ELL 17](#_Toc135258674)

[Sentencias decisión en ELL 18](#_Toc135258675)

[Sentencias Bucle en ELL 20](#_Toc135258676)

[Gramáticas del Lenguaje 22](#_Toc135258677)

[Tabla de errores. 24](#_Toc135258678)

ELL (EasyLearningLenguage)

# Propósito

EasyLearningLenguage (ELL) se propone a ser un lenguaje de programación tipado y de propósito general que será diseñado para permitir el desarrollo de aplicaciones básicas.

Se centrará en la lógica de programación básica, lo que lo hará adecuado para la enseñanza de conceptos fundamentales como variables, operaciones, condicionales y ciclos. Aunque ELL no tendrá una amplia gama de características avanzadas, será diseñado para ser fácilmente escalable a medida que los usuarios adquieren más habilidades en programación.

ELL será creado con el propósito de ser un lenguaje de programación en español de nivel básico.

# Origen del nombre ELL: EasyLearningLenguage

El nombre de nuestro lenguaje es ELL, debido a que es un lenguaje de programación orientado a facilitar el aprendizaje de la lógica básica de programación para programadores novatos.



Figura 1 - Icono del lenguaje

El nombre ELL proviene de las siglas EasyLearningLanguage, lo que en español significa Lenguaje Fácil de Aprender.

Manual de Usuario

El respectivo manual busca proporcionar una guía clara y completa para nuestro lenguaje que permita a los programadores utilizar el lenguaje de manera efectiva, donde como objetivos tiene los siguientes:

* Facilitar el aprendizaje del lenguaje:

El manual proporciona una introducción clara al lenguaje de programación, explicando los conceptos básicos, sintaxis y características únicas del lenguaje.

* Ayudar a los programadores a detectar y corregir errores:

Incluye información sobre los errores que pueden ocurrir durante la programación con el lenguaje, incluyendo como corregir estos errores.

* Proporcionar información de referencia:

Es una fuente completa de información de referencia para el lenguaje, que incluye una lista detallada de las palabras clave, operadores y tipos de datos; permitiendo a los programadores buscar rápidamente información específica cuando la necesita.

Conceptos Básicos

# Tipo de Datos

El lenguaje ELL tiene los siguientes tipos de datos básicos:

* **Entero**: Tipo de dato que representa un numero entero.
* **Flotante**: Tipo de dato que representa un numero en decimal.
* **Cadena**: Tipo de dato que representa un texto.
* **Carácter**: Tipo de datos que representa un símbolo.
* **Booleano**: Tipo de dato que representa aquellos que tienen un valor de verdadero o falso.

# Palabras Clave

Las palabras clave del lenguaje ELL son:

|  |  |
| --- | --- |
| **Palaba clave** | **Descripción** |
| Inicio | Indica el arranque del programa. |
| Fin | Indica el cierre del programa. |
| Establecer | Indica la definición de una variable. |
| Interpretar | Indica la lectura de una variable mediante el tipo de dato. |
| Escribir | Indica la escritura de una variable o literal en terminal. |
| Falso | Indica un valor lógico falso. |
| Verdadero | Indica un valor lógico verdadero. |
| Para | Indica el inicio de un ciclo Para. |
| Hasta que | Indica la condición a evaluar del ciclo Para. |
| Con incremento | Indica el incremento del ciclo Para (el incremento por defecto es 1). |
| FinPara | Indica el final del ciclo Para. |
| Mientras | Indica el inicio del ciclo Mientras. |
| FinMientras | Indica el final del ciclo Mientras. |
| Repetir | Indica el inicio del ciclo Repetir. |
| Finaliza cuando | Indica la condición a evaluar del ciclo Repetir. |
| Si | Indica el inicio del condicional Si. |
| Entonces | Indica el cuerpo del condicional Si. |
| Sino | Indica el caso contrario del condicional Si. |
| FinSi | Indica el final del condicional Si. |
| Conforme | Indica el inicio del condicional Conforme. |
| Hacer | Indica el cuerpo del condicional Conforme. |
| Caso | Indica un caso del condicional Conforme. |
| En otro caso | Indica el caso por defecto del condicional Conforme. |
| FinConforme | Indica el fin del condicional Conforme. |

