ORGANIZACIÓN DE LENGUAJES Y COMPILADORES 2 - SECCIÓN B

CÓDIGO:	**0781**	PUNTEO NETO LABORATORIO:	**32 PUNTOS**
ESCUELA DE INGENIERÍA EN:	**CIENCIAS Y SISTEMAS**	ÁREA A LA QUE PERTENECE:	**CIENCIAS DE LA COMPUTACIÓN**
PRE REQUISITO:	**0772 – ESTRUCTURAS DE DATOS** **0777 – ORGANIZACIÓN DE LENGUAJES Y COMPILADORES 1**	POST REQUISITO:	**0281 – SISTEMAS OPERATIVOS 2** **0972 – INTELIGENICA ARTIFICIAL 1**
CATEGORÍA:	**OBLIGATORIO**	VIGENCIA:	**SEGUNDO SEMESTRE 2026**

Descripción del Laboratorio

El laboratorio de Organización de Lenguajes y Compiladores 2, adscrito al área de Ciencias de la Computación, profundiza en las fases de análisis semántico y síntesis de un compilador, complementando la base teórica de la cátedra con la construcción práctica de un intérprete y un compilador. A lo largo del curso se desarrollan dos proyectos principales: un intérprete que realiza análisis semántico durante la ejecución mediante herramientas generadoras de parser, y un compilador que traduce a lenguajes ensambladores como RISC-V o AArch64. El contenido se organiza en tres unidades. La primera cubre análisis sintáctico (descendente, ascendente y resolución de ambigüedades) y análisis semántico, abordando atributos sintetizados y heredados, alcance y tabla de símbolos, validación de tipos con tipado estático y dinámico, y recuperación de errores. La segunda unidad extiende estos conceptos a la representación semántica de funciones y objetos, incluyendo parámetros y argumentos, hoisting, callstack y closures, así como los fundamentos de programación orientada a objetos: clases, prototipos, constructores, sobrecarga de métodos y su estructura en memoria. La tercera unidad se enfoca en representaciones intermedias y generación de código (código de tres direcciones, bytecode y ensamblador para tipos como enteros, punto flotante, caracteres y strings), así como en los entornos de ejecución: gestión de memoria en stack y heap, descriptores de registros y direcciones, registros de activación, paso de argumentos y resolución de variables. El curso cierra con la optimización de la generación de código, incluyendo la división en bloques básicos y algoritmos de optimización sobre código de tres direcciones. Estas competencias trascienden el campo específico de los compiladores y resultan aplicables en áreas como la ingeniería de software, el diseño de sistemas y el desarrollo de lenguajes de dominio específico.

Resumen de Ponderaciones y Tiempo de Auto-Aprendizaje

TIPO	CANTIDAD	PONDERACIÓN	HORAS DE AUTO-APRENDIZAJE
Proyectos	2	90	75
Tareas	4	10	7
TOTAL	6	**100**	**82**

Desglose de Actividades

TIPO DE ACTIVIDAD	PONDERACIÓN	HORAS DE AUTO-APRENDIZAJE
Proyecto 1	40	35
Proyecto 2	50	40
Tareas (4)	4 (2.5)	7
TOTAL	**100**	**82**

Equipo Académico

Coordinador del Área

Nombre: **M.Sc. Luis Fernando Espino Barrios**	Correo electrónico: **usac.sistemas@gmail.com**

Docente

Nombre: **Ing. Edgar Sabán**	Correo electrónico del Docente: **1766572310101@ingenieria.usac.edu.gt**

	Lunes	Martes	Miércoles	Jueves	Viernes	Sábado
Día		X		x		
Horario		19:00 – 20:40		19:00 – 20:40		
Lugar		MEET		MEET		

Tutor(es)

Nombre del Tutor	**Omar Alejandro Vides Esteban**	
Correo electrónico institucional	**3005502780101@ingenieria.usac.edu.gt**	

