

**Instituto Nacional de Educación Diversificada**

**INED santa cruz naranjo**

**Catedra:** Producción

**Catedrático:** Gustavo Blanco

**Alumno**: Yenifer Danessa Gil Guillén

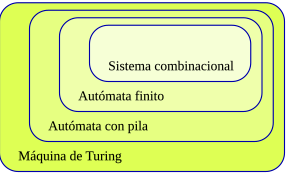
**Grado y carrera:** Quinto Baco

**Fecha:** 24/03/2025

**Tema:** investigación de producción

**Ciclo escolar:** 2025

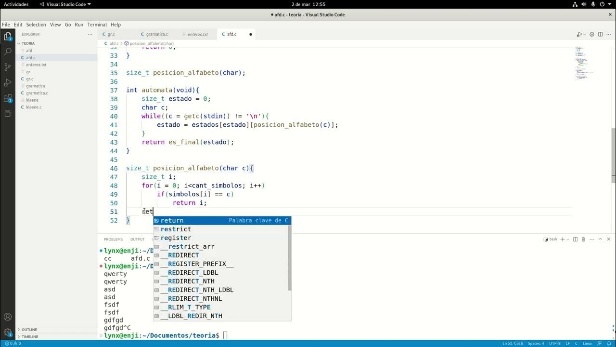
1. **Teoría de autómatas**

La **teoría de autómatas** es una rama de la teoría de la computación que estudia las máquinas abstractas y los problemas que éstas son capaces de resolver. La teoría de autómatas está estrechamente relacionada con la teoría del lenguaje formal ya que los autómatas son clasificados a menudo por la clase de lenguajes formales que son capaces de reconocer. También son de gran utilidad en la teoría de la complejidad computacional.

**Definición y usos de los autómatas finitos deterministas (AFD)**.

Un autómata finito determinista (abreviado AFD) es un autómata finito que además es un sistema determinista es decir, para cada estado en que se encuentre el autómata, y con cualquier símbolo del alfabeto leído, existe siempre no más de una transición posible desde ese estado y con ese símbolo.

Inicialmente un autómata finito determinista se encuentra en el estado inicial.q0A partir de un estado actual, el autómata utiliza la función de transición para determinar el siguiente estado La finalidad de los autómatas finitos es la de reconocer lenguajes regulares, que corresponden a los lenguajes formales más simples según la Jerarquía de Chomsky

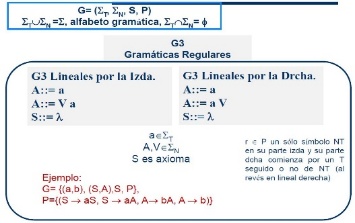


1. **Gramáticas formales**

Una **gramática formal** es una estructura lógico-matemática con un conjunto de reglas de formación que definen las cadenas de caracteres admisibles en un determinado lenguaje formal o lengua natural Las gramáticas formales aparecen en varios contextos diferentes: la lógica matemática, las ciencias de la computación y la lingüística teórica.

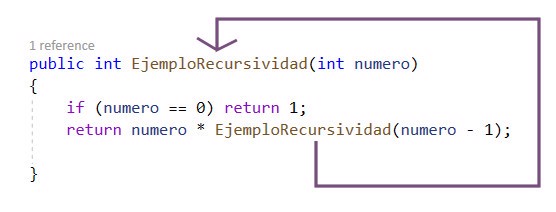
**Gramática regular**

En informática una gramática regular es una gramática formal (V, Σ, R, S) que puede ser clasificada como regular izquierda o regular derecha. **“Las gramáticas regulares sólo pueden generar a los lenguajes regulares de manera similar a los autómatas finitos y las expresiones regulares**”. Dos gramáticas regulares que generan el mismo lenguaje regular se denominan equivalentes. Toda gramática regular es una gramática libre de contexto.



**Gramática recursiva**

En informática**,” una gramática se denomina informalmente gramática recursiva si contiene reglas de producción que son recursivas”**, lo que significa que expandir un no terminal de acuerdo con estas reglas puede eventualmente conducir a una cadena que incluya el mismo no terminal nuevamente. Por ejemplo, una gramática para un lenguaje libre de contexto se deja recursiva si existe un símbolo no terminal *A* qué se puede pasar a través de las reglas de producción para producir una cadena con *A* (como el símbolo más a la izquierda) Todos los tipos de gramáticas en la jerarquía de Chomsky pueden ser recursivas y es la recursión la que permite la producción de conjuntos infinitos de palabras.