Tabla 1 - Palabras clave

# Operadores

Los operadores con los que cuenta el lenguaje ELL son:

|  |  |
| --- | --- |
| **Operador** | **Descripción** |
| = | Operador que indica asignación. |
| + | Operador que indica una suma. |
| - | Operador que indica una resta. |
| \* | Operador que indica una multiplicación. |
| / | Operador que indica una división. |
| % | Operador que indica al módulo. |
| && | Operador lógico AND. |
| || | Operador lógico OR. |
| ! | Operador lógico NOT. |
| == | Operador de comparación Igual que. |
| != | Operador de comparación diferente. |
| > | Operador de comparación mayor que. |
| < | Operador de comparación menor que. |
| >= | Operador de comparación mayor o igual que. |
| <= | Operador de comparación menor o igual que. |

Tabla 2 - Operadores

Sintaxis

# Variables en ELL

Las variables son un espacio de memoria para almacenar un dato. Para definir una variable seguimos esta estructura:

Inicio

Establecer [Tipo\_De\_Dato][Nombre\_Variable];

Fin

Figura 2 - Estructura de cómo crear variables.

Algunos ejemplos de variables serían:

Figura 3 - Ejemplo de creación de variables.

Inicio

Establecer Entero edad; # Esta en una variable entera

Establecer Flotante estatura; # Esta en una variable flotante

Establecer Cadena nombre; # Esta es una variable cadena

Establecer Carácter sexo; # Esta es una variable tipo carácter

Establecer Booleano estaVivo; # Esta es una variable tipo booleana

Fin

# Imprimir en ELL

Imprimir se refiere a mostrar información en la salida de texto del programa; es una forma fácil de visualizar resultados, mensajes o datos. Para imprimir definimos la siguiente estructura:

Inicio

Escribir “Cadena\_de\_Texto”;

Escribir [variable];

Fin

Figura 4 - Estructura de Escribir

Solo se puede hacer una acción a la vez, es decir, si vas a imprimir una cadena solamente será la cadena, pero no puedes imprimir una cadena y una variable juntas.

# Sentencias decisión en ELL

Las sentencias de decisión son sentencias que nos permiten tomar una decisión para poder ejecutar un bloque de sentencias u otro. Las sentencias de decisión que contiene el lenguaje son:

**Si – Entonces – Sino**

La estructura de la sentencia Si-Entonces-Sino es:

Inicio

Si [condición] Entonces

#Bloque de sentencias

Sino

#Bloque de sentencias

FinSi

Fin

Figura 5 - Estructura de Si-Entonces-Sino

La parte del Sino no tiene por qué existir. En este caso tendríamos una sentencia Si-Entonces.

Inicio

Si [condición] Entonces

#Bloque de sentencias

FinSi

Fin

Figura 6 - Sentencia Si - Entonces

La sentencia Si-Entonces-Sino pueden estar anidadas y así nos encontraríamos con una sentencia Si-Entonces-SinoSi, la cual tendría la siguiente estructura:

Inicio

Si [condición] Entonces

#Bloque de sentencias

Sino Si [condición] Entonces

#Bloque de sentencias

Sino Si [condición] Entonces

#Bloque de sentencias

Sino

#Bloque de sentencias

FinSi

Fin

Figura 7-Sentencia Si-Entonces-SinoSi

**Conforme**

Para los casos en los que se tienen muchas ramas o caminos de ejecución en una sentencia Si tenemos la sentencia Conforme. La sentencia Conforme evalúa una expresión y ejecutara el bloque de sentencias que coincida con el valor de la expresión.

El valor de la expresión puede ser numérico o al igual se pueden utilizar expresiones cuya evaluación sean cadenas.

La estructura de la sentencia Conforme es:

# Sentencias Bucle en ELL

Inicio

Conforme [condición] Hacer

Caso [valor1]:

#Bloque de sentencias

Caso [valor2]:

#Bloque de sentencias

En Otro Caso:

#Bloque de sentencias

FinConforme

Fin

Figura 8 - Estructura de sentencia Conforme

Las sentencias de bucle nos van a permitir ejecutar un bloque de sentencias tantas veces como queramos, o tantas veces como se cumpla una condición. Las sentencias de bucle que contiene el lenguaje son:

**Para**

La estructura del bucle Para es:

Inicio

Para [sentencia\_inicio] Hasta que [condición] Con incremento

[valor\_entero\_ó\_decimal]

#Bloque de sentencias

FinPara

Fin

Figura 9-Estructura Para

Las funcionalidades en las que podemos utilizar la sentencia Para puede ser como un contador.

**Mientras**

La estructura repetitiva Mientras realiza una primera evaluación antes de ejecutar el bloque. Si la expresión es verdadera pasa a ejecutar de forma repetitiva el bloque de sentencias.

