

Proyecto

Instituto Tecnológico de Costa Rica
Área de Ingeniería en Computadores
CE3104 – Lenguaje, Compiladores e Intérpretes
I Semestre 2022 Grupo 2



TEC | Tecnológico
de Costa Rica

Objetivo general

- Implementar un compilador por medio de la técnica y el arte.

Objetivos específicos

- Implementación de un compilador para un lenguaje de programación usando las herramientas de validación necesarias.
- Utilización de una herramienta para construir reconocedores (parsers), intérpretes, compiladores y traductores de lenguajes a partir de las descripciones gramaticales de los mismos (conteniendo acciones semánticas a realizarse en varios lenguajes de programación).

Datos Generales

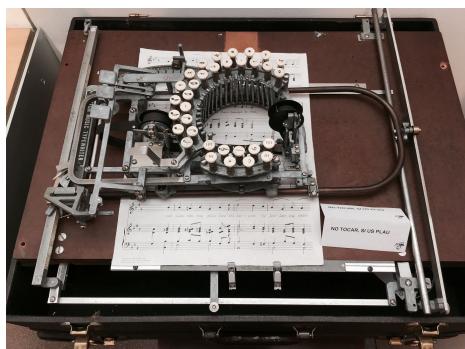
- El valor del Proyecto: 25%
- Nombre código: **Writing Machine**
- El proyecto debe ser implementado por grupos de máximo de 4 personas.
- La fecha de entrega del proyecto es de **3/Junio/2022**
- Cualquier indicio de copia de proyectos será calificado con una nota de 0 y será procesado de acuerdo con el reglamento.

Marco Conceptual

De acuerdo con Wikipedia, las máquinas de escribir fueron herramientas indispensables en las oficinas de todo el mundo, así como para la literatura, el cine, el periodismo, el teatro y cualquier actividad que requiriera escribir desde finales del siglo XIX y casi todo el siglo XX. En la década de 1980 los procesadores de texto para computadoras personales reemplazaron casi totalmente a las máquinas de escribir en los países desarrollados, aunque en otras regiones su uso no se vio afectado por el poco avance de las nuevas tecnologías hasta entrado el siglo XXI.



La invención de la máquina de escribir permitió suplantar a los lentos copistas y le dio un carácter más oficial e impersonal a los escritos comerciales y políticos. El procedimiento mecánico de escritura aceleró el ritmo de las comunicaciones, marcó un punto importante en el desarrollo de las relaciones sociales y le permitió a la mujer ingresar masivamente al mundo laboral como dactilógrafa, entre los siglos XIX y XX. Podríamos decir de cierta manera que la máquina de escribir le abrió el camino a las máquinas eléctricas y por qué no a las computadoras.



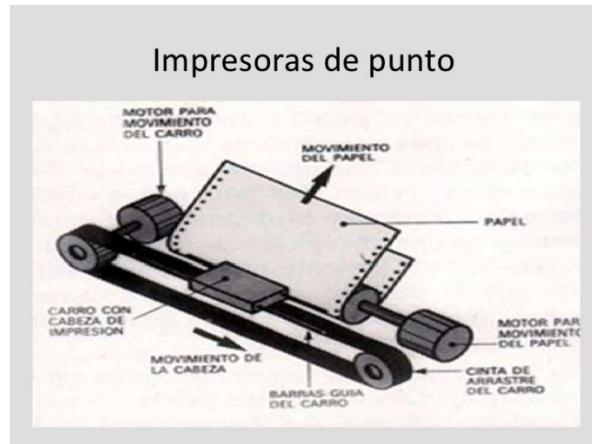
Máquina de escribir las partituras musicales

Debido a que las computadoras generaron muchas facilidades a las personas al momento de escribir, su popularidad comenzó a crecer debido a que eran mucho más útiles que las máquinas de escribir y las fueron desplazando.

Posteriormente llegaron las impresoras, donde las impresoras de impacto trabajan con un cabezal en el que hay agujas, estas agujas golpean una cinta, similar al de una máquina de escribir, que genera la impresión de la letra.

Una impresora es un dispositivo periférico de salida, del ordenador que permite producir una gama permanente de textos o gráficos de documentos almacenados en un formato electrónico, imprimiéndolos en medios físicos, normalmente en papel, utilizando cartuchos de tinta o tecnología láser

La impresora matricial o impresora de matriz de puntos es un tipo de impresora con la cabeza de impresión que se desplaza de izquierda a derecha imprimiendo sobre la página por impacto, oprimiendo una cinta de tinta contra el papel, de forma similar al funcionamiento de una máquina de escribir. Al contrario que las máquinas de escribir o impresoras de margarita, las letras son obtenidas por selección de puntos de una matriz, y por tanto es posible producir distintos tipos de letra, y gráficos en general.

