

**Nombre de la asignatura:** Lenguajes y Autómatas I

**Créditos:** 2 – 3 – 5

### **Aportación al perfil**

- Desarrollar, implementar y administrar software de sistemas o de aplicación que cumpla con los estándares de calidad con el fin de apoyar la productividad y competitividad de las organizaciones.
- Integrar soluciones computacionales con diferentes tecnologías, plataformas o dispositivos.
- Diseñar e implementar interfaces hombre – máquina y máquina – máquina para la automatización de sistemas

### **Objetivo de aprendizaje**

- Aplicar las expresiones regulares, autómatas y gramáticas (elementos de la teoría de la computación) en la construcción de las fases de léxico y sintaxis de un compilador.

### **Competencias previas**

- Implementar pilas de objetos
- Implementar Árboles binarios
- Implementar Matrices de datos primitivas
- Manejar Notación e inducción matemática
- Manipular archivos de texto
- Manejar GUI'S en un lenguaje de programación

### **Temario**

- Introducción a la Teoría de Lenguajes Formales.
  - Alfabeto.
  - Cadenas.
  - Lenguajes.
  - Tipos de lenguajes
  - Herramientas computacionales ligadas con lenguaje
  - Estructura de un traductor
  - Fases de un Compilador
- Expresiones Regulares
  - Definición formal de una ER
  - Operaciones
  - Aplicaciones en problemas reales.

- Autómatas Finitos.
  - Definición formal
  - Clasificación de AF
  - Conversión de un AFND y AFD
  - Representación de ER usando AFND
  - Minimización de estados en un AF
  - Aplicaciones (definición de un caso de estudio)
  
- Máquinas de Turing
  - Definición formal MT
  - Construcción modular de una MT
  - Lenguajes aceptados por la MT.
  
- Análisis léxico.
  - Funciones del analizador léxico
  - Componentes léxicos, patrones y lexemas
  - Creación de Tabla de tokens
  - Errores léxicos
  - Generadores de analizadores Léxicos
  - Aplicaciones (Caso de estudio)
  
- Análisis Sintáctico
  - GLC
  - Árboles de derivación.
  - Formas normales de Chomsky.
  - Diagramas de sintaxis
  - Eliminación de la ambigüedad.
  - Generación de matriz predictiva ( cálculo first y follow)
  - Tipos de analizadores sintácticos
  - Manejo de errores
  - Generadores de analizadores sintácticos

### **Actividades de aprendizaje**

- Identificar alfabetos y lenguajes
- Conocer la función de cada traductor
- Conocer las fases de un compilador
- Obtener un alfabeto a partir de un lenguaje
- Determinar lexemas y componentes léxicos a partir de un lenguaje
  
- Conocer las expresiones regulares y sus operaciones
- Generar cadenas a partir de una expresión regular
- Obtener una expresión regular a partir de la descripción un caso
- Obtener una expresión regular a partir de un grupo de cadenas

- Conocer la notación formal de un AF
  - Conocer la diferencia entre un AFN y AFD
  - Conocer los algoritmos para convertir un AFN a AFD
  - Conocer la forma de minimización de estados en un AF
  - Construir un AF a partir de un ER
  - Construir un AF a partir de la descripción de un caso
  - Convertir un AFN a AFD
  - Minimizar estados en un AF
  - Determinar la notación formal a partir de un autómata
- 
- Conocer la notación formal de un MT
  - Construir una MT a partir de un caso
- 
- Determinar lexemas, componentes léxicos y patrones a partir de un lenguaje
  - Conocer los elementos de una tabla de tokens
  - Definir las reglas de un lenguaje de programación propio
  - Identificar patrones válidos, generar autómatas y tabla de tokens del lenguaje propuesto.
  - Construir un analizador léxico (utilizar un generador de analizador léxico o un LP)
- 
- Conocer la notación formal de una gramática
  - Conocer las formas normales de Chomsky
  - Conocer la notación de los diagramas de sintaxis
  - Construir diagramas de sintaxis de un lenguaje
  - Construir una GLC a partir de los diagramas de sintaxis
  - Eliminar la ambigüedad de una gramática
  - Construir un analizador sintaxis (utilizar un generador de analizador sintáctico o un LP)

### **Sugerencias didácticas transversales para el desarrollo de competencias profesionales**

- Propiciar actividades de búsqueda, selección y análisis de información en distintas fuentes.
- Propiciar el uso de las nuevas tecnologías en el desarrollo de los contenidos de la asignatura.
- Propiciar la planeación y organización del proceso de programación en CNC.

## **Prácticas.**

- Realizar un cuadro comparativo de los traductores que incluya ventajas, desventajas y características.
- Clasificar un lista de lenguajes, ambientes de desarrollo y utilerías en herramientas computacionales
- Clasificar componentes léxicos en un código de programa
- Obtener un alfabeto a partir de un lenguaje y viceversa.
- Obtener expresiones regulares a partir de casos de estudio.
- Realizar un programa que implemente una expresión regular
  
- Realizar programas que implemente lenguajes simples representados con AFD's
- Realizar ejercicios de construcción de AF a partir de ER o casos de estudio
- Realizar conversiones de AFN a AFD
  
- Construir MT a partir a partir de casos de estudio.
- Simular a través de un lenguaje de alto nivel, la representación de una máquina de Turing
  
- Definir las reglas de un lenguaje de programación propio
- Generar el autómata correspondiente al lenguaje definido
- Analizar la funcionalidad de diferentes generadores para análisis léxico de compilador.
- Realizar prácticas en algún generador para analizadores léxico.
- Construir un analizador léxico (utilizar un generador de analizador léxico o un LP)
  
- Construir diagramas de sintaxis para el lenguaje propuesto.
- Construir una GLC para el lenguaje propuesto.
- Analizar la funcionalidad de diferentes generadores para análisis sintáctico.
- Realizar prácticas en algún generador para analizadores sintáctico.
- Construir un analizador sintaxis (utilizar un generador de analizador sintáctico o un LP)

## **Criterios de evaluación:**

La evaluación de la asignatura se hará con base en siguiente desempeño:

- Desarrollar un proyecto final donde se aplique el manejo de expresiones regulares, autómatas y gramáticas formales para la

construcción de las fases del analizador léxico y sintáctico de un compilador.