La estructura de la sentencia Mientras es la siguiente:

Inicio

Mientras [condición] Hacer

#Bloque de sentencias

FinMientras

Fin

Figura 10-Estructura de sentencia Mientras

**Repetir**

La estructura de la sentencia Repetir es la siguiente:

Figura 11-Estructura de sentencia Repetir

Inicio

Repetir

#Bloque de sentencias

Finalizar cuando [condición];

Fin

Tabla de errores

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **Número de error** | **Tipo** | **Ubicación** | **Token** | **Descripción** | **Solución.** |
| 1 | Error léxico | Línea 3, columna 6 | , | Símbolo ‘,’ desconocido | Remover símbolo ‘,’ |
| 2 | Error sintáctico | Línea 3, Columna 24 | ; | Se encontró símbolo “;” | Se esperaba uno de los siguientes:   * <CADENA\_TEXTO> * <CARÁCTER\_TEXTO> * <NUMERO\_ENTERO> * <NUMERO\_DECIMAL> * “Falso” * “Verdadero” * “!” * “(“ * <VARIABLE> |
| 3 | Error sintáctico | Línea 13, columna 32 | Mientras | Se encontró el símbolo “Mientras” | Se esperaba uno de los siguientes:   * “;” * “+” * “-” * “\*” * “/” * “%” * “| |” * “==” * “!=” * “>” * “<” * “>=” * “<=” |
| 4 | Error sintáctico | Línea 19, columna 9 | <VARIABLE> | Se encontró el símbolo <VARIABLE> | Se esperaba uno de los siguientes:   * <CADENA\_TEXTO> * <CARÁCTER\_TEXTO> * <NUMERO\_ENTERO> * <NUMERO\_DECIMAL> * “Falso” * “Verdadero” |
| 5 | Error sintáctico | Línea 1, columna 1 | Fin | Se encontró el símbolo “Fin” | Se esperaba uno de los siguientes:   * <COMENTARIO> * “Interpretar” * “Escribir” * "Establecer" * "Para" * "Mientras" * "Repetir" * "Si" * "Conforme" * <VARIABLE> |

Manual del Sistema

El respectivo manual busca proporcionar una guía clara y completa para nuestro lenguaje que permita a los programadores utilizar el lenguaje de manera efectiva y dar información clara del lenguaje, donde como objetivos tiene los siguientes:

* Describir el sistema: El manual de sistema tiene como objetivo principal proporcionar una descripción completa y detallada del sistema.
* Orientar en la solución de problemas: El manual debe proporcionar información sobre la solución de problemas comunes que los usuarios pueden enfrentar al utilizar el sistema.
* Servir como referencia: El manual de sistema debe ser una referencia útil para los usuarios, brindando información detallada sobre todas las funciones, configuraciones y aspectos técnicos del sistema.

# Requisitos del sistema

* Tener Java instalado.
* Tener JavaCC instalado.

# Descripción del sistema

EC (EasyCompiler) es un compilador de código abierto que admite el lenguaje de programación ELL (EasyLearningLenguage). El compilador se ejecuta en cualquier plataforma que tenga instalada Java.