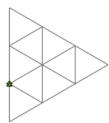


Descripción del problema

Para efectos del proyecto de Compiladores e Intérpretes se desea la construcción de los siguientes elementos:

- Implementación de un Lenguaje de programación propio de acuerdo con las reglas indicadas en el presente enunciado enriquecido con elementos acordados por cada grupo de proyecto.
- Implementación de un Intérprete que reúna las características mostradas en el curso y con la ayuda de software especializado para tal efectos.
- Dispositivo de hardware que permita la impresión física de la lógica de programación colocada en un programa fuente.

Los tres elementos anteriormente mencionados deben estar totalmente relacionados. El dispositivo de hardware debe ser capaz de “dibujar” desde simples símbolos hasta imágenes más complejas, así como imprimir mensajes de texto sin restricciones de formato pues el mensaje será confeccionado como imagen.



Para realizar el elemento de hardware deberá construir un dispositivo que pueda ser capaz de realizar las siguientes funciones:

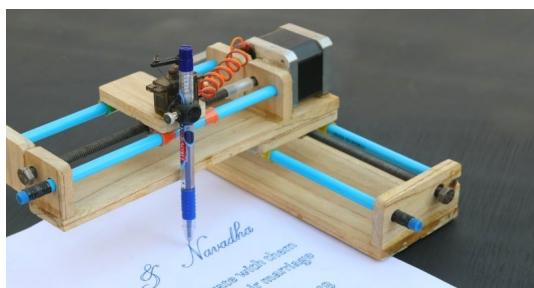
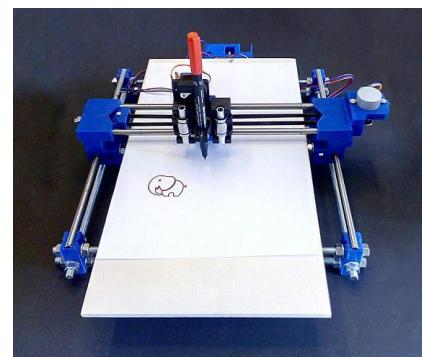
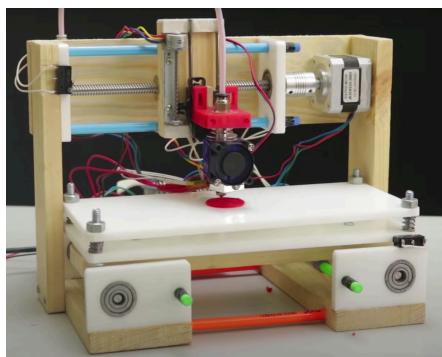
- Debe ser capaz de tener dos compartimentos en el cual podrá ubicar un lapicero (o lápiz de color o similar) en cada uno de ellos.
- Puede subir o bajar cada uno de los compartimentos de tal manera que pueda ser capaz de usarlo sobre algún tipo de superficie.
- El dispositivo debe tener la capacidad de movimientos entre fila (movimiento derecha a izquierda y viceversa) y movimientos entre columnas (movimiento de arriba hacia abajo y viceversa).

- El compartimiento activo puede ser ubicado sobre la superficie en una posición en particular usando una ubicación de coordenadas.

El dispositivo debe ser capaz de realizar impresión de forma continua o de tipo punto, y alternar colores de acuerdo con los compartimientos seleccionados.

Deben utilizar algún tipo de microcontrolador y la circuitería necesaria para llevar a cabo el proyecto, y toda la arquitectura debe ser diseñada, ensamblada y puesta en práctica por el equipo de trabajo, no se debe utilizar soluciones completas no creadas por ustedes mismos. El dispositivo debe evitar a lo máximo la manipulación humana, para su ejecución.

Se deja a creatividad de los integrantes de cada grupo, la búsqueda de la solución que consideren más adecuada y justificar la misma en la documentación del proyecto. A continuación, se presentan algunas imágenes de posibles soluciones, con el único fin de orientar a un entendimiento más ajustado a los requerimientos antes expuestos.



