PROGRAMA DE LABORATORIO

UNIVERSIDAD DE SAN CARLOS DE GUATEMALA
FACULTAD DE INGENIERÍA
ESCUELA DE CIENCIAS Y SISTEMAS

Lenguajes Formales y De Programación

CÓDIGO:	**796**	**PONDERACIÓN:**	**-**
ESCUELA DE INGENIERÍA EN:	**CIENCIAS Y SISTEMAS**	ÁREA A LA QUE PERTENECE:	**DESARROLLO DE SOFTWARE**
PRE-REQUISITO:	**770 – Introducción a la Programación y Computación 1** **795 – Lógica de Sistemas** **960 – Matemática de Cómputo 1**	POST REQUISITO:	**777 - Organización de Lenguajes y Compiladores 1** **772 - Estructuras de Datos** **0778 – Arquitectura de Computadoras y Ensambladores 1**
CATEGORÍA:	**OBLIGATORIO**	VIGENCIA:	**SEGUNDO SEMESTRE 2025**
HORAS POR SEMANA DEL CURSO:	**2 hrs**	HORAS POR SEMANA DEL LABORATORIO:	**2 hrs**
HORAS DE AUTOAPRENDIZAJE:	**2 hrs**	TOTAL DE HORAS DE APRENDIZAJE:	**22 hrs**
CATEDRÁTICO (A):	**ING. ZULMA KARINA AGUIRRE ORDONEZ**	AUXILIAR:	**RIVER ANDERSON ISMALEJ ROMAN**
EDIFICIO:	**T3**	SECCIÓN:	**B-**
SALÓN DEL CURSO:	**Hibrido**	SALON DEL LABORATORIO:	**Virtual**
DÍAS QUE SE IMPARTE EL CURSO:	**Martes**	DIAS QUE SE IMPARTE EL LABORATORIO:	**Martes**
HORARIO DEL CURSO:	**7:10 – 8:50**	HORARIO DEL LABORATORIO:	**8:50 – 10:30**

Breve descripción del Laboratorio

El laboratorio de Lenguajes Formales y De Programación tiene como propósito introducir al estudiante de ciencias de la computación al estudio, análisis y comprensión de lenguajes de programación bajo una estructura que contribuya al desarrollo de las capacidades de manejo y diseño de gramáticas. Se abordan conocimientos de modelos matemáticos que resuelven problemas en lenguajes formales y se aplican a lenguajes reales conocidos como JavaScript.

Índice

Competencias Vinculadas al Perfil del Egresado

Competencias Específicas

No.	Competencia
1	Aplica los conocimientos de su disciplina en la elaboración, fundamentación y defensa de argumentos para prevenir y resolver problemas complejos en su campo profesional, identificando y aplicando innovaciones.
2	Demuestra destreza y habilidad en la selección, uso y adaptación de herramientas metodológicas, tecnológicas, equipos especializados y en la lectura e interpretación de datos, pertinentes al contexto de su ejercicio profesional.
3	Toma decisiones profesionales con base en fundamentos teóricos, datos e información pertinente, válida y confiable.

Competencias Generales

No.	Competencia
1	Aplica principios básicos de ingeniería, ciencias de computación y sistemas de información y comunicación, en la formulación y resolución adecuada de problemas complejos.
2	Aplicar los conocimientos adquiridos en lenguajes formales y programación en contextos reales.
3	Desarrollar habilidades técnicas de programación y análisis de sistemas.

