RESOLUCIÓN 25/18/113-00
ACTA 1226/25/08/2025

"POR LA CUAL SE APRUEBA EL PROGRAMA DE ESTUDIO DE LA ASIGNATURA *DISEÑO DE COMPILADORES,* DE CARRERAS DE GRADO, SEDE SAN LORENZO, FILIAL VILLARRICA Y FILIAL CORONEL OVIEDO"

VISTO: El Memorando DA/1707/2025 del Director Académico de la FP-UNA, Prof. MSc. Felipe Santiago Uzabal Escurra, con el cual remite el Memorando CCPTCC/030/2025 de la Comisión Coordinadora del Proyecto de Transformación Curricular de Carreras de Grado de la FP-UNA, en el que presenta la propuesta de Programas de Estudio de la Asignatura de las Carreras de Grado.

CONSIDERANDO: La Ley 4995/2013 de Educación Superior, el Estatuto de la Universidad Nacional de Asunción y las deliberaciones sobre el tema.

Que la Comisión Coordinadora del Proyecto de Transformación Curricular de Carreras de Grado, solicita la aprobación del Programa de Estudio de la asignatura ***"Diseño de Compiladores"***, de la carrera Ingeniería en Informática – Plan 2026.

Que los programas fueron elaborados conforme a las disposiciones establecidas por el Consejo Nacional de Educación Superior (CONES) en materia de **créditos académicos**, según lo dispuesto en la Resolución CONES N.° 221/2024, que regula el *Sistema de Créditos Académicos – Paraguay* y los criterios para su publicación en las carreras de grado.

EL CONSEJO DIRECTIVO DE LA FACULTAD POLITÉCNICA
RESUELVE:

25/18/113-01 APROBAR el Programa de Estudio de la Asignatura ***"Diseño de Compiladores"*** de las carreras de Grado, Sede San Lorenzo, Filial Villarrica y Filial Coronel Oviedo detallado en el ANEXO 75 de la presente Acta.

25/18/113-02 COMUNICAR, copiar y archivar.

Prof. Abg. Joel Arsenio Benítez Santacruz
Secretario

Prof. Ing. Silvia Teresa Leiva León, MSc.
Presidenta

TEL/FAX: +595 21 5887000 – C.C. 1130 (Asunción) – 2111 (San Lorenzo)
www.pol.una.py

DEPARTAMENTO DE ENSEÑANZA DE INFORMÁTICA
PROGRAMA DE ESTUDIO

I. IDENTIFICACIÓN

Nivel	Grado				
Asignatura	Diseño de Compiladores				
Carrera	**Plan**	**Sede/Filial**	**Carácter**	**Semestre**	**Prerrequisitos**
Licenciatura en Ciencias Informáticas	2026	Sede San Lorenzo/Filial Villarrica/Filial Coronel Oviedo	Electiva	***	Ingeniería de Software I, Base de Datos II, Seguridad Informática, Programación Frontend
Ingeniería en Informática	2026	Sede San Lorenzo	Obligatoria	Séptimo	Estructura de Lenguajes de Programación.

Semanal					**Periodo**				
HT	**HP**	**HTD**	**HTI**	**HS**	**PA**	**THTD**	**THTI**	**THA**	**CA-PY**
2	2	4	4	8	18	72	72	144	5

***HT: Horas Teóricas semanales.**
***HP: Horas Prácticas semanales.**
***HTD: Horas semanales de Trabajo académico con acompañamiento Docente.**
***HTI: Horas semanales de Trabajo académico Independiente del estudiante.**
***HS: Horas Semanales (HTD+HTI).**
***PA: Periodo Académico en semanas.**
*** THTD: Total de Horas de Trabajo académico con acompañamiento Docente (HTD*PA).**
*** THTI: Total de Horas de Trabajo académico Independiente del estudiante (HTI*PA).**
*** THA: Total de Horas de trabajo Académico (THTD+THTI).**
*** CA-PY: Créditos académicos de la asignatura**

II. FUNDAMENTACIÓN

Los compiladores o procesadores de lenguajes son esenciales para el profesional de la informática ya que constituyen el nexo entre el hardware y el software de alto nivel. Los procesadores de lenguajes permiten traducir, interpretar y ejecutar instrucciones escritas en lenguajes de programación comprensibles por los humanos, generalmente cercanos al lenguaje natural, hacia instrucciones de máquina o lenguajes de bajo nivel que pueden ser directamente ejecutados por el hardware. Esta capacidad es fundamental para el desarrollo de software, ya que permite a los programadores abstraerse de la complejidad del hardware y enfocarse en la lógica de las aplicaciones, mejorando así la productividad, la portabilidad del código y la mantenibilidad del software.