Compilador de Writing Machine

Para efectos del proyecto de “Compiladores e Intérpretes” del ITCR, se desea desarrollen desde 0 (*desde el inicio*) el lenguaje “Writing Machine”, el cual tendrá las siguientes características:

- Para efectos de la validación léxica y sintáctica, pueden utilizar **Lex y Yacc**. Esto quiere decir que el intérprete debe “reaccionar” a posibles errores y evitar su ejecución.
- El lenguaje de programación usado para programar se deja a criterio de su propio equipo de trabajo.
- Deben hacer la “impresión” de los programa según el requerimientos de la interfaz que más adelante se detalla.
- Se espera que el proyecto sea **completamente funcional**, y que permita la impresión de todo y cualquier imagen o elemento escrito en el código del lenguaje de programación creado.
- **Se debe presentar por parte del grupo, un programa con una imagen compleja (desde el punto de vista de arte), la cual será evaluada en la revisión.**
- Todo el proyecto debe ser implementado por el grupo de estudiantes, en caso de querer la utilización de bibliotecas, módulos u otras facilidades (no mencionadas anteriormente) y que disminuiría la carga de trabajo, se debe consultar previamente con el profesor y obtener su aprobación en dicho uso, en caso contrario se tomará como trabajo inconcluso.
- Se debe generar una interfaz en la computadora **agradable al usuario tipo IDE**, donde se pueda escribir y editar de manera amigable los programas de acuerdo con el lenguaje de programación creado.

Se debe crear un intérprete que permita la ejecución de las instrucciones necesarias para llevar a cabo cada una de las impresiones requeridas de acuerdo con el código fuente generado usando las reglas del lenguaje de programación.

El intérprete por implementar debe llevar a cabo todas las etapas del front-end de un compilador, y mostrar los correspondientes errores léxicos, sintácticos y semánticos.

Para hacer lo anterior, deberán construir una gramática que soporte la funcionalidad que más adelante se indique. El lenguaje construido debe ser flexible desde el punto de vista, que pueda permitir crear distintos programas que parametrizará la impresión.

En lo que respecta a la implementación del Interpretante:

- Se solicitará en el momento de la revisión, una explicación detallada de la Gramática generada. Se debe demostrar un dominio suficiente sobre la “lógica” colocada en la gramática.

- Se solicitará una explicación completa de lo implementado en **Lex** y **Yacc** específicamente en lo realizado sobre el análisis léxico y el análisis sintáctico.

Entorno de Programación (IDE)

Se debe crear una interfaz gráfica la cual será el **único medio de revisión del proyecto**. En esta interfaz se deben ver claramente los errores y warnings del compilador. Esta interfaz debe contener los siguientes elementos:

- Poder ser capaces de “cargar” programas predefinidos para modificarlos, compilarlos y ejecutarlos.
- Debe existir un botón de “compilación” el cual no ejecute el código sino simplemente realiza “pasadas” de validación según cada etapa de los compiladores.
- Debe existir un botón de “ejecución” el cual ejecutará primero la “compilación” y posteriormente de no existir ningún error, ejecutará el código.
- Debe haber una ventana en donde se pueda cargar o editar los programas y sobre los cuales se ejecutará la compilación y ejecución del código.
- Debe existir una ventana en donde se retornen los valores del comando “PrintLine”.
- Debe haber una ventana en donde aparecerán de forma “decente” los errores que la “compilación” generará. No se permitirá el uso del IDE del lenguaje de programación usado para construir el Compilador ni ningún otro medio. Solamente se hará uso de la interfaz solicitada.
- Se debe tener control sobre el número de línea del código fuente, de tal manera que en caso de presentarse un error a la hora de la compilación, se muestre exactamente la línea sobre la cual se presenta el error, además el error presentado debe ser **amigable**.

Sintaxis

- Se debe respetar la sintaxis que más adelante se indica, pero el grupo puede “enriquecer” dicha gramática añadiendo las sentencias o comportamiento que consideren necesario, pero siempre deben respetar lo colocado en el presente documento. Además se deben tomar en cuenta las siguientes instrucciones:
 - Las variables usadas en el programa fuente deben tener un máximo de 10 posiciones y como mínimo 3 posiciones. Debe iniciar siempre con una letra minúscula, las demás letras pueden ser minúsculas o mayúsculas, y solamente deberá permitir letras, números, y los símbolos especiales “_”, y “@”. Si no se cumple con lo anterior, se debe generar un error.
 - Los comentarios en el código fuente es por línea, y siempre debe iniciar con // y se comenta toda la línea, ejemplo:
 // Esto es un comentario
Todo código fuente debe tener al menos un comentario indicando el nombre y la funcionalidad del código, y esta debe encontrarse en la primer línea de todo

programa. De lo contrario **debe generar un error**. Los demás comentarios pueden estar presentes en cualquier lugar dentro del código del programa, pero siempre debe existir al menos uno en la primera línea de este.