Competencias del Laboratorio

Competencia(s) Específica(s)

No.	Competencia	Nivel de Aprendizaje
1	El estudiante construye autómatas finitos y gramáticas formales mediante técnicas de análisis léxico y sintáctico para una interpretación acertada de entradas varias.	Analizar
2	El estudiante genera gramáticas libres de contexto mediante el análisis sintáctico para reconocer lenguajes formales.	Comprender

3	El estudiante aplica el análisis léxico mediante el uso de expresiones regulares para solucionar problemas que requieran el reconocimiento de lenguajes.	Aplicar
4	El estudiante aplica el analisis sintactico mediante el uso de gramáticas formales y técnicas de parsing (descendente o ascendente) para verificar la estructura gramatical y construir árboles sintácticos que permitan solucionar problemas relacionados con la interpretación de lenguajes formales.	Aplicar
5	El estudiante comprende el funcionamiento del lenguaje de programación JavaScript a través de su aplicación para la resolución de problemas relacionados con las fases del compilador.	Comprende

Competencia(s) General(es)

No.	Competencia	Nivel de Aprendizaje
	El estudiante construye aplicaciones mediante el uso del análisis léxico y sintáctico utilizando herramientas de procesamiento de lenguajes formales para dar solución a enunciados que requieren el reconocimiento de lenguajes.	Crear
	Aplicar los conocimientos adquiridos en lenguajes formales y programación en contextos reales.	Aplicar
	Desarrollar habilidades técnicas de programación y análisis de sistemas.	Aplicar

Diseño Didáctico por Competencias

Esta sección organiza las sesiones del laboratorio en función de las competencias que el

estudiante debe desarrollar. Cada clase incluye valores (saber ser), contenidos teóricos (saber) y habilidades prácticas (saber hacer), permitiendo un aprendizaje integral y aplicado. Las actividades están alineadas con los objetivos del curso y el perfil del egresado.

Unidad 1: JavaScript

- 1.1 Introducción al lenguaje JavaScript
  - 1.1.1 Historia de JavaScript
  - 1.1.2 Aspectos básicos de JavaScript
  - 1.1.3 Reglas de sintaxis
  - 1.1.4 Estructura de un programa
- 1.2 Buenas prácticas y funciones en JavaScript
  - 1.2.1 Convenciones de nombres y estilo de código
  - 1.2.2 Gestión de memoria y optimización
  - 1.2.3 Definición y uso de métodos y funciones
  - 1.2.4 Creación y uso de clases
  - 1.2.5 Diferencias entre métodos y funciones
- 1.3 Estructuras de datos y control en JavaScript
  - 1.3.1 Arreglos en JavaScript
  - 1.3.2 Diccionarios y otras estructuras de datos
  - 1.3.3 Iteración en JavaScript
  - 1.3.4 Lectura de archivos
  - 1.3.5 Escritura de archivos

Unidad 2: Lenguajes Formales

- 2.1 Lenguajes
  - 2.1.1 Lenguaje Natural
  - 2.1.2 Lenguajes Formales
  - 2.1.3 Lenguajes de Programación
- 2.2 Evolución de los Lenguajes de Programación
  - 2.2.1 Paradigmas
  - 2.2.2 Generaciones de lenguajes de programación
- 2.3 Procesadores de Lenguaje
  - 2.3.1 Intérprete
  - 2.3.2 Compilador

        - 2.3.2.1 Estructura de un compilador
    - 2.3.3 Diferencias y Ejemplos
    - 2.3.4 Herramientas
- 2.4 Jerarquía de Chomsky
    - 2.4.1 Clasificación de gramáticas

Unidad 3: Análisis Léxico

- 3.1 Definición y Función del Analizador Léxico
    - 3.1.1 Patrones
    - 3.1.2 Tokens
    - 3.1.3 Lexemas
    - 3.1.4 Errores léxicos
    - 3.1.5 Ejemplos de tokens en diferentes lenguajes
- 3.2 Operaciones entre Lenguajes y Expresiones Regulares
    - 3.2.1 Interoperabilidad entre lenguajes
    - 3.2.2 Concepto de interoperabilidad
    - 3.2.3 Traducción y compilación: proceso de traducción de un lenguaje a otro
    - 3.2.4 Buenas prácticas en interoperabilidad
    - 3.2.5 ¿Qué son las expresiones regulares?
    - 3.2.6 Aplicaciones de expresiones regulares en análisis léxico
    - 3.2.7 Diagrama de transición de estados
    - 3.2.8 Tablas de transición
- 3.3 Autómatas Finitos
    - 3.3.1 Definición de AFN
    - 3.3.2 Estructura de un AFN
    - 3.3.3 Ejemplo de AFN
    - 3.3.4 Definición de AFD
    - 3.3.5 Estructura de un AFD
    - 3.3.6 Ejemplo de AFD
- 3.4 Conversión AFN a AFD
    - 3.4.1 Método del Árbol
    - 3.4.2 Anulables
    - 3.4.3 Cálculo de First, Last y Next