1. **Análisis léxico**

El análisis léxico es el primer paso del procesamiento de texto que se utiliza en muchos algoritmos de inteligencia artificial. Descubra por qué este proceso es un paso clave en el procesamiento del lenguaje natural, ya que permite que las máquinas comprendan el texto humano de manera más eficaz.

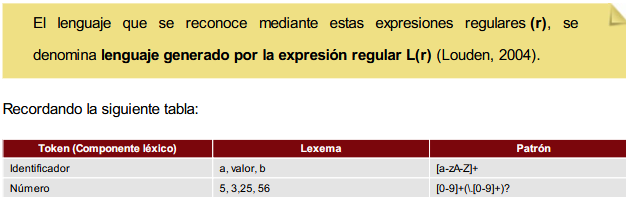
**Tokenización**

La **tokenización es un concepto que cada vez cobra más relevancia en el mundo de la tecnología, la economía y la sociedad.** **Se trata de un proceso que permite transformar cualquier tipo de activo o dato en una unidad digital llamada token, que puede ser almacenada, transferida y gestionada de forma segura y eficiente.**



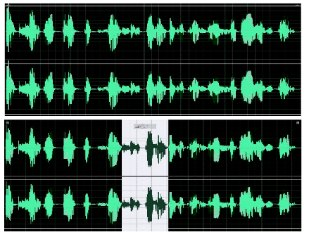
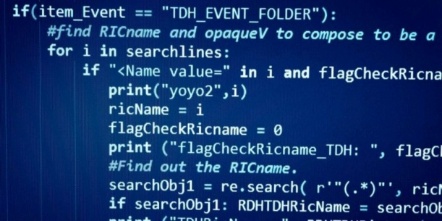
**Expresiones regulares**

Las expresiones regulares **son una forma de especificar patrones, entendiendo por patrón la forma de describir cadenas de caracteres**. Es la forma de definir los tokens o componentes léxicos y, como veremos, cada patrón concuerda con una serie de cadenas. **Se utilizan** **para identificar y extraer tokens (por ejemplo, palabras clave, identificadoras) del código**.



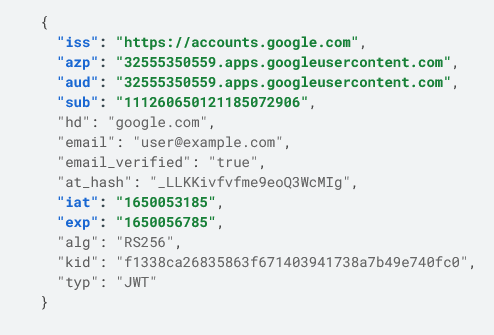
**Reconocimiento de palabras claves**

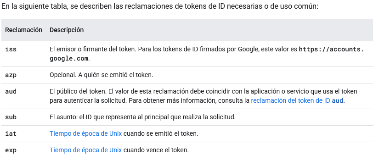
El reconocimiento de palabras clave, conocido en inglés como wordspotting, es una parte de la inteligencia artificial que consiste en identificar palabras específicas dentro de una locución. Es un algoritmo basado en la búsqueda de palabras clave en archivos de audio, por ejemplo llamadas telefónicas, donde localiza e indexa contenido de audio ganando un ahorro importante de tiempo ante la búsqueda manual. El sistema wordspotting se diferencia entre el reconocimiento de palabras aisladas y el reconocimiento de habla continua, es decir, las palabras son reconocidas en un flujo continuo de fonemas



**Identificación de tokens**: clasificación

En diferentes entornos, se usan diferentes tipos de tokens, existen los siguientes tipos de tokens:

* **Tokens de acceso:** Contienen información sobre el tipo de principal que se usó para crear el token
* **Tokens de ID:** Son tokens web que cumplen con la especificación de Open ID Connect (OIDC). Se componen de un conjunto de pares clave-valor llamados reclamaciones.
* **JWT autofirmados:** puedes usar JWT autofirmados para autenticarte en algunas APIs de Google sin tener que obtener un token de acceso del servidor de autorización.
* **Tokens de actualización:** administra la vida útil de los tokens de larga duración.
* **Tokens de actualización:** es una credencial que permite a las aplicaciones obtener nuevos tokens de acceso sin que el usuario deba volver a iniciar sesión.
* **Tokens del portador:** son un tipo de token que permite acceder a recursos asociados a ellos.