- Deben realizar todas las validaciones pertinentes a nivel léxico, sintáctico, semántico, etc.
- El cuerpo básico de un procedimiento debe cumplir con el siguiente formato:

Para nombre_del_procedimiento [lista de parámetros]

... // Instrucciones.

fin

- Un procedimiento puede llevar parámetros de entrada.
- Un procedimiento difiere de otro procedimiento por el nombre del mismo y los parámetros que contenga. Por tanto, cabe mencionar, que la firma de un procedimiento es el nombre y sus parámetros, tomando en cuenta cantidad de parámetros. No puede existir más de un procedimiento con la misma firma, pues generaría un error.
- Adjunto una serie de sentencias mínimas que el Lenguaje debe contener, deben considerar las “expresiones” pues podrían ser colocadas en muchos lugares. Pueden mejorar sustancialmente la sintaxis recomendada, pero se debe respetar lo indicado a continuación.
- También como en otros entornos de programación un procedimiento puede llamar a otro procedimiento, lo que hará esto es realizar las instrucciones que tiene definidas el procedimiento al que se llama.
- Debe existir un procedimiento llamado MAIN que es el principal y el punto de entrada al programa. Este procedimiento no recibe parámetros. Se debe generar un error si este procedimiento no se encuentra, o bien, si hay más de uno en un programa en particular. Además, se debe generar un error si se le coloca parámetros.
- El programa MAIN puede estar colocado en cualquier parte del código.
- Las variables definidas dentro de MAIN se consideran variables globales y las variables definidas en cualquier procedimiento se consideran variables locales.
- Un procedimiento se “llama” simplemente colocando el nombre del procedimiento con sus respectivos parámetros si los tuviera.
- Ejemplo de procedimientos:

**PARA *nombre_del_procedimiento1* [*lista de parámetros*]
 *Lista de instrucciones***

.

.

FIN

**PARA *nombre_del_procedimiento2* [*lista de parámetros*]
 *Lista de instrucciones***

.

.

nombre_del_procedimiento1

FIN

- **Sintaxis mínima solicitada:**

Sentencia en Lenguaje	Acción de la sentencia
<pre>Def(nombre_variable, valor);</pre>	<p>A una variable se le puede asignar un valor, este valor puede ir cambiando a lo largo de la ejecución de un programa. El valor depende del tipo de dato que se quiera guardar en la variable. Ésta puede guardar los siguientes tipos de datos:</p> <ul style="list-style-type: none">- Número- Lógico <p>Ej.</p> <pre>Def (variable1, 5); Def (variable3,TRUE);</pre> <p>El tipo de datos de la variable será asignado con el valor inicial usado en la creación de la variable.</p> <p>Si posteriormente se le asigna otro valor de distinto tipo de datos, se deberá generar un error “semántico”.</p> <p>Si no se le asigna un valor a una variable en su definición, se debe mostrar un error.</p>

<pre>Put(identifica, n);</pre>	<p>Se altera el valor actual de la variable por el dato colocado en la instrucción. n es el valor para usar.</p> <p>Ej.</p> <pre>PUT(mivar, Substr(100, 45));</pre> <p>//mivar quedará con el valor de 55.</p> <p>“n” puede ser:</p> <ul style="list-style-type: none"> • Una constante numérica • Una operación matemática • Otra variable previamente inicializada
<pre>Add(N1);</pre> <pre>Add(N1, N2);</pre>	<p>Incrementa el valor de la variable.</p> <p>Se tiene dos sintaxis:</p> <ol style="list-style-type: none"> 1. Si solo viene un identificador, el incremento será valor identificador + 1 2. En caso de que se tengan una variable y un valor. Se incrementa N1 en N2 veces. 3. N2 puede ser una operación matemática. <p>Importante, N1 siempre debe ser una variable pues es la que sufre el incremento. En caso de no ser una variable, debe generar un error.</p> <p>Ej.</p> <pre>ADD(var);</pre> <pre>ADD(var, 5);</pre> <pre>ADD(var, var3);</pre>
<pre>ContinueUp n;</pre>	<p>Moverse n cantidad de unidades hacia arriba (eje y).</p> <p>Ej.</p> <pre>ContinueUp 10;</pre> <pre>ContinueUp 5*3;</pre> <pre>ContinueUp var1;</pre> <p>“n” puede ser:</p> <ul style="list-style-type: none"> • Una constante numérica • Una operación matemática • Otra variable previamente inicializada

	<p>Dependiendo si el lápiz esta abajo o arriba (toca o no toca el relieve) se imprimirá sobre la superficie.</p>
ContinueDown n;	<p>Moverse n cantidad de unidades hacia abajo (eje y).</p> <p>Ej.</p> <pre>ContinueDown 10; ContinueDown 5*3; ContinueDown var1;;</pre> <p>“n” puede ser:</p> <ul style="list-style-type: none"> • Una constante numérica • Una operación matemática • Otra variable previamente inicializada <p>Dependiendo si el lápiz esta abajo o arriba (toca o no toca el relieve) se imprimirá sobre la superficie.</p>
ContinueRight n;	<p>Moverse n cantidad de unidades hacia la derecha (eje x).</p> <p>Ej.</p> <pre>ContinueRight 9; ContinueRight 5;</pre> <p>“n” puede ser:</p> <ul style="list-style-type: none"> • Una constante numérica • Una operación matemática • Otra variable previamente inicializada <p>Dependiendo si el lápiz esta abajo o arriba (toca o no toca el relieve) se imprimirá sobre la superficie.</p>
ContinueLeft n;	<p>Moverse n cantidad de unidades hacia la izquierda (eje x).</p> <p>Ej.</p> <pre>ContinueLeft 9; ContinueLeft 5;</pre> <p>“n” puede ser:</p>

