**1. Teoría de autómatas**

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

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.

* **Ejemplos y aplicaciones** en informática y ciencias de la computación.

**son las que se ocupan del estudio de la computación, información y automatización**. Si bien tienen un componente teórico importante, también tienen aplicación y carácter práctico e incluso empírico, y su naturaleza es interdisciplinar.



**2. Gramáticas formales**

* **Gramáticas regulares**: concepto y ejemplos de su uso en lenguajes de programación.

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.

* **Gramáticas recursivas**: explicación y diferencias con las gramáticas regulares.

es un conjunto de reglas de producción que se pueden aplicar de forma recursiva Esto significa que al expandir un no terminal, se puede llegar a una cadena que incluya nuevamente el mismo no terminal.

**3. Análisis léxico**

* **Tokenización**: qué es y cómo se usa en el proceso de compilación.

proceso que transforma datos sensibles en datos no sensibles, llamados tokens. Se utiliza en diversos ámbitos, como la protección de datos, la economía, la tecnología y la sociedad.

* **Expresiones regulares**: su función en el análisis léxico.

secuencias de caracteres que permiten encontrar patrones en cadenas de texto. También se les conoce como RegEx, que es un acrónimo de "Regular Expressions"

* **Reconocimiento de palabras clave**: cómo se identifican en los lenguajes de programación.

tecnología que identifica palabras específicas en un archivo de audio. También se le conoce como wordspotting.

* **Identificación de tokens**: clasificación y ejemplos prácticos.

puede referirse a la identificación de tokens de autenticación, tokens digitales, o tokens en blockchain

**4. Análisis sintáctico**

* **Tipos de análisis sintáctico**:
	+ Análisis sintáctico descendente.

**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.
* **Árboles de análisis sintáctico**: qué son y cómo ayudan en la estructura de los programas.

**5. Algoritmos de análisis de cadenas**

* **Algoritmo de Boyer-Moore**: concepto, funcionamiento y su aplicación en la búsqueda de cadenas dentro de textos o programas.

**6. Compiladores e intérpretes**

* **Estructura de un compilador**: definición y componentes.
* **Fases del compilador**:
	+ Análisis léxico.
	+ Análisis sintáctico.
	+ Generación de código.
	+ Optimización de código.
* **Lenguajes de programación utilizados en la creación de compiladores**: como **Lex y Yacc**.

**7. Aplicaciones prácticas de autómatas**

* **Reconocimiento de patrones** en diferentes áreas.
* **Procesamiento de lenguaje natural** y su uso en inteligencia artificial.
* **Análisis de texto y minería de datos**: su importancia en el análisis de información.

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

* **Librerías fundamentales**: **STDIO y CONIO,** un resumen de las funciones de cada una.
* **Lista de 10 librerías más comunes en C/C++**, además de las ya mencionadas.
* **Manejadores de formatos para diferentes tipos de datos**, con ejemplos de uso en código.