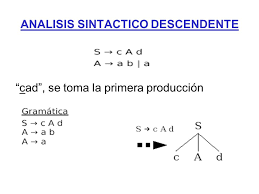
### ****Análisis sintáctico****

### El análisis sintáctico es el proceso de analizar e interpretar datos para comprender su estructura y significado

### 

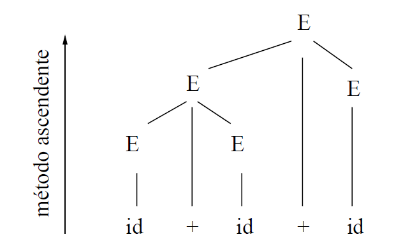
**Tipos de análisis sintáctico**

* **Análisis sintáctico descendente**

 Se puede considerar el análisis sintáctico descendente como un intento de encontrar una derivación por la izquierda para una cadena de entrada También se puede considerar como un intento de construir un árbol de análisis sintáctico para la entrada comenzando desde la raíz y creando los nodos del árbol en orden previo.

* **Análisis sintáctico ascendente**

El análisis ascendente recursivo es una técnica para implementar un analizador LR que utiliza funciones recursivas mutuas en lugar de tablas. Por lo tanto, el analizador se codifica directamente en el lenguaje anfitrión de manera similar al descenso recursivo.



**Árboles de análisis sintáctico**

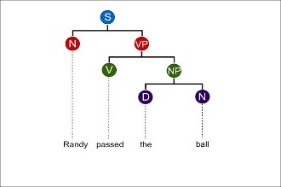
Los árboles de análisis sintáctico son estructuras de datos que representan la estructura de un programa o cadena de texto.

Son una representación gráfica del proceso de derivación de un programa

Son una estructura de datos usada extensamente en compiladores

Son el resultado del analizador sintáctico en la fase de un compilador

Tienen un impacto fuerte en la salida final del compilado



### ****5. Algoritmos de análisis de cadenas****

* **Algoritmo de Boyer-Moore**

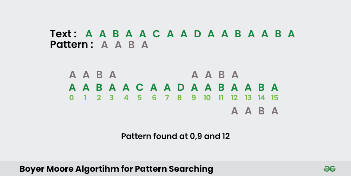
El algoritmo de Boyer-Moore es un método para buscar cadenas dentro de textos o programas. Es eficiente y se considera el estándar de referencia en la literatura de búsqueda de cadenas.

Se pre procesa la cadena que se busca.

Se alinea el patrón con el texto.

Se compara el patrón con el texto de derecha a izquierda.

Si hay una discrepancia, se utiliza la heurística de caracteres incorrectos y sufijos correctos para determinar cuánto debe desplazarse el patrón.

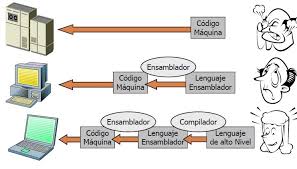


### ****6. Compiladores e intérpretes****

* **Estructura de un compilador**

Un compilador está compuesto por un front end y un back end. El front end analiza el código fuente, mientras que el back end genera el código objeto.

Un compilador está compuesto por componentes como analizadores, tablas y estructuras de datos. Su estructura se divide en dos partes: análisis y síntesis.



**Fases del compilador**:

* + **Análisis léxico**

Proceso que consiste en estudiar el vocabulario de un discurso, con el fin de identificar las palabras que se han utilizado.

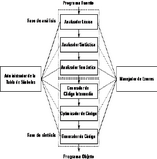
* + **Análisis sintáctico**

El análisis sintáctico es un proceso que identifica y clasifica las funciones de las palabras que forman una oración.

* + **Generación de código**

Un generador de código es una herramienta que produce código automáticamente.

* + **Optimización de código.**

Proceso que mejora la eficiencia de un programa informático. Esto se logra modificando el código para que use menos recursos, como memoria o tiempo de ejecución.

* **Lenguajes de programación utilizados en la creación de compiladores**: como **Lex y Yacc**.
* Ada.
* ALGOL. Algol 60. Algol 68. SMALL.
* BASIC.
* C. C++ Objetive-C. C# (a bytecode)
* D.
* CLEO.
* COBOL.
* Cobra.