- 3.5 Optimización de Estados
    - 3.5.1 Minimización de un AFD
    - 3.5.2 Identificación de estados equivalentes
    - 3.5.3 Representación compacta de AFD
    - 3.5.4 Ventajas de una representación optimizada

Unidad 4: Análisis Sintáctico

- 4.1 Función del Analizador Sintáctico
    - 4.1.1 Definición de análisis sintáctico
    - 4.1.2 Propósito del analizador sintáctico en un compilador
    - 4.1.3 Diferencias entre análisis léxico y sintáctico
    - 4.1.4 Analizadores sintácticos ascendentes
    - 4.1.5 Analizadores sintácticos descendentes
    - 4.1.6 Comparación de técnicas de análisis sintáctico
- 4.2 Lenguajes Libres de Contexto (LFC)
    - 4.2.1 Definición de lenguajes libres de contexto
    - 4.2.2 Características de los LFC
    - 4.2.3 Importancia de los LFC en la teoría de lenguajes de programación
    - 4.2.4 Gramáticas Tipo 2
    - 4.2.5 Ejemplos de generación de cadenas usando gramáticas Tipo 2
    - 4.2.6 Aplicaciones prácticas de gramáticas Tipo 2 en lenguajes de programación
- 4.3 Árboles de Derivación y Ambigüedad
    - 4.3.1 Árboles de derivación
    - 4.3.2 Definición y construcción de árboles de derivación
    - 4.3.3 Ambigüedad
    - 4.3.4 Concepto de ambigüedad en gramáticas
    - 4.3.5 Ejemplos de gramáticas ambiguas
    - 4.3.6 Recursividad
    - 4.3.7 Definición de recursividad en gramáticas
    - 4.3.8 Ejemplos de gramáticas recursivas
- 4.4 Autómatas de Pila
    - 4.4.1 Introducción a los autómatas de pila
    - 4.4.2 Definición y funcionamiento de un autómata de pila

- 4.4.3 Diferencias entre autómatas de pila y autómatas finitos
- 4.4.4 Procesamiento en un autómata de pila
- 4.4.5 Tipos de aceptación (estado final vs. pila vacía)
- 4.4.6 Estado final
- 4.4.7 Pila vacía
- 4.4.8 Autómata de pila generado desde Gramática Tipo 2

Sesión de Diagnóstico

Evaluación de conocimientos previos

Se aplicará una actividad diagnóstica con el objetivo de identificar el nivel de conocimientos y habilidades que los estudiantes poseen al inicio del curso. No influye en la nota final, pero es obligatoria para todos los estudiantes.

Tipo de Actividad	Descripción
Cuestionario de preguntas.	Realización de un cuestionario de preguntas directas para saber con que conocimientos cuentan los estudiantes antes de iniciar el curso.

Presentación del tutor

El tutor se presenta formalmente al grupo, compartiendo su formación académica, experiencia profesional y educativa, así como sus expectativas sobre el curso. También se abordan aspectos como normas de convivencia, canales de comunicación, disponibilidad para consultas y métodos de acompañamiento.

Presentación de los estudiantes

Se escogen un grupo de estudiantes al azar. En su presentación, se les pedirá que compartan información básica como su nombre, intereses personales o profesionales, experiencias previas relacionadas con el curso y sus expectativas. Esta actividad busca promover la interacción, el reconocimiento entre pares y la construcción de un entorno participativo y respetuoso.