# Tabla de valores

|  |  |  |
| --- | --- | --- |
| **Valor** | **Tipo** | **Expresión regular** |
| Inicio | PR\_Arranque\_Programa |  |
| Fin | PR\_Cierre\_Programa |  |
| # | Simbolo\_Comentario |  |
| edad | Variable |  |
| Establecer | PR\_Definicion\_Variable |  |
| Entero | PR\_Tipo\_De\_Dato |  |
| Flotante | PR\_Tipo\_De\_Dato |  |
| Cadena | PR\_Tipo\_De\_Dato |  |
| Carácter | PR\_Tipo\_De\_Dato |  |
| Booleano | PR\_Tipo\_De\_Dato |  |
| ; | Delimitador |  |
| Interpretar | PR\_Lectura |  |
| Escribir | PR\_Escritura |  |
| 123 | Numero\_Entero |  |
| 123.123 | Numero\_Flotante |  |
| "Cadena" | Cadena\_Texto |  |
| 'C' | Caracter\_Texto |  |
| = | Operador\_Asignacion |  |
| + | Operador\_Suma |  |
| - | Operador\_Resta |  |
| \* | Operador\_Mult |  |
| / | Operador\_Div |  |
| % | Operador\_Mod |  |
| Falso | PR\_Booleano\_Falso |  |
| Verdadero | PR\_Booleano\_Verdadero |  |
| && | Operador\_Logico\_AND |  |
| || | Operador\_Logico\_OR |  |
| ! | Operador\_Logico\_NOT |  |
| == | Operador\_igualQue |  |
| != | Operador\_diferente |  |
| > | Operador\_mayorQue |  |
| < | Operador\_menorQue |  |
| >= | Operador\_mayorIgualQue |  |
| <= | Operador\_menorIgualQue |  |
| Para | PR\_Ciclo\_Para |  |
| Hasta que | PR\_Condicion\_Ciclo\_Para |  |
| Con incremento | PR\_Incremento\_Ciclo\_Para |  |
| FinPara | PR\_Fin\_Ciclo\_Para |  |
| Mientras | PR\_Ciclo\_Mientras |  |
| FinMientras | PR\_Fin\_Ciclo\_Mientras |  |
| Repetir | PR\_Ciclo\_Repetir |  |
| Finaliza cuando | PR\_Condicion\_Ciclo\_Repetir |  |
| Si | PR\_Condicional\_Si |  |
| Entonces | PR\_Entonces |  |
| Sino | PR\_Conficional\_Sino |  |
| FinSi | PR\_Fin\_Condicional\_Si |  |
| Conforme | PR\_Condicional\_Conforme |  |
| Hacer | PR\_Hacer |  |
| Caso | PR\_Conforme\_Caso |  |
| : | Operador\_Dos\_Puntos |  |
| En otro caso | PR\_Conforme\_Caso\_Predeterminado |  |
| FinConforme | PR\_Fin\_Condicional\_Conforme |  |
| ( | Parentesis\_Abierto |  |
| ) | Parentesis\_Cerrado |  |

Tabla 3 – Tabla de valores

Conceptos Básicos

# Tipo de Datos

El lenguaje ELL tiene los siguientes tipos de datos básicos:

* **Entero**: Tipo de dato que representa un numero entero.
* **Flotante**: Tipo de dato que representa un numero en decimal.
* **Cadena**: Tipo de dato que representa un texto.
* **Carácter**: Tipo de datos que representa un símbolo.
* **Booleano**: Tipo de dato que representa aquellos que tienen un valor de verdadero o falso.

# Operadores

Los operadores con los que cuenta el lenguaje ELL son:

|  |  |
| --- | --- |
| **Operador** | **Descripción** |
| = | Operador que indica asignación. |
| + | Operador que indica una suma. |
| - | Operador que indica una resta. |
| \* | Operador que indica una multiplicación. |
| / | Operador que indica una división. |
| % | Operador que indica al módulo. |
| && | Operador lógico AND. |
| || | Operador lógico OR. |
| ! | Operador lógico NOT. |
| == | Operador de comparación Igual que. |
| != | Operador de comparación diferente. |
| > | Operador de comparación mayor que. |
| < | Operador de comparación menor que. |
| >= | Operador de comparación mayor o igual que. |
| <= | Operador de comparación menor o igual que. |

Tabla 2 - Operadores

Sintaxis

# Variables en ELL

Las variables son un espacio de memoria para almacenar un dato. Para definir una variable seguimos esta estructura:

Inicio

Establecer [Tipo\_De\_Dato][Nombre\_Variable];

Fin

Figura 2 - Estructura de cómo crear variables.

Algunos ejemplos de variables serían:

Figura 3 - Ejemplo de creación de variables.

Inicio

Establecer Entero edad; # Esta en una variable entera

Establecer Flotante estatura; # Esta en una variable flotante

Establecer Cadena nombre; # Esta es una variable cadena

Establecer Carácter sexo; # Esta es una variable tipo carácter

Establecer Booleano estaVivo; # Esta es una variable tipo booleana

Fin

# Imprimir en ELL

Imprimir se refiere a mostrar información en la salida de texto del programa; es una forma fácil de visualizar resultados, mensajes o datos. Para imprimir definimos la siguiente estructura:

Inicio

Escribir “Cadena\_de\_Texto”;

Escribir [variable];

Fin

Figura 4 - Estructura de Escribir

Solo se puede hacer una acción a la vez, es decir, si vas a imprimir una cadena solamente será la cadena, pero no puedes imprimir una cadena y una variable juntas.

# Sentencias decisión en ELL

Las sentencias de decisión son sentencias que nos permiten tomar una decisión para poder ejecutar un bloque de sentencias u otro. Las sentencias de decisión que contiene el lenguaje son:

**Si – Entonces – Sino**

La estructura de la sentencia Si-Entonces-Sino es:

Inicio

Si [condición] Entonces

#Bloque de sentencias

Sino

#Bloque de sentencias

FinSi

Fin

Figura 5 - Estructura de Si-Entonces-Sino

La parte del Sino no tiene por qué existir. En este caso tendríamos una sentencia Si-Entonces.