Tipo		Lunes	Martes	Miércoles	Jueves	Viernes	Sábado
Clase	Día		X				
	Horario		17:20 - 19:00				
	Lugar		MEET				
Atención al Estudiante	Día	x	x	x	x	x	x
	Horario	09:0 -23:00	09:0 -23:00	09:0 -23:00	09:0 -23:00	09:0 -23:00	09:0 -23:00
	Lugar	UEDI	UEDI	UEDI	UEDI	UEDI	UEDI

Índice

Competencias Vinculadas al Perfil del Egresado

Competencias Específicas

No.	Competencia
1	Demuestra pensamiento crítico, actitud investigativa y rigor analítico en el planteamiento y la resolución de problemas complejos.
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	Aplica estándares de calidad, eficiencia y seguridad en la implementación adecuada de soluciones de software, hardware y TIC en general.
3	Actualiza permanente sus conocimientos relacionados con TIC en general, apoyándose en las estrategias de aprendizaje apropiadas.

Competencias del Laboratorio

Competencia(s) Específica(s)

No.	Competencia	Nivel de Aprendizaje
1	Analiza estructuras de lenguaje mediante la aplicación de reglas formales para la generación de gramáticas.	Analizar
2	Analiza la semántica de un lenguaje de programación mediante la validación de tipos, alcance y estructuras de control para determinar la corrección de expresiones y sentencias.	Analizar
3	Evalúa la representación semántica de funciones y objetos mediante el análisis de alcance, tipos y tabla de símbolos para determinar su correcta representación en intérpretes y compiladores.	Evaluar

4	Analiza las equivalencias sintácticas y semánticas de un lenguaje de alto nivel mediante el mapeo de construcciones a instrucciones de Representación Intermedia tales como C3D, cuádruplas o Bytecode.	Comprender
5	Generar un intérprete utilizando herramientas generadoras de parser en un lenguaje dado realizando el análisis semántico durante la ejecución.	Aplicar
6	Generar un compilador utilizando como representación intermedia lenguajes ensambladores como RISC-V o Aarch64.	Crear

Competencia(s) General(es)

No.	Competencia	Nivel de Aprendizaje
1	Identifica motivaciones y estructura esencial de un compilador mediante un repaso práctico de análisis sintáctico para establecer las bases teóricas del curso.	Comprender

Diseño Didáctico

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	Cuestionario de diagnóstico de conocimientos previos del 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
Cuestionario	Diagnóstico para saber qué aspectos reforzar durante el curso.

Sesión No. 2, Unidad No. 1 - Análisis Sintáctico

Área Actitudinal (Saber ser)

Nombre del valor: Integridad y Veracidad Técnica
La integridad en la ingeniería de sistemas implica el compromiso con la precisión técnica y el seguimiento riguroso de estándares, garantizando que el software se comporte de manera predecible. La veracidad asegura que los reportes del sistema (como el manejo de errores) sean reflejos exactos y transparentes de la lógica procesada.

Área de Conocimiento (Saber)

Competencia(s)	
Analiza estructuras de lenguaje mediante la aplicación de reglas formales para la generación de gramáticas.	
Tema	Subtema
Análisis Sintáctico	Análisis Descendente
Análisis Sintáctico	Análisis Ascendente
Análisis Sintáctico	Resolución de ambigüedades sintácticas
Análisis Sintáctico	Esquemas de Traducción

Área de Habilidades (Saber Hacer)

Competencia	
Analiza estructuras de lenguaje mediante la aplicación de reglas formales para la generación de gramáticas.	
Tipo de Actividad	Ponderación

Ejercicio práctico	-

Sesión No. 3, Unidad No. 1 – Análisis Semántico

Área Actitudinal (Saber ser)

Nombre del valor: Rigor y Precisión Técnica
Es el compromiso con la exactitud y el cumplimiento estricto de las especificaciones lógicas en la construcción de sistemas. Se traduce en una atención meticulosa a los detalles técnicos para evitar errores de interpretación que comprometan la funcionalidad.