Presentación del programa del curso

Se presenta el contenido del programa del curso, se aclaran dudas y se fomenta el compromiso del estudiante con su aprendizaje.

Evaluación de conocimientos del laboratorio actual

Se realiza una evaluación o práctica que permite conocer el grado de familiaridad de los estudiantes con las herramientas, entornos o competencias técnicas necesarias para el laboratorio actual.

Tipo de Actividad	Descripción
Ejercicio práctico en JavaScript y VSCode	Ejercicio práctico en JavaScript y utilizando el entorno de desarrollo VSCode para conocer

	qué tan familiarizado este el estudiante con este entorno y que tan bien conoce el lenguaje de programación JavaScript.

Sesión No. 1, Unidad No. 1 - JavaScript

Valor de la semana (Saber ser)

Disciplina: Se requiere precisión para respetar normas y estructuras del lenguaje.

Nombre:
Introducción al lenguaje JavaScript

Conocimiento (Saber)

Competencia(s)	
El estudiante comprende el funcionamiento del lenguaje de programación Java a través de su aplicación para la resolución de problemas relacionados con las fases del compilador.	
Tema	Subtema
Introducción al lenguaje JavaScript	Historia de JavaScript
Introducción al lenguaje JavaScript	Aspectos Básicos de JavaScript
Introducción al lenguaje JavaScript	Reglas de Sintaxis
Introducción al lenguaje JavaScript	Estructura de un programa

Habilidades (Saber Hacer)

Competencia	Tipo de Actividad	Ponderación
El estudiante comprende el funcionamiento del lenguaje de programación JavaScript a través de su aplicación para la resolución de problemas relacionados con las fases del compilador.	Ejercicio de programación básica en JavaScript	

Sesión No. 2, Unidad No. 1 - JavaScript

Valor de la semana (Saber ser)

Disciplina: Fomenta el respeto por normas y estándares que garantizan claridad en el código.

Nombre:
Buenas Prácticas y funciones en JavaScript

Conocimiento (Saber)

Competencia(s)	
El estudiante comprende el funcionamiento del lenguaje de programación Java a través de su aplicación para la resolución de problemas relacionados con las fases del compilador.	
Tema	Subtema
Buenas Prácticas y funciones en JavaScript	Convenciones de nombres y estilo de código
Buenas Prácticas y funciones en JavaScript	Gestión de memoria y optimización
Buenas Prácticas y funciones en JavaScript	Definición y uso de métodos y funciones
Buenas Prácticas y funciones en JavaScript	Creación y uso de clases
Buenas Prácticas y funciones en JavaScript	Diferencias entre métodos y funciones

Habilidades (Saber Hacer)

Competencia	Tipo de Actividad	Ponderación
El estudiante comprende el funcionamiento del lenguaje de programación JavaScript a través de su aplicación para la resolución de problemas relacionados con las fases del compilador.	Desarrollo de una aplicación pequeña utilizando buenas prácticas	

Sesión No. 3, Unidad No. 1 - JavaScript

Valor de la semana (Saber ser)

Pensamiento Lógico: Es necesario razonar paso a paso el proceso completo del funcionamiento de un programa y cómo se almacenan los datos.

Nombre:
Estructuras de Datos y Control en JavaScript

Conocimiento (Saber)

Competencia(s)	
El estudiante comprende el funcionamiento del lenguaje de programación Java a través de su aplicación para la resolución de problemas relacionados con las fases del compilador.	
Tema	Subtema
Estructuras de Datos y Control en JavaScript	Arreglos en JavaScript
Estructuras de Datos y Control en JavaScript	Diccionarios y estructuras de Datos
Estructuras de Datos y Control en JavaScript	Iteración en JavaScript
Estructuras de Datos y Control en JavaScript	Lectura de Archivos
Estructuras de Datos y Control en JavaScript	Escritura de Archivos