Una vez que el profesional informático entienda los principios y técnicas involucradas en este proceso, puede aplicarlas para resolver problemas más generales, ya que cuenta con la capacidad de definir los problemas como si fuera un lenguaje de entrada, de manera simple. De esta manera, podrá implementar una solución de forma eficiente, elegante y simple ya que podrá generar en el código fuente del programa en forma automática. En la actualidad podrá sumar herramientas de IA para complementar este proceso, pudiendo converger los beneficios de los diferentes modelos estudiados cuando así lo requiera.

Los compiladores, analizan el código fuente y lo optimizan, generando un programa ejecutable eficiente en términos de uso de recursos y rendimiento. El conocimiento de cómo los compiladores realizan esta traducción y optimización es crucial para el profesional de la informática, ya que le permite escribir código que aproveche de manera óptima las características del lenguaje y la arquitectura del hardware. Además, comprender los procesos de compilación ayuda a diagnosticar errores de programación, optimizar el rendimiento y asegurar la calidad y seguridad del software producido. Los procesadores de lenguajes y los compiladores no solo son herramientas fundamentales para la creación de programas, sino que también son claves en la educación de un profesional informático para comprender y aplicar los principios de la ciencia de la computación en la práctica y la solución de problemas reales.

De naturaleza teórico-práctica, la asignatura combina conceptos teóricos fundamentales como gramáticas y análisis semántico con aplicaciones prácticas que incluyen la construcción de analizadores y traductores. Además, fomenta competencias críticas como el razonamiento lógico, la resolución de problemas y la implementación de soluciones innovadoras en diversos contextos tecnológicos.

La inclusión de esta asignatura en la malla curricular asegura que los estudiantes comprendan las bases científicas del procesamiento de lenguajes, mientras desarrollan habilidades prácticas para diseñar sistemas que sean eficientes, escalables y relevantes para las necesidades actuales y futuras del campo de la informática.

La relación entre los ejes temáticos y las unidades es la siguiente:

- Análisis léxico: se desarrolla en las secciones 2.6, 3.1 a 3.7, cubriendo desde el reconocimiento de token y expresiones regulares hasta el diseño de analizadores léxicos.
- Análisis sintáctico: se aborda en las secciones 4.1 a 4.6, incluyendo gramáticas, análisis descendente y ascendente, y la relación con el análisis léxico.
- Análisis semántico: se trata en las secciones 5.3 a 5.6, enfocándose en tablas de símbolos, esquemas de traducción, y la verificación de reglas semánticas.
- Generación de código intermedio: se desarrolla en las secciones 6.1 a 6.3, abarcando lenguajes intermedios y la representación de declaraciones y proposiciones.
- Optimización de código: Incluye elementos de las secciones 5.6, 7.1 y 7.2, con enfoque en la simplificación y mejora del código durante el procesamiento.
- Generación de código de máquina: se relaciona con las secciones 6.1 a 6.3, en las cuales se introduce la conversión a lenguajes intermedios como paso previo a la generación de código final.
- Manejo de errores: se aborda en las secciones 4.6 y 5.6, explicando cómo detectar y corregir errores en distintas etapas del compilador.
- Análisis y optimización de tiempo de ejecución: se menciona en la sección 7.2, con aplicación de técnicas para mejorar el rendimiento de los programas procesados.
- Herramientas para la construcción de compiladores: se desarrolla en las secciones 4.7 y 7.2, explorando herramientas modernas, incluidas las basadas en inteligencia artificial.

III. COMPETENCIAS DEL PERFIL DE EGRESO ASOCIADAS

1. Liderar y trabajar en equipo con eficacia y responsabilidad tomando decisiones basadas en evidencias.
2. Aplicar en la práctica profesional los valores humanos, la ética y los mecanismos de seguridad laboral.
3. Actuar proactivamente frente a los problemas sociales y ambientales.
4. Adaptarse respetuosamente a contextos nuevos o adversos, así como a diversidades personales, disciplinares y culturales.
5. Evaluar el comportamiento de diversos fenómenos disciplinares e interdisciplinares relacionados con la informática con una visión de sistema, mediante modelos teóricos

validados y actualizados, capaces de abarcarlos integralmente en un contexto de incertidumbre.