	<ul style="list-style-type: none"> • Una constante numérica • Una operación matemática • Otra variable previamente inicializada <p>Dependiendo si el lápiz esta abajo o arriba (toca o no toca el relieve) se imprimirá sobre la superficie.</p>
Pos (X, Y) ;	<p>Coloca el lapicero en la posición de las coordenadas X, Y.</p> <p>Ej. <code>Pos(100, 0);</code></p> <p>X y Y puede ser:</p> <ul style="list-style-type: none"> • Una constante numérica • Una operación matemática • Otra variable previamente inicializada <p>Dependiendo si el lápiz esta abajo o arriba se imprimirá sobre la superficie un punto.</p>
PosX n;	<p>Coloca el lapicero en la coordenada X especificada.</p> <p>Ej. <code>PosX 100;</code></p> <p>Se debe recordar y respetar la coordenada Y en la cual se encuentra actualmente el lapicero.</p> <p>“n” puede ser:</p> <ul style="list-style-type: none"> • Una constante numérica • Una operación matemática • Otra variable previamente inicializada <p>Dependiendo si el lápiz esta abajo o arriba se imprimirá sobre la superficie un punto.</p>

<pre>PosY n;</pre>	<p>Coloca el lapicero en la coordenada Y especificada.</p> <p>Ej. <code>PosY 100;</code></p> <p>“n” puede ser:</p> <ul style="list-style-type: none"> • Una constante numérica • Una operación matemática • Otra variable previamente inicializada <p>Se debe recordar y respetar la coordenada X en la cual se encuentra actualmente el lapicero.</p> <p>Dependiendo si el lápiz esta abajo o arriba se imprimirá sobre la superficie un punto.</p>
<pre>UseColor valor;</pre>	<p>El hardware debe disponer de dos posibles compartimientos de color definidos como 1 y 2.</p> <p>La idea es que en cada compartimiento se coloque un lapicero con distinto color, por ejemplo 1:Negro, 2:Rojo.</p> <p>Cuando el dispositivo escriba usará el color asignado en <code>UseColor</code>. Por default se deberá usar el compartimiento 1. Esto quiere decir que si no se asigna ningún valor a <code>UseColor</code>, este tomará por defecto el 1.</p> <p>El compartimiento de color se puede cambiar en cualquier momento, de la siguiente manera:</p> <p>Ej. <code>UseColor 2;</code></p> <p>Se debe validar que no se use ningún otro valor distinto a los indicados anteriormente, y por ende generar el error correspondiente.</p>
<pre>Down;</pre>	<p>El lapicero es colocado “pegado” a la superficie de tal manera que se pueda imprimir sobre dicha superficie a partir de ese momento.</p>

	<p>Ej. Down;</p> <p>El evento se hace sobre el compartimiento asignado en “UseColor” por tanto no se debe hacer ninguna otra especificación del compartimiento a usar.</p>
Up;	<p>El lapicero se “despega” de la superficie de impresión de tal manera que no es posible escribir sobre dicha superficie.</p> <p>Ej. Up;</p> <p>El evento se hace sobre el compartimiento asignado en “UseColor” por tanto no se debe hacer ninguna otra especificación del compartimiento a usar.</p>
Beginning;	Coloca el lápiz al inicio, esto quiere decir en la posición [1,1]
Speed n	<p>Determina la velocidad en milisegundo en la impresión .</p> <p>Ej. Speed 100;</p>
Run [órdenes]	<p>Permite ejecutar las “Órdenes”. Corresponde al cuerpo de instrucciones.</p> <p>Ej. Run [PosY 10; Down;];</p> <p>Las sentencias dentro del RUN deben estar separadas por punto y coma, sino se retorna un error.</p>
Repeat n [órdenes]	<p>Ejecuta todas las instrucciones que se encuentran en el cuerpo (Órdenes) por n cantidad de veces.</p> <p>Ej. Repeat 4 [PosY 10; ContinueRight 9;]</p>