Habilidades (Saber Hacer)

Competencia	Tipo de Actividad	Ponderación
El estudiante comprende el funcionamiento del lenguaje de programación JavaScript a través de su aplicación para la resolución de problemas relacionados con las fases del compilador.	Desarrollo de un programa para lectura de archivos	

Sesión No. 4, Unidad No. 2 – Lenguajes Formales

Valor de la semana (Saber ser)

Pensamiento Crítico: Los lenguajes de programación y su evolución no son solo una cuestión de memorizar reglas, sino también de comprender por qué ciertas decisiones se tomaron en el diseño de lenguajes.

Nombre:
Lenguajes

Conocimiento (Saber)

Competencia(s)	
El estudiante analiza la evolución y clasificación de los lenguajes formales y de programación mediante la comparación de sus características, estructuras y herramientas para comprender su organización y funcionamiento.	
Tema	Subtema
Lenguajes	Lenguajes Natural
Lenguajes	Lenguajes Formales
Lenguajes	Lenguajes de Programación
Lenguajes	Evolución de los Lenguajes de Programación
Lenguajes	Paradigmas
Lenguajes	Generaciones de lenguajes de programación

Habilidades (Saber Hacer)

Competencia	Tipo de Actividad	Ponderación
El estudiante analiza la evolución y clasificación de los lenguajes formales y de programación mediante la comparación de sus características, estructuras y herramientas para comprender su organización y funcionamiento	Actividad de Investigación sobre Lenguajes	
El estudiante analiza la evolución y clasificación de los	Diseño de	

lenguajes formales y de programación mediante la comparación de sus características, estructuras y herramientas para comprender su organización y funcionamiento	una Gramática Formal y su Implementación	

Sesión No. 5, Unidad No. 2 - Lenguajes Formales

Valor de la semana (Saber ser)

Pensamiento Lógico y Analítico: Es esencial para comprender cómo se procesan las instrucciones, cómo se generan los errores y cómo se optimizan los procesos de traducción de lenguajes.

Nombre:
Procesadores de Lenguaje

Conocimiento (Saber)

Competencia	
El estudiante analiza la evolución y clasificación de los lenguajes formales y de programación mediante la comparación de sus características, estructuras y herramientas para comprender su organización y funcionamiento.	
Tema	Subtema
Procesadores de Lenguaje	Interprete
Procesadores de Lenguaje	Compilador
Procesadores de Lenguaje	Estructura de un compilador
Procesadores de Lenguaje	Diferencias y Ejemplos
Procesadores de Lenguaje	Herramientas
Procesadores de Lenguaje	Jerarquía de Chomsky

Habilidades (Saber Hacer)

Competencia	Tipo de Actividad	Ponderación
El estudiante analiza la evolución y clasificación de los	Prueba	

lenguajes formales y de programación mediante la comparación de sus características, estructuras y herramientas para comprender su organización y funcionamiento.	Teórica sobre la Jerarquía de Chomsky •	

Sesión No. 6, Unidad No. 3- Análisis Léxico

Valor de la semana (Saber ser)

Claridad: Porque identificar lexemas y tokens requiere expresiones regulares bien definidas y sin ambigüedad.

Nombre:
Definición y Función del analizador léxico Operaciones Entre Lenguajes y Expresiones regulares

Conocimiento (Saber)

Competencia	
El estudiante aplica el análisis léxico mediante el uso de expresiones regulares para solucionar problemas que requieran el reconocimiento de lenguajes.	
Tema	Subtema
Definición y Función del analizador léxico	Patrones
Definición y Función del analizador léxico	Tokens
Definición y Función del analizador léxico	Lexemas
Definición y Función del analizador léxico	Errores Léxicos
Definición y Función del analizador léxico	Ejemplos de tokens en diferentes lenguajes
Operaciones Entre Lenguajes y Expresiones regulares	Interoperabilidad entre Lenguajes
Operaciones Entre Lenguajes y Expresiones regulares	Concepto de interoperabilidad
Operaciones Entre Lenguajes y Expresiones regulares	Traducción y Compilación: Procesó de traducción de un lenguaje a otro