6. Planificar, proyectar, diseñar y ejecutar proyectos sostenibles e integrales para la resolución de problemas, la mejora y la innovación en el ámbito de la informática.

IV. ORGANIZACIÓN DE LA ASIGNATURA

Unidades	Contenidos	Resultados de aprendizaje
1. Introducción.	1.1 Procesadores de lenguaje. 1.2 La estructura de un compilador. 1.3 La evolución de los lenguajes de programación. 1.4 La ciencia de construir un compilador. 1.5 Aplicaciones de los compiladores. 1.6 Compiladores, traductores e intérpretes.	1. Identifica los diferentes procesadores de lenguaje y sus roles en el contexto del desarrollo y ejecución de programas. 2. Describe la estructura de un compilador, incluyendo sus fases principales y su funcionamiento. 3. Analiza la evolución de los lenguajes de programación y su impacto en el diseño y la implementación de compiladores. 4. Explica los principios fundamentales de la ciencia de construcción de un compilador, incluyendo conceptos teóricos y técnicos. 5. Reconoce las diversas aplicaciones de los compiladores en el desarrollo de software y otros campos tecnológicos. 6. Diferencia entre compiladores, traductores e intérpretes, explicando sus características, similitudes y diferencias.
2. Traductor simple dirigido por la sintaxis.	2.1 Introducción. 2.2 Definición de sintaxis. 2.3 Traducción orientada a la sintaxis. 2.4 Análisis sintáctico. 2.5 Un traductor para expresiones simples: 2.5.1 Sintaxis abstracta y concreta. 2.5.2 Adaptación del esquema de traducción. 2.5.3 Procedimiento para los no terminales. 2.5.4 Simplificación del	1. Introduce los conceptos básicos sobre la sintaxis y su importancia en el procesamiento de lenguajes. 2. Define el concepto de sintaxis y explica su rol en la traducción de lenguajes. 3. Aplica principios de traducción orientada a la sintaxis para resolver problemas específicos en procesamiento de lenguajes. 4. Analiza los procesos involucrados en el análisis sintáctico y su relación con la estructura de los lenguajes de programación. 5. Construye un traductor para expresiones simples, utilizando conceptos como sintaxis abstracta y concreta, adaptación de

Unidades	Contenidos	Resultados de aprendizaje
	traductor. 2.5.5 El programa completo. 2.6 Análisis léxico. 2.6.1 Eliminación de espacio en blanco y comentarios. 2.6.2 Lectura adelantada. 2.7 Tabla de símbolos. 2.7.1 Tabla de símbolos por alcance. 2.7.2 El uso de las tablas de símbolos. 2.8 Herramientas de IA para complementar el procesamiento de lenguajes. 2.8.1 ChatGPT, TDS y escritura de códigos. 2.8.2 Alcance y limitaciones.	esquemas de traducción, y simplificación del procedimiento. 6. Implementa técnicas de análisis léxico, incluyendo la eliminación de espacios en blanco y comentarios, y el uso de lectura adelantada para optimizar el procesamiento. 7. Gestiona una tabla de símbolos, distinguiendo entre tablas por alcance y el uso de estas para manejar información de los lenguajes procesados. 8. Evalúa el uso de herramientas de inteligencia artificial como ChatGPT y TDS en la escritura de códigos, identificando sus alcances y limitaciones en el procesamiento de lenguajes.
3. Análisis léxico.	3.1 La función del analizador léxico. 3.2 Especificaciones para tokens, lexemas y patrones. 3.3 Expresiones y definiciones regulares. 3.4 Reconocimiento de tokens. 3.5 Autómatas finitos. 3.6 De las expresiones regulares a los autómatas. 3.7 Diseño de un generador de analizadores léxicos. 3.8 Minimización de estados de un AFD.	1. Explica la función del analizador léxico y su rol en el procesamiento de lenguajes. 2. Especifica las características de token, lexemas y patrones, y su relevancia en el análisis léxico. 3. Aplica expresiones regulares y definiciones para describir patrones léxicos de un lenguaje de programación. 4. Reconoce la relación entre token y sus patrones correspondientes mediante técnicas de análisis. 5. Describe el concepto y las propiedades de los autómatas finitos, incluyendo su funcionamiento básico. 6. Transforma expresiones regulares en autómatas finitos, aplicando los métodos de construcción correspondientes. 7. Diseña un generador de analizadores léxicos, integrando conceptos como reconocimiento de token y autómatas. 8. Optimiza el desempeño de un