	<p>“n” puede ser:</p> <ul style="list-style-type: none"> • Una constante numérica • Una operación matemática • Otra variable previamente inicializada
If (condición) [órdenes]	<p>Realiza las instrucciones indicadas SI se cumple con la condición colocada.</p> <p>Ej. SI (10>2) [PoxY 10; ContinueRight 9;];</p> <p>La condición siempre será una operación de dos operandos y un operador.</p>
IfElse condición [instrucciones1] [instrucciones2]	<p>Realiza las primeras instrucciones si la condición expresada se cumple, y realiza el otro grupo de instrucciones sino se cumple la condición.</p> <pre>IfElse (var1=1) [Down; PosX 100;] [PosY 10; ContinueRight 9;];</pre> <p>Otro ejemplo</p> <pre>IfElse (Equal(10,2*5)) [Down; PosX 100;] [PosX 10; ContinueRight 9;];</pre>
Until [órdenes] [condición]	<p>Repite la lista de instrucciones tantas veces hasta que se cumpla la condición. Primero ejecuta el conjunto de instrucciones, de esta forma se asegura que las instrucciones se ejecutan al menos una vez antes de comprobar la condición. En caso de no cumplirse, se vuelve a ejecutar el conjunto de instrucciones.</p> <p>Ej.</p> <pre>Until [Down; PosX 100;] [bucle = 1];</pre> <pre>Until [Down; PosX 100;] [Equal(10,2*5)];</pre>

While [condición] [instrucciones]	<p>Repite la lista de instrucciones tanta veces hasta que se cumpla la condición. Si la condición expresada NO se cumple no se ejecutan las instrucciones ni una sola vez.</p> <pre>While [bucle=1] [Up; PosY 90];</pre> <pre>While [Equal(var1, 2*5)] [Down, PosX 90];</pre>
Equal (N1, N2)	<p>Devuelve TRUE si N1 y N2 son iguales, de lo contrario devuelve FALSO.</p> <p>Ej.</p> <pre>Equal(10, 2*5);</pre> <pre>Equal(10, 5);</pre> <pre>Equal(var1, 2*5);</pre> <p>Puede comparar valores, operaciones y/o variables.</p>
And (N1, N2)	<p>Devuelve TRUE si tanto la condición N1 como N2 son ciertos.</p> <p>Ej.</p> <pre>And (true,true);</pre> <pre>And(10>2) , (2>5) ;</pre> <p>Se pueden comparar valores u operaciones y/o variables.</p>
Or(N1, N2)	<p>Devuelve True si al menos una de las condiciones es cierta.</p> <p>Ej.</p> <pre>Or(10>2) , (2>5) ;</pre> <p>Se pueden comparar valores u operaciones y/o variables.</p>

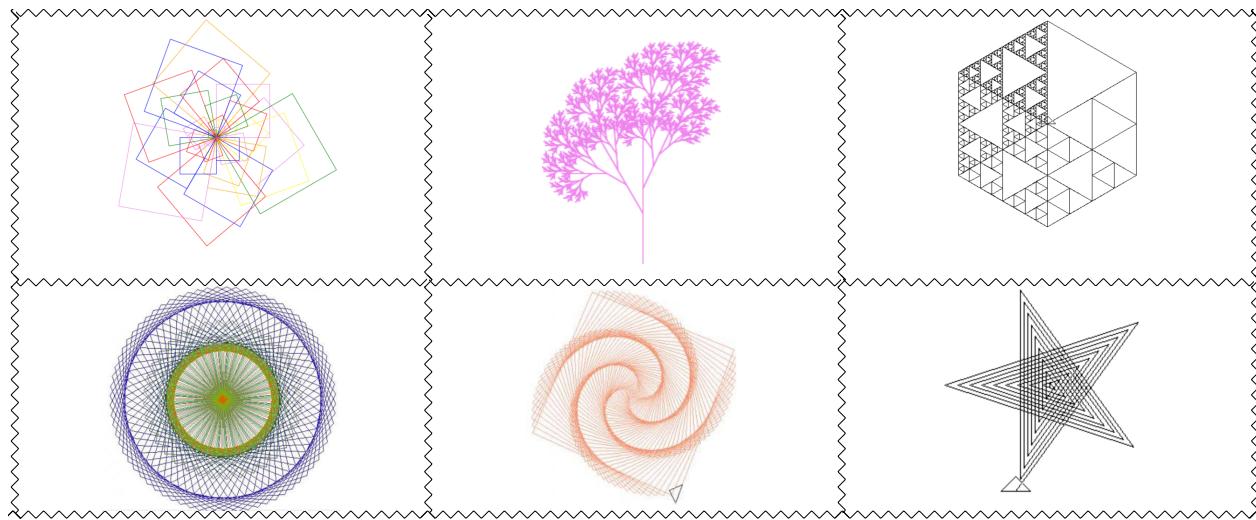
Greater(N1, N2)	Devuelve cierto si N1 es mayor a N2 Ej. <code>Greater(10, 2*5);</code> Se pueden comparar valores u operaciones y/o variables.
Smaller(N1, N2)	Devuelve cierto si N1 es menor a N2. Ej. <code>Smaller(10, 2*5);</code> Se pueden comparar valores u operaciones y/o variables.
Substr(N1, N2)	Restar números (n1 – n2 – n3 ...) Ej. <code>PosX Substr(100, 45);</code> Se pueden comparar valores u operaciones y/o variables.
Random(n)	Generar un número aleatorio comprendido entre 0 y n. Ej. <code>PosX Random(360);</code>
Mult(N1, N2)	Multiplicar números (N1 * N2 * N3 ...) Ej. <code>Mult(2, 5)</code> <code>Mult(2, Mult(5, 3));</code> Se pueden comparar valores u operaciones y/o variables.
Div (N1, N2)	Dividir dos números (N1 / N2) Ej. <code>Div(12, 4);</code> Se pueden utilizar valores, operaciones y/o variables.