Área de Conocimiento (Saber)

Competencia(s)	
Analiza la semántica de un lenguaje de programación mediante la validación de tipos, alcance y estructuras de control para determinar la corrección de expresiones y sentencias.	
Tema	Subtema
Análisis Semántico	Atributos Sintetizados y Heredados
Análisis Semántico	Alcance y Tabla de Símbolos
Análisis Semántico	Validación de Tipos
Análisis Semántico	Entornos

Área de Habilidades (Saber Hacer)

Competencia	
Analiza la semántica de un lenguaje de programación mediante la validación de tipos, alcance y estructuras de control para determinar la corrección de expresiones y sentencias.	
Tipo de Actividad	Ponderación
Ejercicio práctico	-

Sesión No. 4, Unidad No. 1 – Análisis Semántico

Área Actitudinal (Saber ser)

Nombre del valor: Ética en la Gestión de Errores y Robustez
Se refiere a la honestidad y compromiso del desarrollador por gestionar los fallos de manera que el sistema sea resiliente. Implica diseñar software que priorice la seguridad del usuario ante entradas inesperadas, garantizando una operación estable, transparente y confiable.

Área de Conocimiento (Saber)

Competencia(s)	
Analiza la semántica de un lenguaje de programación mediante la validación de tipos, alcance y estructuras de control para determinar la corrección de expresiones y sentencias.	
Tema	Subtema
Análisis Semántico	Tipado Dinámico
Análisis Semántico	Expresiones y Sentencias
Análisis Semántico	Estructuras de Control
Análisis Semántico	Recuperación de Errores

Área de Habilidades (Saber Hacer)

Competencia	
Analiza la semántica de un lenguaje de programación mediante la validación de tipos, alcance y estructuras de control para determinar la corrección de expresiones y sentencias.	
Tipo de Actividad	Ponderación
Ejercicio práctico	-

Sesión No. 5, Unidad No. 2 - Funciones

Área Actitudinal (Saber ser)

Nombre del valor: Ética del Alcance
Mantener el uso de variables dentro de su contexto lógico y justo.

Área de Conocimiento (Saber)

Competencia(s)	
Evalúa la representación semántica de funciones, y objetos mediante el análisis de alcance, tipos y tabla de símbolos para determinar su correcta representación en intérpretes y compiladores.	
Analiza la semántica de un lenguaje de programación mediante la validación de tipos, alcance y estructuras de control para determinar la corrección de expresiones y sentencias.	
Tema	Subtema
Funciones	Funciones y sus tipos
Funciones	Parámetros y Argumentos de una función
Funciones	Hoisting
Funciones	Callstack
Funciones	Closures

Área de Habilidades (Saber Hacer)

Competencia	
Evalúa la representación semántica de funciones, y objetos mediante el análisis de alcance, tipos y tabla de símbolos para determinar su correcta representación en intérpretes y compiladores.	
Tipo de Actividad	Ponderación
Ejercicio práctico	-

Sesión No. 6, Unidad No. 2 - Programación Orientada a Objetos

Área Actitudinal (Saber ser)

Nombre del valor: Precisión
Ya que al aplicar técnicas de optimización y análisis de código, se requiere atención a los detalles para lograr mejoras reales y evitar errores.

Área de Conocimiento (Saber)

Competencia(s)	
Evalúa la representación semántica de funciones y objetos mediante el análisis de alcance, tipos y tabla de símbolos para determinar su correcta representación en intérpretes y compiladores.	
Analiza la semántica de un lenguaje de programación mediante la validación de tipos, alcance y estructuras de control para determinar la corrección de expresiones y sentencias.	
Tema	Subtema
Programación Orientada a Objetos	Clases y Objetos
Programación Orientada a Objetos	Prototipos
Programación Orientada a Objetos	Constructores
Programación Orientada a Objetos	Sobrecarga de Métodos
Programación Orientada a Objetos	Estructura en Memoria

Área de Habilidades (Saber Hacer)

Competencia	
Evalúa la representación semántica de funciones, y objetos mediante el análisis de alcance, tipos y tabla de símbolos para determinar su correcta representación en intérpretes y compiladores	
Tipo de Actividad	Ponderación
Otros	-

Sesión No. 7, Unidad No. 3 - Representaciones intermedias y generación de código

Área Actitudinal (Saber ser)

Nombre del valor: Rigor
porque se trabaja con conceptos técnicos de bajo nivel, donde la precisión y estructura lógica son clave, especialmente en temas como ensamblador o representación en memoria.