Unidades	Contenidos	Resultados de aprendizaje
		autómata finito determinista (AFD) mediante técnicas de minimización de estados.
4. Análisis sintáctico.	3.2 Introducción al análisis sintáctico. Gramáticas libres de contexto. 3.3 Escritura de una gramática. 3.4 Análisis sintáctico descendente y gramáticas LL(1). 3.5 Análisis sintáctico ascendente y gramáticas LR(1). 3.6 Técnicas de conversión entre el analizador sintáctico y léxico para la solución de problemas. 3.7 Uso de las herramientas de IA en procesadores de lenguajes genéricos.	1. Introduce los conceptos básicos del análisis sintáctico y su rol en el procesamiento de lenguajes. 2. Explica las características y aplicaciones de las gramáticas libres de contexto en el diseño de lenguajes. 3. Escribe gramáticas libres de contexto, aplicando reglas formales para describir estructuras de lenguaje. 4. Analiza el funcionamiento del análisis sintáctico descendente, incluyendo las gramáticas LL(1) y sus propiedades. 5. Describe el proceso del análisis sintáctico ascendente, destacando el uso de gramáticas LR(1) en su implementación. 6. Aplica técnicas de conversión entre el analizador sintáctico y léxico para resolver problemas de procesamiento de lenguajes. 7. Utiliza herramientas de inteligencia artificial en el diseño e implementación de procesadores de lenguajes genéricos, evaluando su impacto y efectividad.
5. Traducción orientada por la sintaxis y análisis semántico.	5.1 Definiciones dirigidas por la sintaxis. 5.2 Órdenes de evaluación para las definiciones dirigidas por la sintaxis. 5.3 Aplicaciones de la traducción dirigida por la sintaxis y el analizador semántico. 5.4 Esquemas de traducción. 5.5 Eliminación de recursión por la izquierda con variables heredadas. 5.6 Análisis semántico, tablas de símbolos y hashing.	1. Define las definiciones dirigidas por la sintaxis y su rol en el procesamiento de lenguajes. 2. Establece los órdenes de evaluación para implementar las definiciones dirigidas por la sintaxis en procesos de traducción. 3. Aplica los principios de la traducción dirigida por la sintaxis en el diseño de analizadores semánticos, destacando sus aplicaciones prácticas. 4. Desarrolla esquemas de traducción basados en la sintaxis

Unidades	Contenidos	Resultados de aprendizaje
		para resolver problemas específicos de procesamiento de lenguajes. 5. Optimiza procesos mediante la eliminación de recursión por la izquierda utilizando variables heredadas en las gramáticas. 6. Implementa técnicas de análisis semántico, incluyendo el uso eficiente de tablas de símbolos y métodos de hashing para la gestión de datos semánticos.
6. Generación de código intermedio.	6.1 Lenguajes intermedios. 6.2 Declaraciones. 6.3 Proposición de asignación.	1. Explica el concepto de lenguajes intermedios y su importancia en la etapa de traducción de compiladores. 2. Describe las diferentes declaraciones en lenguajes de programación y su representación en lenguajes intermedios. 3. Analiza el proceso de traducción de proposiciones de asignación en lenguajes de alto nivel hacia lenguajes intermedios.
7. Unificación de las técnicas.	7.1. Alcance, impacto y beneficios de las técnicas para el procesamiento de lenguajes. 7.2. Definición de problemas y soluciones aplicando técnicas de análisis léxico y sintáctico.	1. Evalúa el alcance, impacto y beneficios de las técnicas para el procesamiento de lenguajes en diversos contextos tecnológicos. 2. Define problemas específicos y propone soluciones utilizando técnicas de análisis léxico y sintáctico, aplicándolas de manera efectiva.

V. ESTRATEGIAS DIDÁCTICAS

En el desarrollo del programa se aplicarán algunas estrategias didácticas apropiadas con el objetivo de promover el acceso al conocimiento y fijación de los conceptos desarrollados. asegurar la aprehensión de los conocimientos desarrollados y el dominio de estos para la resolución de problemas. Además, el entendimiento de la teoría combinada con la ejecución práctica de procesos y procedimientos permiten encarar problemas nuevos. Por esta razón se plantean las siguientes estrategias didácticas:

- **Aprendizaje basado en problemas:** planteamiento de problemas con enfoques que requieran la aplicación y/o combinación de los conceptos aprendidos. Trabajo realizado y presentado por los alumnos en clase con posibilidad de generar participación en debates, preguntas e interacción con toda la clase.