<p>Sum(N1, N2)</p>	<p>Suma números.</p> <p>Ej. PosX Sum(1, 2);</p> <p>Se pueden utilizar valores, operaciones y/o variables.</p>
<p>PrintLine</p>	<p>PrintLine ("Hola Mundo");</p> <p>Esta operación genera un cuadro de dialogo a nivel del software presentando en la pantalla el texto colocado.</p> <p>Puede recibir como parámetro un texto, una variable o una constante.</p> <p>Puede recibir más de un parámetro a la vez.</p> <p>Def (variable1, 1);</p> <p>PrintLine ("Este es el proyecto número ", variable1, " de Compiladores ", 2022);</p> <p>debe imprimir en pantalla un cuadro de dialogo con algo similar a:</p> <p>"Este es el proyecto número 1 de Compiladores 2022"</p> <p>Esta sentencia es sumamente importante en la revisión del proyecto.</p>

Para la solución del problema del proyecto presentado se espera del estudiante:

- Fuerte investigación sobre las posibles soluciones en las distintas etapas del problema
- Búsqueda de distintas fuentes que servirán de insumo técnico-práctico para las soluciones así como la ayuda de otras áreas para cumplir con el objetivo del proyecto.
- Selección de criterios acordes al nivel de nuestro ambiente Tec en donde se seleccionen las mejores soluciones correspondientes.
- Una solución solvente de acuerdo con lo esperado.

Para efectos de la revisión del proyecto, uno de los rubros de evaluación será presentar el código suficiente para poder recrear lo más parecido posible a por lo menos una de las siguientes imágenes:



Documentación y aspectos operativos

→ Se deberá entregar un documento que contenga:

- ✓ Diagrama de arquitectura de la solución que refleje un nivel de detalle suficiente para entender a términos generales el funcionamiento del proyecto. Deben investigar cómo se hace un diagrama de arquitectura general a nivel profesional.
- ✓ Problemas conocidos: En esta sección se detalla cualquier problema que no se ha podido solucionar en el trabajo.
- ✓ Problemas encontrados: descripción detallada, intentos de solución sin éxito, soluciones encontradas con su descripción detallada, recomendaciones, conclusiones y bibliografía consultada para este problema específico.
- ✓ Conclusiones y Recomendaciones del proyecto. Conclusiones y recomendaciones con sentido y que confirmen una experiencia profunda en el proyecto.
- ✓ Bibliografía consultada en todo el proyecto

Atributos

Se debe entregar un documento aparte que contenga una portada y el siguiente detalle.

Debe comentar de qué manera se aplicaron los atributos que posee el curso y su impacto, esto por medio de una tabla en donde se detalle para cada atributo lo siguiente:

- Aplicación del atributo en la solución del proyecto
- Impacto del proyecto en la sociedad
- Retroalimentación obtenida gracias al proyecto

Favor colocar la información antes solicitada para cada uno de los atributos siguientes:

- **Conocimiento base de ingeniería**
- **Impacto de la ingeniería en la sociedad y el medio ambiente**
- **Aprendizaje continuo**

Conocimiento de ingeniería
Capacidad para aplicar los conocimientos a nivel universitario de matemáticas, ciencias naturales, fundamentos de ingeniería y conocimientos especializados de ingeniería para la solución de problemas complejos de ingeniería.
Medio ambiente y Sostenibilidad
Capacidad para comprender y evaluar la sostenibilidad y el impacto del trabajo profesional de ingeniería, en la solución de problemas complejos de ingeniería en los contextos sociales y ambientales.
Aprendizaje Continuo
Capacidad para reconocer las necesidades propias de aprendizaje y la habilidad de vincularse en un proceso de aprendizaje independiente durante toda la vida, en un contexto de amplio cambio tecnológico.