Área de Conocimiento (Saber)

Competencia(s)	
Analiza las equivalencias sintácticas y semánticas de un lenguaje de alto nivel mediante el mapeo de construcciones a instrucciones de Representación Intermedia tales como C3D, cuádruplas o Bytecode.	
Tema	Subtema
Representaciones intermedias y generación de código	Codigo de 3 direcciones
Representaciones intermedias y generación de código	Bytecode
Representaciones intermedias y generación de código	Assembly: Risc-V y ARM64
Representaciones intermedias y generación de código	Enteros
Representaciones intermedias y generación de código	Punto Flotante
Representaciones intermedias y generación de código	Char y Strings

Área de Habilidades (Saber Hacer)

Competencia	
Analiza las equivalencias sintácticas y semánticas de un lenguaje de alto nivel mediante el mapeo de construcciones a instrucciones de Representación Intermedia tales como C3D, cuádruplas o Bytecode.	
Tipo de Actividad	Ponderación
Ejercicio práctico	-

Sesión No. 8, Unidad No. 3 - Entornos y Memoria en un Compilador

Área Actitudinal (Saber ser)

Nombre del valor: Responsabilidad computacional
El uso correcto de las tablas de símbolos ayuda a la eficiencia del programa.

Área de Conocimiento (Saber)

Competencia(s)	
Analiza las equivalencias sintácticas y semánticas de un lenguaje de alto nivel mediante el mapeo de construcciones a instrucciones de Representación Intermedia tales como C3D, cuádruplas o Bytecode.	
Tema	Subtema
Entornos y memoria en un compilador	Tabla de Símbolos
Entornos y memoria en un compilador	Stack, Heap y punteros
Entornos y memoria en un compilador	Resolución de variables
Entornos y memoria en un compilador	Estructuras de Control en C3D (Etiquetas y Saltos)

Área de Habilidades (Saber Hacer)

Competencia	
Analiza las equivalencias sintácticas y semánticas de un lenguaje de alto nivel mediante el mapeo de construcciones a instrucciones de Representación Intermedia tales como C3D, cuádruplas o Bytecode.	
Tipo de Actividad	Ponderación
Otros	-

Sesión No. 9, Unidad No. 3 - Entornos y Memoria en un Compilador

Área Actitudinal (Saber ser)

Nombre del valor: Rigor técnico
La traducción de funciones e instrucciones ocupa la mayor parte de complejidad en un compilador por lo que mejora su uso.

Área de Conocimiento (Saber)

Competencia(s)	
Generar un compilador utilizando como representación intermedia lenguajes ensambladores como RISC-V o Aarch64.	
Tema	Subtema
Entornos y memoria en un compilador	Memoria en tiempo de ejecución
Entornos y memoria en un compilador	Descriptores de registros y direcciones
Entornos y memoria en un compilador	Registros de activación
Entornos y memoria en un compilador	Paso de argumentos

Área de Habilidades (Saber Hacer)

Competencia	
Generar un compilador utilizando como representación intermedia lenguajes ensambladores como RISC-V o Aarch64.	
Tipo de Actividad	Ponderación
Ejercicio práctico	-

Sesión No. 10, Unidad No. 3 - Representaciones Intermedias y Generación de Código

Área Actitudinal (Saber ser)

Nombre del valor: Responsabilidad computacional
El uso de la memoria permite que los programas generados por el compilador tengan un rendimiento adecuado.