### ****7. Aplicaciones prácticas de autómatas****

**Reconocimiento de patrones** en diferentes áreas.

Tiene aplicación en visión artificial, segmentación de imágenes, detección de objetos, procesamiento de datos de radar, reconocimiento de voz, clasificación de texto y mucho más. Existen dos métodos de clasificación en el reconocimiento de patrones: supervisada y no supervisada.

* **Procesamiento de lenguaje natural** y su uso en inteligencia artificial.

El procesamiento del lenguaje natural (PLN) es una rama de la inteligencia artificial (IA) que permite a las computadoras comprender, interpretar y generar lenguaje humano

* **Análisis de texto y minería de datos**: su importancia en el análisis de información.

Incrementa sus ingresos, recortar costos, mejorar sus relaciones con clientes, reducir riesgos y más.permite extraer información relevante de grandes cantidades de datos, lo que ayuda a tomar decisiones.

### ****8. Lenguaje de programación C y C++****

* **Librerías fundamentales**: **STDIO y CONIO**

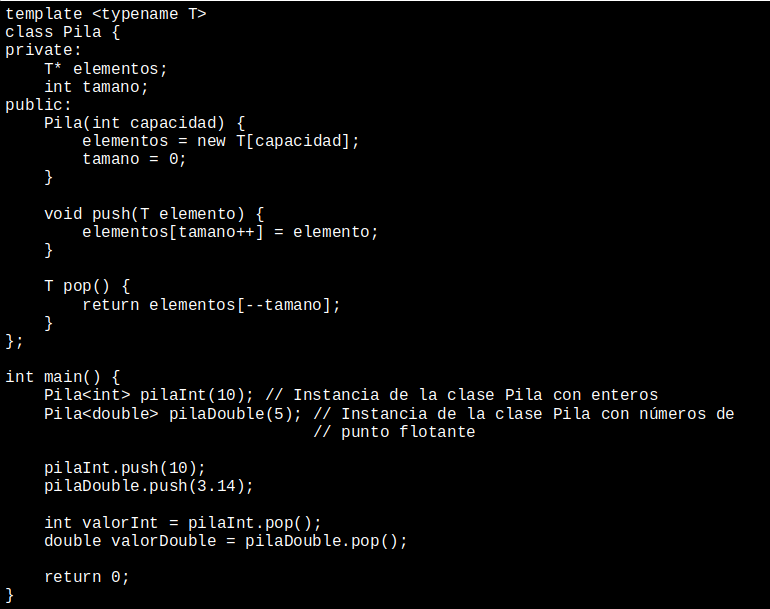
STDIO

* Significa "entrada/salida estándar"
* Contiene funciones como printf(), scanf(), getchar()
* Se comunica con STDIN y STDOUT
* Permite redirigir la salida a un archivo o a otro programa
* Es parte de la biblioteca estándar de C
* El lenguaje C++ también tiene su propia implementación de estas funciones

CONIO

* Significa "entrada/salida de consola"
* Contiene funciones como clrscr(), getch(), textcolor()
* Se comunica directamente con el hardware de vídeo
* Es más rápido que STDIO
* Se utiliza principalmente en sistemas DOS y algunos compiladores específicos
* Permite una interacción más directa con el usuario
* Es útil para escribir programas interactivos de pantalla completa
* **Lista de 10 librerías más comunes en C/C++**
* **iostream**: Especializada en la lectura y escritura de archivos
* **cmath**: Contiene funciones para operaciones matemáticas y transformaciones
* **cstring**: Contiene funciones para manipular elementos tipo string
* **ctime**: Contiene funciones para obtener y manipular información de tiempo y fecha
* **algorithm**: Contiene funciones para utilizarse en rangos de elementos
* **chrono**: Para usar mediciones de tiempo
* **complex**: Para usar números complejos
* **exception**: Excepciones estándar
* **functional**: Objetos para funciones
* **initializer\_list**: Para inicializar listas
* **Manejadores de formatos para diferentes tipos de datos**

Los gestores de **base de datos**, sistemas gestores de **base** de datos (SGBD) o DataBase Managenent System (DBMS) son sistemas que permiten la creación, gestión y administración de **bases de datos**, así como la elección y manejo de las estructuras necesarias para el almacenamiento y búsqueda de información.



Resumen de los títulos de la investigación