Evaluación

1. La implementación del dispositivo corresponde a un 40% de la nota final del proyecto.
2. El interprete corresponde a un 50% de la nota final del proyecto y su comunicación con la interfaz.
3. La documentación tendrá un valor de un 10% de la nota final del proyecto, cumplir con los puntos especificados en la documentación no significa que se tienen todos los puntos.
4. Cada grupo recibirá una nota en cada uno de los siguientes apartados:
 - a. Funcionalidad
 - b. Compilador
 - c. Documentación
5. De las notas mencionadas en el punto anterior se calculará la Nota Final del Proyecto.
6. No debe imprimirse la documentación, y esta debe encontrarse en formato PDF.
7. Documentación digital:
 8. Incluya dentro de su documentación todo el código fuente generado.
 9. Se debe "subir" toda la documentación al TecDigital
10. Las citas de revisión oficiales serán determinadas por el profesor durante las lecciones o mediante algún medio electrónico.
11. El profesor llevará un listado de todos y cada uno de los requerimientos mencionados en el presente enunciado del proyecto, de tal manera que se pueda evaluar cada uno de ellos y dar el puntaje determinado.
12. **El grupo debe contar con un grupo de procedimientos definidos previamente que demuestre los avances en la ejecución del proyecto.** Estos procedimientos deben cumplir con las reglas indicadas previamente en este documento.
13. El profesor deberá revisar el código para "recrear" alguna de las imágenes presentadas anteriormente. Podrá hacer cambios en el código para modificar la imagen del proyecto.
14. **El grupo deberá presentar un programa fuente** creado por ellos mismos en los cuales se realicen por lo menos 2 imágenes y 2 "letreros" con mensaje dibujado usando letras. Ej. TEC.
15. Dentro de la revisión del proyecto se deberá presentar la gramática creada y dar una explicación detallada de la manera en que se implementó el compilador mostrando todos los elementos necesarios para entender su funcionalidad.
16. Todos los integrantes del grupo deberán estar presente en la defensa del proyecto, y todos deben estar preparados para contestar las preguntas que se les puede realizar en base a la implementación realizada. Solamente en casos justificados y vistos previamente con por lo menos un día de anticipación se podrá ausentar algún miembro del equipo.
17. Aun cuando el código y la documentación tienen sus notas por separado, se aplican las siguientes restricciones
 - a. Si no se entrega documentación, automáticamente se obtiene una nota de 0.
 - b. Sí la documentación no se entregan en la fecha indicada se obtiene un 0.
 - c. Sí el código no compila se obtendrá una nota de 0, por lo cual se recomienda realizar la defensa con un código funcional.
18. La revisión de la documentación será realizada por parte del profesor, no durante la defensa del proyecto. El único requerimiento que se consultará durante la defensa del proyecto es el

- diagrama de arquitectura, documentación interna y cualquier otra documentación que el profesor considere necesaria.
19. Cada grupo tendrá como máximo 25 minutos para exponer su trabajo al profesor y realizar la defensa de éste, es responsabilidad de los estudiantes mostrar todo el trabajo realizado, por lo cual se recomienda tener todo listo antes de ingresar a la defensa, y será responsabilidad del grupo administrar el tiempo dispuesto para su revisión de tal manera que el profesor pueda observar todo el producto terminado.
 20. Todo error que salga durante la ejecución del proyecto y que se considere debió haber sido contemplada durante el desarrollo del proyecto, se castigará con 2 puntos de la nota final del proyecto.
 21. Cada grupo es responsable de llevar los equipos requeridos para la revisión, si no cuentan con estos deberán avisar al menos 2 días antes de la revisión a el profesor para coordinar el préstamo de estos.
 22. No se revisarán funcionalidades incompletas o no integradas.
 23. Durante la revisión únicamente podrán participar los miembros del grupo, asistentes, otros profesores y el coordinador del área.

Links de ayuda.

A continuación encontrará una serie de videos que podrían ser de ayuda. Solamente se muestran para efectos de ayuda, y de ninguna manera es una imposición de la forma de desarrollar la solución que cada equipo debe analizar y producir.

<https://youtu.be/nkO8--Zyl8w?list=RDCMUcaaKcUUrlKx1iLbLOASgzsQ>

https://youtu.be/S8YVIR_1hlo

<https://youtu.be/nkO8--Zyl8w?list=RDCMUcaaKcUUrlKx1iLbLOASgzsQ>

<https://youtu.be/bvSwznwqlHU?list=RDCMUcaaKcUUrlKx1iLbLOASgzsQ>