Operaciones Entre Lenguajes y Expresiones regulares	Buenas Prácticas en Interoperabilidad
Operaciones Entre Lenguajes y Expresiones regulares	¿Qué son las expresiones regulares?
Operaciones Entre Lenguajes y Expresiones regulares	Aplicaciones de Expresiones Regulares en Análisis Léxico
Operaciones Entre Lenguajes y Expresiones regulares	Diagrama de transición de estados

Habilidades (Saber Hacer)

Competencia	Tipo de Actividad	Ponderación
El estudiante aplica el análisis léxico mediante el uso de expresiones regulares para solucionar problemas que requieran el reconocimiento de lenguajes.	Actividad: Escribir y depurar expresiones regulares en Java.	

Sesión No. 7, Unidad No. 3 - Análisis Léxico

Valor de la semana (Saber ser)

Responsabilidad: Al asumir el compromiso de definir correctamente las expresiones, considerando su impacto en la correcta interpretación del lenguaje.

Nombre:
Autómata Finitos

Conocimiento (Saber)

Competencia
El estudiante aplica el análisis léxico mediante el uso de expresiones regulares para solucionar problemas que requieran el reconocimiento de lenguajes.

Tema	Subtema
Autómata Finitos	Definición de AFN
Autómata Finitos	Estructura de un AFN
Autómata Finitos	Ejemplo de AFN
Autómata Finitos	Definición de AFD
Autómata Finitos	Estructura de un AFD
Autómata Finitos	Ejemplo de AFD

Habilidades (Saber Hacer)

Competencia	Tipo de Actividad	Ponderación
El estudiante aplica el análisis léxico mediante el uso de expresiones regulares para solucionar problemas que requieran el reconocimiento de lenguajes.	Practica: Construir AFN en papel, convertirlos a AFD en clase y luego implementar el AFD en código.	

Sesión No. 8, Unidad No. 3 - Análisis Léxico

Valor de la semana (Saber ser)

Disciplina: Al aplicar de forma ordenada las técnicas de conversión y análisis léxico dentro del tiempo asignado.

Nombre:
Conversión AFN a AFD

Conocimiento (Saber)

Competencia
El estudiante aplica el análisis léxico mediante el uso de expresiones regulares para solucionar problemas que requieran el reconocimiento de lenguajes.

Tema	Subtema
Conversión AFN a AFD	Método del Árbol
Conversión AFN a AFD	Anulables
Conversión AFN a AFD	Cálculo de First, Last y Next

Habilidades (Saber Hacer)

Competencia	Tipo de Actividad	Ponderación
El estudiante aplica el análisis léxico mediante el uso de expresiones regulares para solucionar problemas que requieran el reconocimiento de lenguajes.	Practica: Dibujar el árbol sintáctico de una expresión regular dada.	

Sesión No. 9, Unidad No. 3 - Análisis Léxico

Valor de la semana (Saber ser)

Precisión: Aplica con exactitud los pasos para minimizar autómatas sin perder el lenguaje reconocido.

Nombre:
Optimización de Estados

Conocimiento (Saber)

Competencia
El estudiante aplica técnicas de minimización de autómatas finitos deterministas (AFD) mediante la identificación de estados equivalentes y particiones sucesivas para reducir el

número de estados sin alterar el lenguaje reconocido.	
Tema	**Subtema**
Optimización de Estados	Minimización de un AFD
Optimización de Estados	Identificación de Estados Equivalentes
Optimización de Estados	Representación Compacta de AFD
Optimización de Estados	Ventajas de una representación optimizada

Habilidades (Saber Hacer)

Competencia	**Tipo de Actividad**	**Ponderación**
El estudiante aplica técnicas de minimización de autómatas finitos deterministas (AFD) mediante la identificación de estados equivalentes y particiones sucesivas para reducir el número de estados sin alterar el lenguaje reconocido.	Practica: Generar el diagrama del AFD en formato DOT.	