- **Debate:** exposición por parte del docente de los conceptos básicos por unidad, con materiales de lectura y ejemplos orientados a la enseñanza de las competencias específicas de la asignatura. El docente asume el rol de expositor y moderador a la vez, buscando generar el debate a través de preguntas sobre lo expuesto y desde la participación de los estudiantes.

- **Clase invertida:** con materiales didácticos dispuestos en el aula virtual previamente y presentado en clases presenciales estudios de caso, los estudiantes toman el control del desarrollo de conceptos relacionados con el contenido. Esta estrategia didáctica se ve fortalecida cuando el proceso de inversión se vuelve dinámico y cambia durante la clase, dejando al docente en un segundo plano.

- **Aprendizaje colaborativo.** Esta estrategia es especialmente aplicada en los trabajos prácticos y otras tareas donde los estudiantes pueden interactuar entre ellos y en forma colaborativa enfrentar los desafíos que se proponen.

- **Gamificación:** se plantean problemas en un entorno de juego para que el estudiante pueda con su participación comprender mejor el impacto de algunos conceptos en ambientes cercanos a la realidad. Esto especialmente se da en la parte final de la asignatura, cuando el conjunto de las técnicas y principios desarrollados le permiten enfrentar situaciones de realidad simulada.

La elección particular de la estrategia didáctica aplicada será explícita en el Planeamiento de la Asignatura, de acuerdo con el perfil de los estudiantes, los recursos disponibles y el contexto educativo.

VI. ESTRATEGIAS EVALUATIVAS

Procesos de producción grupales e individuales, pruebas individuales orales y/o escritas durante el desarrollo de las unidades con libro abierto. Además, espacios de diálogos e interpretaciones que los estudiantes realicen sobre los contenidos, así como debates y retroalimentación en casos necesarios. Se sumarán actividades que puedan ampliar el conocimiento con énfasis en el impacto del contenido en la profesión.

Las estrategias que se describen serán valoradas y en su conjunto aportarán para la calificación y promoción, siendo aplicadas según las normativas institucionales vigentes.

VII. MEDIOS AUXILIARES

Aula virtual, pizarrón, proyector, marcadores, equipo de audio, ordenadores, wifi, celulares, plataformas de videoconferencia, salas de chats, videos de clase, solucionarios, presentaciones electrónicas, herramientas asíncronas (grupo de correos, blogs, posts) y herramientas de IA para el procesamiento de lenguajes.

VIII. BIBLIOGRAFÍA

- Aho A., Lam M., Srthi R. & Ullman J. (2008) Compiladores. Principios, técnicas y herramientas. Segunda edición. México. PEARSON Education. ISBN 978-9702611332.
- Cooper K. & Torczon L. (2022) Engineering a Compiler. Tercera edición. USA. Morgan Kaufmann. ISBN 978-0128154120
- Thain D. (2020) Introduction to Compilers and Language Design. Segunda Edición. USA. Publicación independiente por el autor. ISBN 979-8655180260
- Camón J.E. (2020) Introducción a la Teoría de la Computación y el Diseño de Compiladores: Libro de Texto para estudiantes de Ingeniería de Sistemas y Tecnologías de la Información. Segunda edición. España. Editorial Académica Española. ISBN 978-620039833
- Ramos Serrano C. (2021) Entiende el diseño del Compilador. USA. Edición Kindle.
- Sandler N. Writing a C Compiler: Build a real programming language from scratch. (2023) Primera edición. USA. Publicación independiente por el autor. ISBN 978-1718500426
- Siguenza Ryan. (2023). Los Compiladores y su rol dentro de la inteligencia artificial. [Archivo PDF] https://www.researchgate.net/publication/370631406_LOS_COMPILADORES_Y_SU_ROL_DENTRO_DE_LA_INTELIGENCIA_ARTIFICIAL
- ChatGPT y los modelos de lenguaje en la práctica. Instituto de Ingeniería del Conocimiento. Universidad Autónoma de Madrid. España. (2022). https://www.iic.uam.es/procesamiento-del-lenguaje-natural/chatgpt-y-modelos-de-lenguaje