Inicio

Si [condición] Entonces

#Bloque de sentencias

FinSi

Fin

Figura 6 - Sentencia Si - Entonces

La sentencia Si-Entonces-Sino pueden estar anidadas y así nos encontraríamos con una sentencia Si-Entonces-SinoSi, la cual tendría la siguiente estructura:

Inicio

Si [condición] Entonces

#Bloque de sentencias

Sino Si [condición] Entonces

#Bloque de sentencias

Sino Si [condición] Entonces

#Bloque de sentencias

Sino

#Bloque de sentencias

FinSi

Fin

Figura 7-Sentencia Si-Entonces-SinoSi

**Conforme**

Para los casos en los que se tienen muchas ramas o caminos de ejecución en una sentencia Si tenemos la sentencia Conforme. La sentencia Conforme evalúa una expresión y ejecutara el bloque de sentencias que coincida con el valor de la expresión.

El valor de la expresión puede ser numérico o al igual se pueden utilizar expresiones cuya evaluación sean cadenas.

La estructura de la sentencia Conforme es:

# Sentencias Bucle en ELL

Inicio

Conforme [condición] Hacer

Caso [valor1]:

#Bloque de sentencias

Caso [valor2]:

#Bloque de sentencias

En Otro Caso:

#Bloque de sentencias

FinConforme

Fin

Figura 8 - Estructura de sentencia Conforme

Las sentencias de bucle nos van a permitir ejecutar un bloque de sentencias tantas veces como queramos, o tantas veces como se cumpla una condición. Las sentencias de bucle que contiene el lenguaje son:

**Para**

La estructura del bucle Para es:

Inicio

Para [sentencia\_inicio] Hasta que [condición] Con incremento

[valor\_entero\_ó\_decimal]

#Bloque de sentencias

FinPara

Fin

Figura 9-Estructura Para

Las funcionalidades en las que podemos utilizar la sentencia Para puede ser como un contador.

**Mientras**

La estructura repetitiva Mientras realiza una primera evaluación antes de ejecutar el bloque. Si la expresión es verdadera pasa a ejecutar de forma repetitiva el bloque de sentencias.

La estructura de la sentencia Mientras es la siguiente:

Inicio

Mientras [condición] Hacer

#Bloque de sentencias

FinMientras

Fin

Figura 10-Estructura de sentencia Mientras

**Repetir**

La estructura de la sentencia Repetir es la siguiente:

Figura 11-Estructura de sentencia Repetir

Inicio

Repetir

#Bloque de sentencias

Finalizar cuando [condición];

Fin

Gramáticas del Lenguaje

Tabla de errores.

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **Número de error** | **Tipo** | **Ubicación** | **Token** | **Descripción** | **Solución.** |
| 1 | Error léxico | Línea 3, columna 6 | , | Símbolo ‘,’ desconocido | Remover símbolo ‘,’ |
| 2 | Error sintáctico | Línea 3, Columna 24 | ; | Se encontró símbolo “;” | Se esperaba uno de los siguientes:   * <CADENA\_TEXTO> * <CARÁCTER\_TEXTO> * <NUMERO\_ENTERO> * <NUMERO\_DECIMAL> * “Falso” * “Verdadero” * “!” * “(“ * <VARIABLE> |
| 3 | Error sintáctico | Línea 13, columna 32 | Mientras | Se encontró el símbolo “Mientras” | Se esperaba uno de los siguientes:   * “;” * “+” * “-” * “\*” * “/” * “%” * “| |” * “==” * “!=” * “>” * “<” * “>=” * “<=” |
| 4 | Error sintáctico | Línea 19, columna 9 | <VARIABLE> | Se encontró el símbolo <VARIABLE> | Se esperaba uno de los siguientes:   * <CADENA\_TEXTO> * <CARÁCTER\_TEXTO> * <NUMERO\_ENTERO> * <NUMERO\_DECIMAL> * “Falso” * “Verdadero” |
| 5 | Error sintáctico | Línea 1, columna 1 | Fin | Se encontró el símbolo “Fin” | Se esperaba uno de los siguientes:   * <COMENTARIO> * “Interpretar” * “Escribir” * "Establecer" * "Para" * "Mientras" * "Repetir" * "Si" * "Conforme" * <VARIABLE> |