Sesión No. 10, Unidad No. 4 - Análisis Sintáctico

Valor de la semana (Saber ser)

Curiosidad intelectual: Investiga formas de representar correctamente un lenguaje.

Nombre:
Función del Analizador Sintáctico Lenguajes Libres de Contexto Arboles de Derivación y Ambigüedad

Conocimiento (Saber)

Competencia

El estudiante genera gramáticas libres de contexto mediante el análisis sintáctico para reconocer lenguajes formales.	
Tema	**Subtema**
Función del Analizador Sintáctico	Definición de análisis sintáctico
Función del Analizador Sintáctico	Propósito del analizador sintáctico en un compilador
Función del Analizador Sintáctico	Diferencias entre análisis léxico y sintáctico
Función del Analizador Sintáctico	Analizadores sintácticos ascendentes
Función del Analizador Sintáctico	Analizadores sintácticos descendentes
Lenguajes Libres de Contexto	Definición de lenguajes libres de contexto (LFC)
Lenguajes Libres de Contexto	Características de LFC
Lenguajes Libres de Contexto	Importancia de los LFC en la teoría de lenguajes de programación
Lenguajes Libres de Contexto	Gramáticas Tipo 2
Lenguajes Libres de Contexto	Ejemplos de generación de cadenas usando gramáticas Tipo 2
Lenguajes Libres de Contexto	Aplicaciones prácticas de gramáticas Tipo 2 en lenguajes de programación
Arboles de Derivación y Ambigüedad	Árboles de derivación

Habilidades (Saber Hacer)

Competencia	**Tipo de Actividad**	**Ponderación**
El estudiante genera gramáticas libres de contexto mediante el análisis sintáctico para reconocer lenguajes formales.	Practica: Tomar una gramática ambigua y reescribirla para eliminar la ambigüedad.	

Sesión No. 11, Unidad No. 4 - Análisis Sintáctico

Valor de la semana (Saber ser)

Autonomía: Desarrolla aplicaciones de análisis sintáctico de forma independiente.

Nombre:
Arboles de Derivación y Ambigüedad Autómatas de Pila

Conocimiento (Saber)

Competencia	
El estudiante construye aplicaciones mediante el uso del análisis léxico y sintáctico para dar solución a enunciados que requieren el reconocimiento de lenguajes	
Tema	**Subtema**
Arboles de Derivación y Ambigüedad	Definición y construcción de árboles de derivación
Arboles de Derivación y Ambigüedad	Ambigüedad
Arboles de Derivación y Ambigüedad	Concepto de ambigüedad en gramáticas
Arboles de Derivación y Ambigüedad	Ejemplos de gramáticas ambiguas
Arboles de Derivación y Ambigüedad	Recursividad
Arboles de Derivación y Ambigüedad	Definición de recursividad en gramáticas
Arboles de Derivación y Ambigüedad	Ejemplos de gramáticas recursivas
Autómatas de Pila	Introducción a los autómatas de pila
Autómatas de Pila	Definición y funcionamiento de un autómata de pila
Autómatas de Pila	Diferencias entre autómatas de pila y autómatas finitos
Autómatas de Pila	Procesamiento
Autómatas de Pila	Tipos de aceptación
Autómatas de Pila	Estado Final
Autómatas de Pila	Pila vacía
Autómatas de Pila	Autómata de Pila desde Gramática Tipo 2

Habilidades (Saber Hacer)

Competencia	Tipo de Actividad	Ponderación
El estudiante construye aplicaciones mediante el uso del análisis léxico y sintáctico para dar solución a enunciados que requieren el reconocimiento de lenguajes	Practica: Dada una gramática Tipo 2, construir el PDA correspondiente según el teorema "PDA desde Gramática"	

Tiempo de Auto-aprendizaje

Tipo	Horas de Auto-aprendizaje
Proyectos	200
Prácticas	15
Tareas	10
Total	**225**

Rúbrica de Evaluación

Cada una de las actividades del laboratorio (proyectos, prácticas, tareas y otras) cuenta con una rúbrica de evaluación específica, la cual está detallada en el documento que se entrega al estudiante al momento de asignar la actividad. Estas rúbricas describen los criterios de evaluación, niveles de desempeño esperados y la ponderación correspondiente de cada aspecto evaluado.