Área de Conocimiento (Saber)

Competencia(s)	
Generar un compilador utilizando como representación intermedia lenguajes ensambladores como RISC-V o Aarch64.	
Tema	Subtema
Entornos y memoria en un compilador	Memoria heap en tiempo de ejecución
Representaciones intermedias y generación de código	Arreglos y String, métodos de guardado
Representaciones intermedias y generación de código	Utilización de la pila para guardar punteros

Área de Habilidades (Saber Hacer)

Competencia	
Generar un compilador utilizando como representación intermedia lenguajes ensambladores como RISC-V o Aarch64.	
Tipo de Actividad	Ponderación
Ejercicio práctico	-

Sesión No. 11, Unidad No. 3 - Optimización de la Generación de Código

Área Actitudinal (Saber ser)

Nombre del valor: Organización y estructura
El manejo de entornos y tabla de símbolos requiere una organización sistemática para gestionar correctamente el alcance de variables y sus valores.

Área de Conocimiento (Saber)

Competencia(s)	
Generar un compilador utilizando como representación intermedia lenguajes ensambladores como RISC-V o Aarch64.	
Tema	Subtema
Entornos y memoria en un compilador	Utilización del puntero link para funciones recursivas
Optimización de la generación de código	Métodos de optimización
Optimización de la generación de código	Fases para optimizar en la creación del compilador

Área de Habilidades (Saber Hacer)

Competencia	
Generar un compilador utilizando como representación intermedia lenguajes ensambladores como RISC-V o Aarch64.	
Tipo de Actividad	Ponderación
Ejercicio práctico	-

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.

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.

Bibliografía

- Libro de Texto: Compiladores. Principios, Técnicas y Herramientas Aho, Sethi y Ullmam. PEARSON ADDISON-WESLEY, 2008, segunda edición.

E-Grafía

- Carter, Z. (2009). Jison – Documentación. Disponible en: https://zaa.ch/jison/docs/
- Klein, G., Rowe, S. y Décamps, R. (1998). JFlex – Manual. Disponible en: https://jflex.de/manual.html
- Viswanadha, S. y Sankar, S. (s. f.). JavaCC – Documentación. Disponible en: https://javacc.github.io/javacc
- CUP – Documentación. Disponible en: http://www2.cs.tum.edu/projects/cup/docs.php
- PLY – Documentación. Disponible en: https://www.dabeaz.com/ply/
- Python – Ply
    - https://ericknavarro.io/2020/02/10/24-Mi-primer-proyecto-utilizando-PLY/
    - https://ericknavarro.io/2020/03/15/26-Interprete-sencillo-utilizando-PLY/
    - http://kunusoft.com/slides/compi2/manual_ply_python/
    - http://kunusoft.com/slides/compi2/guia_heredados_pila/index.php?pic=0 o
- Java – JavaCC
    - https://ericknavarro.io/2020/02/10/23-Mi-primer-proyecto-utilizando-JavaCC/
    - https://ericknavarro.io/2020/03/07/25-Interprete-sencillo-utilizando-Javacc/ o Java-Flex-y-Cup
    - https://ericknavarro.io/2019/04/26/02-Mi-primer-proyecto-utilizando-Jlex-y-Cup-Windows/
    - https://ericknavarro.io/2019/04/26/05-Interprete-sencillo-utilizando-Java-Jlex-y-Cup/Javascript/ NodeJS – Jison
    - https://ericknavarro.io/2019/07/21/17-Mi-primer-proyecto-utilizando-Jison
    - https://ericknavarro.io/2019/08/01/20-Interprete-sencillo-utilizando-Jison-con-Nodejs-Ubuntu/
- C# – Irony
    - https://ericknavarro.io/2019/07/24/19-Mi-primer-proyecto-utilizando-Irony-Windows/
    - https://ericknavarro.io/2019/08/07/22-Interprete-sencillo-utilizando-Irony-con-CS/
- Visual Basic – Gold Parser
    - https://ericknavarro.io/2019/07/22/18-Mi-Primer-Proyecto-Utilizando-GOLD-Parser-Windows/
    - https://ericknavarro.io/2019/08/07/21-Interprete-sencillo-utilizando-GOLD-Parser-y-Visual-Basic