Es responsabilidad del estudiante leer detenidamente la rúbrica asignada antes de iniciar el desarrollo de la actividad. Comprender los criterios de evaluación no solo permite orientar adecuadamente el trabajo, sino también mejorar el desempeño académico y fomentar la autorregulación del aprendizaje.

En caso de no recibir la rúbrica al momento de la asignación, el estudiante debe solicitarla directamente al tutor académico, ya que constituye una herramienta esencial para el cumplimiento de los objetivos de aprendizaje y la evaluación transparente.

Resumen de Ponderaciones

Tipo	Valor
Actividades en Clase	6
Proyectos	65
Prácticas	15
Tareas	4
Examen Final	10
Total	**100**

Normativa Académica y Ética del Curso

En concordancia con el perfil del estudiante de la Facultad de Ingeniería de la Universidad de San Carlos de Guatemala, se espera un alto nivel de compromiso con la excelencia académica y la ética profesional. Por ello, que se establece los siguientes lineamientos de carácter obligatorio que regulan el comportamiento académico del estudiante:

Plagio y copias

- Todo proyecto será sometido a verificación para confirmar su autoría y originalidad, con la finalidad de evitar cualquier plagio, copia o que la actividad no haya sido realizada por el estudiante.
- Cualquier evidencia de lo antes descrito en las distintas actividades será sancionada con una calificación de 0 (cero) y el caso será reportado al Docente quien a su vez informará a la Escuela de Ciencias y Sistemas para su seguimiento institucional.

Prórrogas y reposiciones

- No se otorgarán prórrogas para entregas de actividades.
- No se permitirá la reposición de proyectos bajo ninguna circunstancia.

Requisitos para evaluación final del curso

- Es obligatorio aprobar el laboratorio para tener derecho a la evaluación final del curso.
- La calificación de prácticas, proyectos y otras actividades que se indique será asignada de forma presencial, en la fecha y hora establecidas por el tutor académico.

Asistencia

- Para obtener la nota del laboratorio, se requiere un mínimo del 80% de asistencia a las sesiones de laboratorio.
- En caso de inasistencia, sólo se aceptarán justificaciones válidas respaldadas por

constancia oficial.

Entregas

- No se aceptarán entregas tardías de tareas, prácticas, exámenes cortos, exámenes finales o proyectos sin justificación.

Medio oficial de entrega

- La plataforma UEDI de la Facultad será el único medio oficial para la entrega de actividades del curso.
-

Equipo Académico

Coordinador del Área

Nombre: Carlos Alonzo	Correo electrónico:

Sección B

Docente

Nombre del Docente	Correo electrónico
Ing. Zulma Karina Aguirre Ordonez	zaguirre@ingenieria.usac.edu.gt

	Lunes	Martes	Miércoles	Jueves	Viernes	Sábado
Día		X				
Horario		7:10 – 8:50				
Lugar		Meet				

Tutor(es)

Nombre del Tutor	**River Anderson Ismalej Roman**	
Correo electrónico institucional	**3169233891503@ingenieria.usac.edu.gt**	

Tipo		Lunes	Martes	Miércoles	Jueves	Viernes	Sábado
Clase	Día		X				
	Horario		8:50 – 10:30				
	Lugar		Meet				
Atención al Estudiante	Día		X				
	Horario		8:50 – 10:30				
	Lugar		Meet				

Bibliografía

Cococys

E-Grafía

Cococys